z/OS

Problem Management

Version 2 Release 1

SC23-6844-01

Note

Before using this information and the product it supports, read the information in "Notices" on page 301.

This edition applies to Version 2 Release 1 of z/OS (5650-ZOS) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Figures vii

Tables ix

About this information xi
Who should use this information xi
How to use this information xi
Where to find more information xii
  Information updates on the web xii

How to send your comments to IBM xiii
If you have a technical problem xiii

Summary of changes for z/OS Problem Management xv

Part 1. Problem management overview 1

Chapter 1. Introduction 3
Overview of problem resolution 3
Steps for diagnosing problems on z/OS 4
Gathering diagnosis data 5
Problem categories 6
Searching problem reporting databases 8
  Extracting problem symptoms and search arguments 9
  Formats for symptoms 9
  Searching for a known problem 10
  Steps for searching problem reporting databases 12
Determining the level of z/OS 13

Chapter 2. Common tools for problem determination 15
Messages 15
  BPXMTEXT for z/OS UNIX reason codes . . . 16
IPCS. 16
Logs. 17
Traces 19
Dumps 20
IBM Omegamon for z/OS Management Console . . 22
Sending problem documentation to IBM. 22
IBM documentation 23

Chapter 3. Best practices for large stand-alone dump 25
Using AutoIPL for stand-alone dumps 25
Planning a multivolume stand-alone dump data set 26
Creating the multivolume SADUMP 27
Defining a dump directory for large stand-alone and SVC dumps 27
Preparing the dump for further processing with IPCS COPYDUMP 27
Compressing data for faster transmission and analysis. 28
| Transmitting dump data to IBM 29
Setting up remote access 29
Testing your stand-alone dump operations 29
Automating the SADMP process 30
  Sample JCL for post-processing. 30
IBM System Test example. 31

Part 2. Runtime Diagnostics 33

Chapter 4. Runtime Diagnostics 35
How Runtime Diagnostics works 35
Enabling Runtime Diagnostics 36
  Running Runtime Diagnostics with mixed releases of z/OS. 38
  Reports from Runtime Diagnostics. 38
Runtime Diagnostics symptoms 39
Runtime Diagnostics messages 45
  Understanding the messages Runtime Diagnostics issues 45
  Test messages ignored by Runtime Diagnostics 46
Runtime Diagnostics DEBUG options. 46

Chapter 5. Messages that Runtime Diagnostics analyzes 51
BPX message analysis 51
IEA message analysis 52
IEE message analysis 53
IOS message analysis 54
IRA message analysis 54
IXC message analysis 55
IXL message analysis 57

Chapter 6. Using OPERLOG 59
Determining hardcopy medium settings 59
Setting up OPERLOG 59
  Steps for setting up OPERLOG 59

Part 3. Predictive Failure Analysis 63

Chapter 7. Predictive Failure Analysis overview and installation 65
Avoiding soft failures 65
Overview of Predictive Failure Analysis 65
  How PFA works with a typical remote check . . 66
  How PFA interacts with IBM Health Checker for z/OS 67
How PFA invokes Runtime Diagnostics 67
Migration considerations for PFA 69

Using the *migrate* or *new* parameters when running AIRSHREP.sh 69
How PFA uses the ini file. 71
Installing PFA 71
Steps for installing PFA 72
Updating the Java path 76

Chapter 8. Managing PFA checks . . . 79
Understanding how to modify PFA checks 79
MODIFY PFA, DISPLAY 81
MODIFY PFA, UPDATE 84
Using and configuring supervised learning 86

Chapter 9. Predictive Failure Analysis checks 89
PFA_COMMON_STORAGE_USAGE 89
PFA_ENQUEUE_REQUEST_RATE. 97
PFA_JES_SPOOL_USAGE 107
PFA_LOGREC_ARRIVAL_RATE 113
PFA_MESSAGE_ARRIVAL_RATE 120
PFA_SMF_ARRIVAL_RATE 136

Part 4. Diagnosing by problem type 153

Chapter 10. Diagnosing an abend. . . 155
Overview of an abend 155
Steps for diagnosing an abend. 158
Obtaining the abend and reason code 159
Steps for obtaining the abend code 159
Identifying the module and component 163
Steps for identifying the module and component 164
Searching the problem reporting databases . . . 166
Steps for searching the problem reporting databases. 167
Gathering additional problem data for abends . . 167
Steps for gathering additional data for abends 168

Chapter 11. Diagnosing a system hang or wait state 177
Overview of a hang or wait 177
Steps for diagnosing a system hang 178
Collecting the problem description 181
Steps for collecting the problem description . . 181
Diagnosing a hang or wait during IPL 182
Steps for diagnosing a hang or wait during IPL 182
Diagnosing an enabled wait state. 184
Steps for diagnosing an enabled wait state . . 184
Diagnosing a coded disabled wait state. 186
Steps for diagnosing a coded disabled wait state 186
Diagnosing a system partitioned from a sysplex because of status update missing 188
Steps for diagnosing a system partitioned because of status update missing 188
Searching the problem reporting databases . . . 188
Steps for searching the problem reporting databases. 188
Gathering additional data for hangs and waits . . 190
Steps for gathering messages and logrec for hangs 190

Chapter 12. Diagnosing a job or subsystem hang 193
Overview of a hang or wait 193
Steps for diagnosing a job or subsystem hang 193
Gathering additional data for a job or subsystem hang 195
Step for gathering additional data 195
Determining the status of a hung job or subsystem 195
Steps for determining the status of a hung job or subsystem 195
Determining if a job is waiting for resources . . . 196
Steps for determining if a job is waiting for resources 196
Determining address space dispatchability . . . 196
Steps for examining address space dispatchability 196
Examining the SRB status 197
Steps for examining the SRB status 197
Examining the TCB status 198
Steps for examining the TCB status 198
Examining why a job is not running. 200
Steps for examining why a job is not running 200

Chapter 13. Diagnosing a loop 203
Overview of a loop 203
Steps for diagnosing a loop. 205
Gathering additional data for a loop. 206
Steps for gathering loop data 206
Analyzing the dump to determine the type of loop 207
Step for analyzing the dump for loop type . . 207
Diagnosing a disabled loop. 208
Steps for diagnosing a disabled loop 208
Diagnosing an enabled loop 209
Steps for diagnosing an enabled loop 209
Diagnosing an excessive spin (spin loop) 212
Steps for diagnosing an excessive spin 212
Analyzing a logrec error record 214
Steps for analyzing a logrec error record . . . 214
Searching the problem reporting databases . . . 215
Steps for searching the problem reporting databases. 215

Chapter 14. Diagnosing an output problem. 217
Overview of analyzing output problems 217
Steps for diagnosing output problems 217
Collecting problem data for an output problem . . 219
Step for collecting problem data 219
Analyzing data set allocation for an output problem 219
Steps for analyzing data set allocation 219
Analyzing the inputs and outputs 220
Steps for analyzing the inputs and outputs . . 220
Analyzing installation exits for an output problem 221
Steps for analyzing installation exits. 221
Identifying the program or component 221
Steps for identifying the program or component 221

Searching the problem reporting databases for an output problem. 223
Step for searching the problem reporting database 223
Gathering additional data for output problems . . 223
Steps for gathering additional information for output problems 223
Reporting output problems to IBM 225

Chapter 15. Diagnosing a performance problem. 227
Overview of a performance problem 227
Steps for diagnosing a performance problem 227
Collecting data using commands 228
Steps for collecting data using DISPLAY . . . 228
Steps for using JES2 commands to collect data 230
Checking for resource contention or loop 231
Steps for checking resource contention 231
Searching the problem reporting database 232
Steps for searching the problem reporting databases. 232
Gathering additional data for performance problems 233
Steps for gathering additional information for performance problems 233
Analyzing a dump for performance problems . . 234
Steps for collecting and analyzing a dump for performance problems 234
Reporting performance problems to IBM 235

Part 5. Diagnosing component-specific problems . . . 237

Chapter 16. Catalog component operational problem determination . . 239
Catalog component-specific problems and recovery 239
Hang in the Catalog address space or in the user address waiting on a request to the Catalog address space 239
Damaged or broken catalogs 240
Slow performance in various address spaces due to requests to the catalog address space taking excessive time 241

Chapter 17. PDSE operational problem determination 243
PDSE specific problems 244
ABEND0F4 failures 244
MSGIGW038A possible PDSE problems . . . 245
PDSE data set corruption 247
Failure of the SMSPDSE or SMSPDSE1 address space 249

Chapter 18. RRS operational problem determination 251
Basic RRS problem determination functions . . . 251
Collecting documentation for RRS 251
Dumping RRS information 251
Important RRS CTRACE information 252
RRS recovery options. 252
RRS warm start 252
RRS cold start 253
RRS component-specific problems and recovery 253
RRS resource contention. 253
RRS suspended, waiting for signal from system logger 254
RRS log stream gap condition 255
RRS log stream data loss condition 256
RRS high processor usage 257
RRS address space hang. 257
RRS high storage usage 257
Resource manager is unable to start with RRS 259
Resource manager termination delay 259
RRS transaction hang. 260
| RRS severe error on RRS RM.DATA log stream,
| message ATR250E 261
Resolving RRS problems in a sysplex cascaded transaction environment. 262
Collecting documentation for a sysplex cascaded transaction environment. 262
Sysplex cascaded transaction hang 263
Sysplex cascaded transaction hang messages ATR246I and ATR247E 264

Chapter 19. System Data Mover (SDM) operational problem determination . . 265
SDM specific problems 265
ANTP0095I Unable to determine PPRC paths 265
ANTX5104E RC=0901 (XRC) 266
ANTX5104E RC=0647 REASON=0053 (XRC) . . 267
ANTX5104E RC=0647 REASON=0002 (XRC) . . 267
ANTAS00* ASIDs consuming excessive storage below 2GB 268
Converting to IR, RC=1017 269
Microcode issue impacting concurrent copy . . 269

Chapter 20. VSAM component operational problem determination . . 271
VSAM specific problems 271
VSAM Index Trap 271
Hang in VSAM record management code . . . 273
Loop in VSAM record management code . . . 273
Unexpected return codes from VSAM record management 274
Issues opening, closing, extending VSAM data sets. 275

Chapter 21. VSAM record-level sharing (RLS) operational problem determination 277
VSAM record-level sharing (RLS) specific problems 278
ABEND0F4 failures 278
HANG/WAIT in RLS/TVS. 279
SMSVSAM will not start up 280
Share Control Datasets not specified. 280

Part 6. Diagnosis reference material 283

Chapter 22. Diagnosis information for z/OS base elements and features . . . 285

Chapter 23. Reporting problems to IBM. 287
Software support service checklist 287
Automatic problem reporting 289
Invoking IPCS as a background job 289
Step for invoking IPCS as a background job . . 289

Chapter 24. Problem diagnostic worksheet 291

Part 7. Appendixes 295

Appendix. Accessibility 297
Accessibility features 297
Using assistive technologies 297
Keyboard navigation of the user interface 297
Dotted decimal syntax diagrams 297

Notices 301
Policy for unsupported hardware. 302
Minimum supported hardware 303
Trademarks 303

Index 305

Figures

1. IBMLink example 13
2. Status message that reports Runtime Diagnostics success 38
3. Status message that reports qualified success with events 39
4. Status message that reports qualified success with no events 39
5. Runtime Diagnostics ENQ contention report 40
6. Runtime Diagnostics HIGHCPU and LOCK report 41
7. Runtime Diagnostics LOOP and HIGHCPU report 42
8. HZR event for the z/OS UNIX file system latch contention and waiting threads record. . 43
9. HZR event for the GRS latch contention event record 44
10. Runtime Diagnostics critical message analysis 44
11. Message output for target system 46
12. LPAR view of the PFA components 66
13. Runtime Diagnostics report within the PFA message arrival rate check 68
14. Runtime Diagnostics report within the SMF arrival rate check. 69
15. Defining the file system 75
16. Common storage usage prediction report 93
17. Prediction report for enqueue request rate higher than expected - total jobs 101
18. Prediction report for enqueue request rate lower than expected - total jobs 102
19. Total system exception report: enqueue request rate higher than expected 103
20. Total system exception report: low enqueue request rate 103
21. JES spool usage persistent jobs exception report 110
22. JES spool usage no problem report 110
23. LOGREC arrival rate prediction report 116
24. Message arrival rate prediction report: tracked jobs higher than expected. 126
25. Message arrival rate prediction report: tracked jobs lower than expected 127
26. Message arrival rate prediction report: other persistent jobs with high arrival rate 128
27. Message arrival rate prediction report: other persistent jobs with low arrival rate 129
28. Message arrival rate prediction report: non-persistent jobs higher than expected . . 130
29. Message arrival rate prediction report: total system higher than expected 131
30. Message arrival rate prediction report: total system lower than expected. 132
31. SMF arrival rate: tracked jobs higher than expected 141
32. SMF arrival rate: tracked jobs lower than expected 142
33. SMF arrival rate: other persistent jobs higher than expected 143
34. SMF arrival rate: other persistent jobs lower than expected 144
35. SMF arrival rate: non-persistent jobs with high counts 145
36. SMF arrival rate: no problem and total system higher than expected 146
37. SMF arrival rate: no problem and total system lower than expected 147
38. Flowchart for abend analysis 159
39. RSMDATA SUMMARY report 174
40. Flowchart for system hang analysis 180
41. IPL statistics example 182
42. IPCS STATUS WORKSHEET report from a stand-alone dump 186
43. Flowchart for job or subsystem hang analysis 194
44. Flowchart for diagnosis of a loop 206
45. System trace table entry 209
46. Flowchart for output problem analysis 218
47. Flowchart for performance problem analysis 228
48. Output from the DISPLAY SLIP command 229
49. JCL for invoking IPCS as a background job 290

Tables

1. Problem indicators by type 7
2. Obtaining search arguments from SVC dump, stand-alone dump or SYSMDUMP using IPCS commands 9
3. Standardized symptom keyword list 10
4. ENQ checking: IBM-supplied address spaces 40
5. PFA EXCLUDED_JOBS file format 87
6. PFA_COMMON_STORAGE_USAGE check parameters 91
7. Multiplier notation used in values for PFA reports 93
8. PFA_ENQUEUE_REQUEST_RATE check parameters 98
9. Multiplier notation used in values for PFA reports 104
10. PFA_JES_SPOOL_USAGE check parameters 108
11. Multiplier notation used in values for PFA reports 111
12. PFA_LOGREC_ARRIVAL_RATE check parameters 114
13. Multiplier notation used in values for PFA reports 116
14. PFA_MESSAGE_ARRIVAL_RATE check parameters 122
15. Multiplier notation used in values for PFA reports 133
16. PFA_SMF_ARRIVAL_RATE check parameters 138
17. Multiplier notation used in values for PFA reports 148
18. Summary of IPCS dump subcommands by problem 172
19. Common wait states that occur during IPL 183
20. Common disabled wait states 187
21. Selecting the problem type for jSTATUS CPU output 189
22. Example of using the message prefix to identify the component 222
23. Severity levels and examples 288
24. What is the impact of your problem? 291
25. How is your system configured? 291
26. What are the external symptoms 291
27. What symptom information did you collect? 292
28. Which type of documentation did you obtain?. 292
29. What recovery actions did you attempt? 293

About this information

This information is designed to help you avoid potential problems and diagnose problems on z/OS®, its subsystems, its components, and problems in applications running under the system. Using this information, you can:

- Identify a potential problem
- Identify the problem type
- Determine the failing subsystem, component, job, or application
- Collect the correct data needed to diagnose the problem
- Develop a search argument and use it to search problem reporting databases
- Know the correct problem data to collect before reporting the problem to IBM® or the independent software vendor.

This information can help you determine why a problem occurred and where a problem occurred; it does not describe how to fix program instructions in your own code.

Who should use this information

This information is for anyone who diagnoses software problems that occur while running the operating system. This person is typically a system programmer for the installation. This information is also for application programmers who are testing their programs.

The level of detail at which this information is written assumes that the reader:

- Understands basic system concepts and the use of system services
- Codes in Assembler language, and reads Assembler and linkage editor output
- Codes Job Control Language (JCL) statements for batch jobs and cataloged procedures
- Understands the commonly used diagnostic tasks and aids, such as message logs, dumps, and Interactive Problem Control System (IPCS)

How to use this information

Use the procedures in this information to properly collect problem data, avoid potential problems, and diagnose failures.

If your installation does not want to debug the problem or does not have the source code involved in the problem, use the diagnosis procedures to collect the problem data needed for reporting the problem to IBM or other software vendors. The techniques described in this information are also relevant to non-IBM problems.

If your installation wants to debug the problem and has the source code, use the procedures to collect problem data and debug the problem. If the problem is in IBM code, report the problem to IBM. Where possible, IBM will debug the problem and provide a fix.

Where to find more information

Where necessary, this information references information in other documents, using cross-document links that go directly to the topic in reference using shortened versions of the document title. For complete titles and order numbers of the documents for all products that are part of z/OS, see *z/OS Information Roadmap*.

This information also references diagnosis books for specific components, see Chapter 22, "Diagnosis information for z/OS base elements and features," on page 285.

Information updates on the web

For the latest information updates that have been provided in PTF cover letters and Documentation APARs for z/OS, see the z/OS APAR book (http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/Shelves/ZDOCAPAR).

This information is updated weekly and lists documentation changes before they are incorporated into z/OS publications.

How to send your comments to IBM

We appreciate your input on this publication. Feel free to comment on the clarity, accuracy, and completeness of the information or provide any other feedback that you have.

Use one of the following methods to send your comments:

1. Send an email to mhvrcfs@us.ibm.com.
2. Send an email from the "Contact us" web page for z/OS (http://www.ibm.com/systems/z/os/zos/webqs.html).
3. Mail the comments to the following address:
   IBM Corporation
   Attention: MHVRCFS Reader Comments
   Department H6MA, Building 707
   2455 South Road
   Poughkeepsie, NY 12601-5400
   US
4. Fax the comments to us, as follows:
   From the United States and Canada: 1+845+432-9405
   From all other countries: Your international access code +1+845+432-9405

Include the following information:

- Your name and address.
- Your email address.
- Your telephone or fax number.
- The publication title and order number:
  z/OS Problem Management
  SC23-6844-01
- The topic and page number that is related to your comment.
- The text of your comment.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute the comments in any way appropriate without incurring any obligation to you.

IBM or any other organizations use the personal information that you supply to contact you only about the issues that you submit.

If you have a technical problem

Do not use the feedback methods that are listed for sending comments. Instead, take one of the following actions:

- Contact your IBM service representative.
- Call IBM technical support.
- Visit the IBM Support Portal at z/OS support page (http://www.ibm.com/systems/z/support/).

Summary of changes for z/OS Problem Management

This information includes terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Summary of changes for z/OS Problem Management as updated June 2014

New information about component-specific problem determination, Part 5, "Diagnosing component-specific problems," on page 237, was missing from the online version of this information. This is now corrected thanks to a thoughtful reader!

Summary of changes for z/OS V2R1 Problem Management

- See Part 5, "Diagnosing component-specific problems," on page 237 for new information about component-specific problem determination.
- The default value for trackedmin is changed to three in the following Predictive Failure Analysis (PFA) checks:
  - "PFA_ENQUEUE_REQUEST_RATE" on page 97
  - "PFA_MESSAGE_ARRIVAL_RATE" on page 120
  - "PFA_SMF_ARRIVAL_RATE" on page 136
- For PFA, changes to the Java™ settings.
- For PFA, clarified when to use the *migrate* and *new* parameters for AIRSHREP.sh.
- **Deleted:** PFA_FRAMES_AND_SLOTS_USAGE

For specific enhancements made to z/OS Version 2, Release 1 (V2R1), see the following publications:

- *z/OS Summary of Message and Interface Changes*
- *z/OS Introduction and Release Guide*
- *z/OS Planning for Installation*
- *z/OS Migration*

Part 1. Problem management overview

Before you begin diagnosing problems, or using PFA or Runtime Diagnostics, it is important to understand the basics and best practices of problem management covered in this section.

Chapter 1. Introduction

If a problem occurs, this information can help you determine what happened, why it happened, and how to find the fix or report the problem to IBM.

This information can also help you avoid problems and soft failures, find specific information about tools and service aids, and locate diagnosis information in the z/OS library. For details, see:

- Chapter 2, "Common tools for problem determination," on page 15
- Part 3, "Predictive Failure Analysis," on page 63
- Chapter 22, "Diagnosis information for z/OS base elements and features," on page 285.

This chapter covers these topics:

- "Overview of problem resolution"
- "Gathering diagnosis data" on page 5
- "Problem categories" on page 6
- "Searching problem reporting databases" on page 8
- "Extracting problem symptoms and search arguments" on page 9
- "Formats for symptoms" on page 9
- "Determining the level of z/OS" on page 13

Overview of problem resolution

Typical z/OS problems are classified by the following symptoms:

- **Abend** – an error or abnormal end of a program or job.
- **Wait or Hang** – a coded wait state is loaded or the system or a job appears hung or does not complete.
- **Loop** – the system or program executes infinitely typically using large or higher amounts of processor resource.
- **Incorrout** – there is incorrect or missing output from a program or job.
- **Performance** – processing is using too much system resource and impacting other parts or users of the system, or processes are taking too long.
- **Message** – an error is reported through a message to the operator or in a log.

When an error occurs, z/OS provides various forms of diagnosis information that contains symptoms. These symptoms can help you with diagnosis and be used in problem source identification (PSI). PSI is the determination of what caused the error based on answers to these questions:

- Why was an abend issued?
- What caused storage to be exhausted?
- Why is the job hung?
- What is causing the job to loop?

This document is designed to help answer these questions and others and make efficient use of your time when diagnosing, searching, and reporting a problem to IBM.

PSI is useful even if the root cause of the problem is not identified. During the process, information and symptoms are gathered to check for a known problem or report a new problem. To ease and expedite problem identification, it is important to provide all the background information available. This includes information about:

- Hardware involved
- System and application software levels
- External symptoms
- Problem impact
- Diagnostic data produced

By providing sufficient information during the first call to IBM or the individual software vendor, you might avoid having to re-create the problem.

The problem diagnostic worksheet contains key information needed to expedite problem resolution. If you are an experienced z/OS system programmer, use the Chapter 24, "Problem diagnostic worksheet," on page 291 as a reminder of the important information to gather and report. For example:

- Extract the diagnostic data and symptoms
- Build a search argument
- Search for a known problem
- Gather available diagnostic information
- Report a new problem.

Steps for diagnosing problems on z/OS

To diagnose a problem, follow these steps:

1. When the problem occurs, gather all the available diagnosis information for problem. Use the Chapter 24, "Problem diagnostic worksheet," on page 291 as a template for recording data. This might also include your internal problem report describing external symptoms, what might have triggered the problem, and what was done to recover, including the following types of diagnostic information:
   - Dumps
   - Traces
   - Error messages
   - SYS1.LOGREC entries
   - External symptoms
   - Hardware devices
   - Processor models
   - Any other information

   These topics can help you collect the data more effectively:
   - "Gathering diagnosis data" on page 5
   - "Problem categories" on page 6
2. After the problem type is identified, see these diagnosis procedures to identify the source and extract symptoms:
   - Chapter 10, "Diagnosing an abend," on page 155
   - Chapter 11, "Diagnosing a system hang or wait state," on page 177
   - Chapter 12, "Diagnosing a job or subsystem hang," on page 193

- Chapter 13, “Diagnosing a loop,” on page 203
- Chapter 14, “Diagnosing an output problem,” on page 217
- Chapter 15, “Diagnosing a performance problem,” on page 227

3. While using the procedure, build a search argument from the data collected. See “Extracting problem symptoms and search arguments” on page 9 for more information.
4. Perform the search. Keep in mind that you might refine your search with more data from the problem. See “Searching problem reporting databases” on page 8 for more information.
5. If the problem is not found in a database, report it as a new problem providing the documentation and information collected in Chapter 24, “Problem diagnostic worksheet,” on page 291. See Chapter 23, “Reporting problems to IBM,” on page 287 for more information.

Tip: Sometimes information is found that is useful in routing the problem to the right place. For example, if the problem is an ABEND0C4 in module XYZ and your search shows multiple hits for ABEND0C4 and XYC and information about product from another company, contact that company or search that company's problem reporting databases.

Gathering diagnosis data

It is important to gather the external symptoms and know the impact to the system or sysplex to define the scope of the problem. There can be many symptoms.

For example: Shortly after JOB A started, a dump was produced for an ABEND0C4, the system went into a WAIT064, and was partitioned from the sysplex.

1. Start with diagnosis of the ABEND0C4, which appears to be the trigger, but also understand the cause of the WAIT064 and why the job failure resulted in a system outage. It is important to check for known problems for both symptoms.
2. Next, gather all the diagnosis data available from the time frame the problem or failure occurred.

To identify a system problem, look at the diagnostic data such as:

- External symptoms and the initial problem report. Look for indications, which can include:
  - Messages
  - Job hang
  - System hang
  - High processor usage
  - Incorrect output
  - Dumps produced
  - System slowdown
  - Jobs not starting
- SVC dumps produced as indicated by these messages:

  IEA794I

  ```
  IEA794I SVC DUMP HAS CAPTURED: DUMPID=dumpid REQUESTED BY JOB (*MASTER*)
          DUMP TITLE=dump-title
  ```

IEA911E

```
IEA911E {COMPLETE|PARTIAL} DUMP ON SYS1.DUMPnn

            DUMPid=dumpid REQUESTED BY
            JOB (jobname)
            FOR ASIDS(id,id,...)
            [REMOTE DUMPS
            REQUESTED | REMOTE
            DUMP FOR SYSNAME: sysname]
            INCIDENT TOKEN:incident-token
            [SDRSN =
            vvvvvvvv wwwwwwww xxxxxxxx
            zzzzzzzz]
            [reason-text]
            [ERRORID = SEQyyyyyy
            CPUzz ASIDasid
            TIMEhh.mm.ss.f]
            [TSOID = tsoid]
            [ID = uuuuuuuuuu]
```

IEA611I

```
IEA611I {COMPLETE|PARTIAL} DUMP ON dsname

            text
```

- SYS1.LOGREC data set, which is a repository for information about hardware and system-level software errors.
- Logs from the time frame the problem occurred. This can include SYSLOG, OPERLOG, job log(s), and others.
- Traces associated with the problem.

Tip: After a problem has been isolated to a particular component, query using the TRACE command to see if detailed component traces or GTF was active at the time. For example, if the error is announced by ISGxxxx messages, then check for SYSGRS CTRACE. The message prefix (three or more characters) determines the component owner. See the topic about identifying modules, components, and products in *z/OS MVS Diagnosis: Tools and Service Aids*.

Problem categories

The problem indicator table contains examples of indicators. Some problems might need to be investigated using more than one diagnostic procedure to find the cause. If there are several indicators, look for the earliest problem that caused the other problems.

For example, you find several abends and a wait state, look for the earliest abend code and begin diagnosis there.

Table 1. Problem indicators by type. Where to find problem indicators by type

Problem type	Indicator	System action	System programmer action
Abend See Chapter 10, "Diagnosing an abend," on page 155.	SVC dump taken and a record of the error (in Logrec)	Produces dump	Review dump to determine if further diagnosis is required
	Message received indicating a system or user abend	Produces record of error	Review response of system message to determine the impact of the abend on the installation.
	An ABEND dump is produced	Continue processing	Review the dump to determine if further diagnosis is required.
	• SVC dump produced • Error recorded to SYS1.LOGREC • Error message issued • SYSUDUMP, SYSABEND or CEEDUMP produced	System actions are the same as the indicators listed previously. **Note:** The system might also initiate recovery actions. See SYSLOG and component trace to determine what these recovery action were. Some recovery actions can cause data to be lost, requiring the installation to resubmit jobs or transactions.	1. Use IPCS to do problem diagnosis on the dump. 2. Look up abend and reason code recorded for more information about error. 3. Look up the message to gather more information about cause of the error and the system programmer action to correct. 4. Review the dump to do problem diagnosis.
Job hang/wait or loop Chapter 12, "Diagnosing a job or subsystem hang," on page 193	Job does not end, no further output is produced, and the job can or cannot be CANCEL'ed or FORCE'd	No response	Use the DUMP command to obtain an SVC dump of the hung job. If the DUMP is not successful, consider taking a stand-alone dump.
System hang or wait Chapter 11, "Diagnosing a system hang or wait state," on page 177	Disabled wait indicated on the HMC and wait state message issued	The system issues a wait state message and loads a disable wait state PSW. The system might load the following into the PSW: X'070E0000 00000000'	Take a stand-alone dump.
	Many jobs are hung in the system	Resource contention	Enter the DISPLAY GRS,C command to check for ENQ resource and latch contention and take a dump of the holder of the resource including SDATA=GRSQ. **Note:** Use the DISPLAY GRS,ANALYZE command to aid in the discovery of blockers in the system.
	No response to system or subsystem commands entered	No response	Partition the system from the sysplex and take a stand-alone dump.

Table 1. Problem indicators by type (continued). Where to find problem indicators by type

Problem type	Indicator	System action	System programmer action
Loop Chapter 13, "Diagnosing a loop," on page 203	High processor resource being consumed locking out other work; Excessive spin detected with IEE178I or ABEND071 issued, or both.	ABEND071 issued in an attempt to stop the looping program	Use an online monitor, such as Resource Measurement Facility™ RMF™ or IBM OMEGAMON® z/OS Management Console, to determine whether the problem originates from a high priority job in normal processing or from a problem.
	A job is using a high percentage of central processor storage	Processing degrades	Use an online monitor, such as RMF, to determine whether the problem originates from a high priority job in normal processing or from a problem.
Enabled wait or performance degradation Chapter 15, "Diagnosing a performance problem," on page 227	System processing slows.	Processing degrades	Use an online monitor, such as RMF, to determine where the problem originates.
	There is a series of WAIT messages followed by a burst of activity	Processing continues	Use an online monitor, such as RMF, to determine where the bottleneck is occurring.
Output problem Chapter 14, "Diagnosing an output problem," on page 217	Job output is missing or is incorrect.	Processing continues	Use GTF or SLIP to trace input and output.

Searching problem reporting databases

While you are diagnosing a system problem, you will collect data about that problem:

- What was the abend code?
- What did the registers and PSW contain at the time of error?
- What is the failing module or CSECT?
- What components or products were involved with the error?

The answers to these questions are the material for a search argument. A search argument is a list of symptoms for a problem. A search argument is also called a symptom string.

This section contains these topics:

- "Extracting problem symptoms and search arguments" on page 9 describes how to develop a search argument while you are performing diagnosis.
- "Formats for symptoms" on page 9 distinguishes between the types of symptom formats.
- "Searching for a known problem" on page 10 lists the symptoms used in search arguments.
- "Steps for searching problem reporting databases" on page 12 explains the steps to begin your search.

Extracting problem symptoms and search arguments

Obtain search arguments from an SVC dump, SYSMDUMP dump, or stand-alone dump by using IPCS subcommands.

For most problems, use three to five symptoms in the search argument. If the first search produces no matches, remove some symptoms and search again. If the first search produces too many matches, add one or more symptoms and search again. Also, try different symptoms. Searching is an iterative process.

The following are suggestions for selecting symptoms:

- Start with the symptom keyword, for example ABEND0C4, WAIT or LOOP and the module or CSECT name and component ID. Add or remove symptoms from the search argument depending on the number of matches produced.
- Symptoms about data areas are useful for identifying a problem. Use the names of a data area and the incorrect field in the data area as symptoms.
- If searching does not produce a match, remove some symptoms or use different symptoms and try again.

Table 2. Obtaining search arguments from SVC dump, stand-alone dump or SYSMDUMP using IPCS commands

IPCS subcommand	Dump output heading
STATUS FAILDATA	Search Argument Abstract
VERBEXIT DAEDATA	DUMP ANALYSIS AND ELIMINATION (DAE)
VERBEXIT LOGDATA	SEARCH ARGUMENT ABSTRACT
VERBEXIT SYMPTOM	Primary Symptom String

Build a free-format search from the IPCS reports by extracting:

- CSECT name
- Abend code
- Reason code
- Component id

Ensure use of the standardized symptom keyword. For example, system abend code presented as S005C in the IPCS ST FAILDATA report is converted to ABEND05C and reason code PRCS/00000214 is converted to RSN00000214. Table 3 on page 10 contains a list of the standardized symptoms.

Formats for symptoms

Symptom strings or search arguments are presented in several different formats. They include:

- Free-Format symptom: is commonly used to search on the Internet and in IBMLINK for a known problem. The symptoms in the freely formatted string are standardized (see Table 3 on page 10).

  For example:
  - `ABEND0C4 5752SCXCF IXCS2STB`
  - A module CSECT name: `IEAABCD`
- RETAIN® symptom string: Use RETAIN symptoms:
  - With a tool such as Info/Management to search the RETAIN database

- When reporting a problem to IBM
- In descriptions of problems in APARs and program temporary fixes (PTF)

RETAIN symptoms are also called *structured symptoms* and *failure keywords*. An example of a module CSECT name as a RETAIN symptom is: `RIDS/IEAABCD`

The table of RETAIN and MVS™ symptoms is in the topic on specifying symptoms in *z/OS MVS Diagnosis: Reference*.

- MVS symptom, is used by dump analysis and elimination (DAE) when determining if a dump is a duplicate of a previous dump; MVS symptoms are not used for searching problem databases. These symptoms are contained in the DAE data set. An example of a module CSECT name as an MVS symptom is: `CSECT/IEAABCD`. For a complete example, see the topic on dump analysis and elimination (DAE) in *z/OS MVS Diagnosis: Tools and Service Aids*.

Searching for a known problem

Use the following search argument of standardized symptoms when performing a search for a known problem in the Technical Support database, IBMLink or when reporting a problem to IBM:

- Always concatenate a number to a word when it modifies that word (for example, SVC13, ABEND0C4)
- Use the word "missing" whenever messages do not appear as expected
- Never abbreviate system commands
- Include = in a search argument with no blanks on either side (for example, DISP=MOD is correct)
- Do not use hyphens in search arguments (for example, SC231234 is the correct way to enter a publication number).

The standardized symptom table describes common symptom keywords to use while doing a search for a known problem or reporting a problem to IBM:

Table 3. Standardized symptom keyword list

Free-format symptom	Problem data
ABENDxxx	Any system abend except JES3; where xxx is the hexadecimal value of the abend code, always 3 digits including leading zeros
ABENDDMxxx	JES3 abend; where xxx is the hexadecimal value of the abend code, always 3 digits including leading zeros
ABENDUxxxx	User abend; where xxxx is the user abend code
AMODE31	Program running in AMODE 31 (31-bit mode)
AMODE64	Program running in AMODE 64 (64-bit mode)
ARnn	Access register; where nn is the decimal register number without leading zeros
CRnn	Control register; where nn is the decimal register number without leading zeros
D/Txxxx	Device type; where xxxx is the device number
DATASET	Data set
DEQ	Dequeue
DESCCODEnn	WTO descriptor code; where nn is the decimal value of the code, 1-13, without leading zeros
ENQ	Enqueue
ERRNO2n...n	Where n...n is the 4 byte hexadecimal value of the errno2

Table 3. Standardized symptom keyword list (continued)

Free-format symptom	Problem data
ERRNOJRn...n	Where n...n is the 4 byte hexadecimal value of the errnojr
ERRNOnnn	Where nnn is the errno in decimal
HANG	Always include this form of the word
I/O	Input Output
KEYn	PSW Key or Storage Key (in hex)
KEYnn	PSW Key or Storage Key (in dec)
LATCH#nn	Where nnn is decimal latch number without leading zeros (for example: LATCH#2)
LOOP	Always include this form of the word
LPAR	Logical Partition (PR/SM™)
LU62	Logical Unit 6.2 protocol
MIH	Missing interrupt handler
MSGxxxx	Any message except JES2 messages; where xxxx is the complete message id of any length
MSGHASPxxx	JES2 messages; note that the '$' prefix has been removed and xxx is the message id of any length
OVERLAY	Storage overlay; always include this form of the word
PAGEFIX	Page-Fix
PICxx	Program Interrupt Code associated with ABEND0Cx; where xx is the interrupt code, always 2 digits with leading zeros
Rxxx	Release level; where xxx is the product release level
RCnn	Return code; where nn is decimal or hexadecimal and at least two digits
REGnn	General purpose register; where nn is the decimal register number without leading zeros
ROUTCODEnnn	WTO route code; where nnn is the decimal value of the code, 1-128, without leading zeros
RSNxxx	Reason code; where xxx is the hexadecimal reason code of any length
SADMP	Stand-alone dump
SIGxxxx	Where xxx is the name of the signal (for example: SIGTERM)
SIO	Start Input Output
SMFTYPEnnn	SMF type records; where nnn is the decimal value of the record, 0-255, without leading zeros
SUBTYPEnnn	SMF subtype records; where nnn is the decimal value of subtype, 0-255, without leading zeros. Also make sure the SMFTYPEnnn is included
SPnnn	Subpool number; where nnn is the decimal value of subpool, 0-255, with no leading zeros
SVCnnn	Supervisor Call; where nnn is the decimal value of the SVC, 0-255, with no leading zeros
VOLSER	Volume serial
WAIT	Always use this form of the word

Table 3. Standardized symptom keyword list (continued)

Free-format symptom	Problem data
WAITxxx	System wait state; where xxx is the hex value of the wait code, always 3 digits including leading zeros
Z/ARCHITECTURE	64-bit mode

Related information:

- See *z/OS MVS IPCS Commands* for the subcommands.
- See *z/OS MVS Diagnosis: Reference* for logrec record formats.
- For formatting of logrec records, see Recording logrec error records in *z/OS MVS Diagnosis: Tools and Service Aids*.

Steps for searching problem reporting databases

About this task

Often the problem has already been reported and fixed. Using the symptom string or search argument extracted, you can do a search of the technical database associated with the product identified.

1. Go to one of these Web sites:
   - IBM technical support for z/OS at: www.ibm.com/systems/z/os/zos/support/.
   - IBMLink at ibm.com/ibmlink/link2.
2. Select the documents you want to search for problem related information. For example, select APARs, FAQs, Technotes, or Flashes.
3. If the Internet is not available at your installation, call the IBM support center and ask them to do the search for you.

Search arguments are used to search problem reporting databases. If the problem being diagnosed was already reported and the symptoms entered into the database, the search will produce a match.

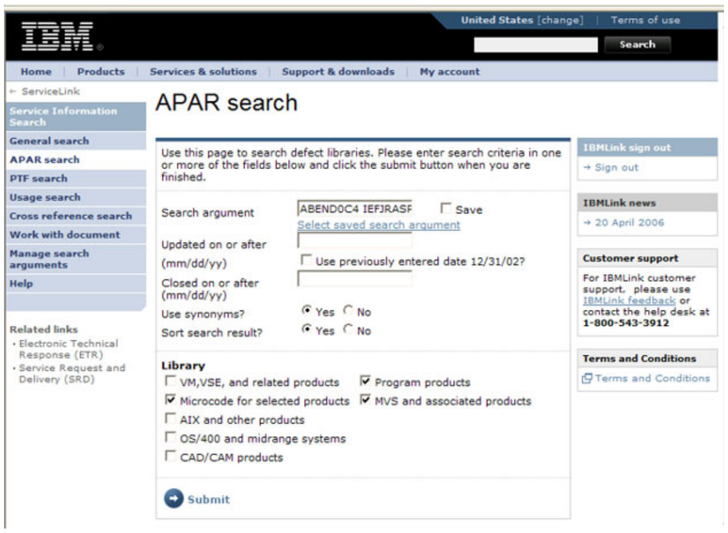
IBMLink & zSeries Software Support contains an overview of IBMLink services.

Example

If your installation has access to IBMLink, an interactive online database program, you can:

- Search for an existing authorized program analysis report (APAR) that is similar to your problem.
- Search for an available program temporary fix (PTF) for the existing APAR.
- Order the PTF if it is available.
- Create an Electronic Technical Response (ETR) problem report to get assistance from a service representative.

The following IBMLink example uses a free-format symptom value of `ABEND0C4 IEFJRASP` in the search entry.



APAR search

Use this page to search defect libraries. Please enter search criteria in one or more of the fields below and click the submit button when you are finished.

Search argument: ABENDOC4 IEFJRASF ☐ Save
Select saved search argument
Updated on or after (mm/dd/yy):
☐ Use previously entered date 12/31/02?
Closed on or after (mm/dd/yy):
Use synonyms? [✓ Yes] No
Sort search result? [✓ Yes] No

Library
☐ VM,VSE, and related products
☑ Program products
☑ Microcode for selected products
☑ MVS and associated products
☐ AIX and other products
☐ OS/400 and midrange systems
☐ CAD/CAM products

Customer support
For IBMLink customer support, please use IBMLink feedback or contact the help desk at 1-800-543-3912

Figure 1. IBMLink example

Determining the level of z/OS

When you report problems to the IBM Support Center, you must provide the name and level of the operating system or systems. If you have communication with your console, you can use the DISPLAY command or you can query the dump using IPCS.

Use the console command DISPLAY IPLINFO or the IPCS command IPLINFO in a dump to obtain the following information:

- The date and time of the IPL
- The release level of the system
- The license value for the system
- The contents of parmlib members IEASYSxx and IEASYMxx
- LOADxx information used for the IPL
- The architecture level of the IPL
- The IODF (input/output definition file) device
- The IPL device and volume serial
- The status of MTL (manual tape library) tape devices.

For example:

```
D IPLINFO
    IEE254I  11.14.07 IPLINFO DISPLAY 350
     SYSTEM IPLED AT 01.15.39 ON 11/01/2007
     RELEASE z/OS 01.09.00    LICENSE = z/OS
     USED LOAD08 IN SYS0.IPLPARM ON ACB2
     ARCHLVL = 2   MTLSHARE = N
     IEASYM LIST = (X6,U6,0L,R8)
     IEASYS LIST = (ST,LN) (OP)
     IODF DEVICE ACB2
     IPL DEVICE 3C2A VOLUME D83EL
```

If you cannot communicate through the console, use IPCS to perform the following steps to determine which system or systems you are using:

1. Use the IPCS subcommand CBFORMAT with the communications vector table (CVT) control block to determine the product level.

   In the **CBFORMAT CVT** example output, the *PRODN* field indicates an MVS operating system level of *SP7.0.7* and the *PRODI* field indicates the FMID as *HBB7720*.

   ```
   CVT: 00FD48A0
      -0028  PRODN.... SP7.0.7   PRODI.... HBB7720   VERID....
      -0006  MDL...... 2084      RELNO.... 038
      +0000  TCBP..... 00000218  0EF00.... 00FEA3EC  LINK..... 00FD481C
      +000C  AUSCB.... 00FD57E8  BUF...... 00000000  XAPG..... 00FE0380
   ```

2. Determine if the system is running as a uniprocessor or multiprocessor. In the IPCS STATUS WORKSHEET output `PROCESSOR RELATED DATA`, find the MVS Diagnostic Worksheet:

   ```
                                MVS Diagnostic Worksheet

   Dump Title: W059 SLIP TRAP


   CPU Model 2084 Version 00 Serial no. 220CBE Address 00
   Date: 09/15/2006     Time: 09:33:32.124515 Local


    CSD  Available CPU mask: FFC0  Alive CPU mask: FFC00000  00000000
         Number of active CPUs: 0000000nn
   ```

   In `0000000nn`, the variable *nn* indicates the number of processors running.

   In this output example, there are ten active processors.

   ```
   CSD  Available CPU mask: FFC0  Alive CPU mask: FFC00000  00000000
     No. of active CPUs: 0000000A
   ```

- See *SMP/E for z/OS User's Guide* for using SMP/E.
- See *z/OS MVS IPCS Commands* for more information about the CBFORMAT subcommand.
- See *z/OS MVS Data Areas* in the z/OS Internet library (http://www.ibm.com/systems/z/os/zos/bkserv/) for the format of the SCCB.

Chapter 2. Common tools for problem determination

z/OS contains many tools and service aids to assist you when a problem does occur. The more you know about these tools and service aids, the easier it is for you to diagnose problems and send data to IBM. This chapter provides an overview of the some commonly used tools and where to find more information about each of them.

This chapter covers:

- "Messages"
  - "BPXMTEXT for z/OS UNIX reason codes" on page 16
- "IPCS" on page 16
- "Logs" on page 17
- "Traces" on page 19
- "Dumps" on page 20
- "IBM Omegamon for z/OS Management Console" on page 22
- "Sending problem documentation to IBM" on page 22
- "IBM documentation" on page 23

Messages

z/OS issues messages from z/OS elements, features, program products, and application programs running on the system. The system issues messages in different ways and to different locations:

- Automated messaging services automatically react to certain messages.
- Most messages are issued through WTO and WTOR macros to one of these locations:
  - Console
  - Hard-copy log
  - Job log
  - SYSOUT data set

  Routing codes determine where the messages are displayed or printed. The routing codes for messages issued by the operating system are included with each message.
- Unless specified otherwise, messages, in general, go to the system log (SYSLOG).
- Dump messages are issued through the dumping services routines and are found in:
  - SVC dumps, stand-alone dumps, or SYSMDUMP ABEND dumps formatted by the interactive problem control system (IPCS)
  - Trace data sets formatted by the interactive problem control system (IPCS)
  - ABEND dumps or SNAP dumps produced by the dumping services

  In dump or trace data sets formatted by IPCS, the messages are shown on a terminal or in a printed dump.
- Some messages are issued through DFSMS/MVS access methods directly to one of these locations:
  - Output data set

– Display terminal

For z/OS V1R13 and earlier releases, find a message quickly using a web search engine or LookAt: www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/.

BPXMTEXT for z/OS UNIX reason codes

BPXMTEXT is shipped in SYS1.SBPXEXEC and must be in SYSEXEC or SYSPROC to use. It can run from TSO, IPCS, or the z/OS UNIX Shell. You can use BPXMTEXT to interpret errnojr values from zFS (reason code qualifier=EFxx), TCP/IP (reason code qualifier=7xxx), and the C/C++ run-time library (reason code qualifier=Cxxx).

To determine the meaning of reason codes for z/OS UNIX and zSeries File System (zFS), use **BPXMTEXT**.

From TSO, enter TSO BPXMTEXT *xxxxxxxx*, where *xxxxxxxx* is the reason code.

Here's an example:

```
EQQPH35I: EQQPH35I BPX1ATX FAILED WITH RC=0157, RSN=0B1B03AC
```

To find the meaning of `RSN=0B1B03AC` from TSO, enter:

```
BPXMTEXT 0B1B03AC
```

You get this result:

```
BPXPREXC date JRAuthCaller: The caller of this service is authorized.
Authorized callers are not permitted to load or call unauthorized programs.
Action: System key, supervisor state, or APF authorized callers cannot load
or call unauthorized programs.
```

IPCS

Interactive Program Control System (IPCS) is a powerful diagnostic tool in the MVS system that aids the diagnosis of software failures. IPCS provides formatting and analysis support for dumps and traces produced by MVS, other program products, and applications that run on MVS.

Dumps (SVC dump, stand-alone dump, SYSMDUMP) and traces (system trace, GTF trace, and CTRACE) need to be formatted before analysis can begin. IPCS provides the tools to format dumps and traces in both an online and batch environment. IPCS provides you with commands that will let you interrogate specific components of the operating system and allow you to review storage locations associated with an individual task or control block. IPCS allows you to quickly review and isolate key information that will assist with your problem determination process.

Using dump and trace data sets and, in some cases, active storage as a source IPCS analyzes information and produces reports that can be viewed at a Time Sharing Option Extensions (TSO/E) terminal or can be printed.

Related information: For complete information about IPCS, see these procedures and documents:

- "Invoking IPCS as a background job" on page 289
- *z/OS MVS IPCS User's Guide*
- *z/OS MVS IPCS Commands*

- *z/OS MVS IPCS Customization*
- Using IPCS to format component dump data in *z/OS MVS Diagnosis: Reference*.

Logs

Do not overlook log data — it should be the first place to look when reviewing a problem. z/OS communicates problems through messages that it writes to logs. Six logs contain the primary sources of problem data:

SYSLOG

The SYSLOG is a SYSOUT data set provided by the job entry subsystem (either JES2 or JES3). SYSOUT data sets are output spool data sets on direct access storage devices (DASD). An installation should print the SYSLOG periodically to check for problems. The SYSLOG consists of:

- All messages issued through WTL macros
- All messages entered by LOG operator commands
- Typically, the hard-copy log
- Any messages routed to the SYSLOG from any system component or program

View SYSLOG through the Spool Display and Search Facility (SDSF) using the LOG option. A small amount of the SYSLOG is also stored in memory and is included when an address space is dumped. This is referred to as master trace (MTRACE) data and can be accessed from IPCS using the **VERBX MTRACE** command.

This example shows the MVS SYSLOG without time stamps.

```
STC18213 00000090 $HASP100 BPXAS ON STCINRDR
STC18213 00000090 $HASP373 BPXAS STARTED
STC18213 80000010 IEF403I BPXAS - STARTED - TIME=13.36.36 - ASID=001F - SC53
STC16316 00000291 IST663I IPS SRQ REQUEST FROM ISTAPNCP FAILED, SENSE=08570002
     111 00000291 IST664I REAL OLU=USIBMSC.S52TOS48 REAL DLU=USIBMSC.S48TO
     111 00000291 IST889I SID = ED0385CAAEEAAF28
     111 00000291 IST264I REQUIRED RESOURCE S48TOS52 NOT ACTIVE
     111 00000291 IST314I END
STC16352 00000291 IST663I IPS SRQ REQUEST FROM ISTAPNCP FAILED, SENSE=087D0001
     883 00000291 IST664I REAL OLU=USIBMSC.S52TOS48 ALIAS DLU=USIBMSC.S48TO
     883 00000291 IST889I SID = ED0385CAAEEAAF28
     883 00000291 IST314I END
STC28215 00000291 IST663I IPS SRQ REQUEST TO ISTAPNCP FAILED, SENSE=08570002 86
     864 00000291 IST664I REAL OLU=USIBMSC.S52TOS48 ALIAS DLU=USIBMSC.S48TO
     864 00000291 IST889I SID = ED0385CAAEEAAF28
     864 00000291 IST264I REQUIRED RESOURCE S48TOS52 NOT ACTIVE
     864 00000291 IST891I USIBMSC.SC48M GENERATED FAILURE NOTIFICATION
     864 00000291 IST314I END
```

Job log

Messages sent to the job log are intended for the programmer who submitted a job. Specify the system output class for the job log in the MSGCLASS parameter of the JCL JOB statement.

OPERLOG

Operations log (OPERLOG) is an MVS system logger application that records and merges messages about programs and system functions (the hardcopy message set) from each system in a sysplex that activates OPERLOG.

In SDSF the OPERLOG panel displays the merged, sysplex-wide system message log. You can use the parameters of the LOG command to select the OPERLOG panel or the single-system SYSLOG panel. The OPERLOG panel displays the data from a log stream, a collection of log data used by the MVS System Logger to provide the merged, sysplex-wide log.

Logs

An individual product has its own log file. These log files might contain data that is valuable when diagnosing a problem. It is particularly important to look for events that precede an actual abend or failure because the problem, in many cases, will have been caused by a previous action.

This example shows the SYSOUT data sets that might be associated with a CICS® address space:

```
NP DDNAME    StepName ProcStep DSID Owner
   JESJCLIN                       1 CICSTS
   JESMSGLG   JES2                2 CICSTS
   JESJCL     JES2                3 CICSTS
   JESYSMSG   JES2                4 CICSTS
   $INTTEXT   JES2                5 CICSTS
   CAFF       SCSCPAA1          101 CICSTS
   CINT       SCSCPAA1          103 CICSTS
   DFHCXRF    SCSCPAA1          104 CICSTS
   COUT       SCSCPAA1          105 CICSTS
   CEEMSG     SCSCPAA1          106 CICSTS
   CEEOUT     SCSCPAA1          107 CICSTS
   PLIMSG     SCSCPAA1          108 CICSTS
   CRPO       SCSCPAA1          109 CICSTS
   MSGUSR     SCSCPAA1          110 CICSTS
```

The key SYSOUT data sets to review for problem determination data are the JESMSGLG and MSGUSR data sets. The CEEMSG and CEEOUT data sets will contain Language Environment® (LE) problem data typically associated with application problems.

The CICS JESMSGLG SYSOUT data set includes information related to CICS startup and errors related to system problems, not specifically transaction related.

Logrec Error Recording

Log recording (logrec) log stream is an MVS System Logger application that records hardware errors, selected software errors, and symptom records across the sysplex.

Use the records in the logrec data set or the logrec log stream as additional information when a dump is produced. The information in the records can point you in the right direction while supplying you with symptom data about the failure. Use the Environmental Record, Editing, and Printing program (EREP) to:

- Print reports about the system records
- Determine the history of the system
- Learn about a particular error

Logrec data is written to the SYS1.LOGREC data set and is also written to internal storage that is included in a dump. The SYS1.LOGREC data set can be interrogated using the ICFEREP1 program, or if the abend has triggered a dump, the EREP data can be reviewed using the IPCS **VERBX LOGDATA** command. Generally, the error log entries at the end of the display, if they have an influence on the problem being reviewed, have time stamps that relate to or immediately precede the actual abend; although there is no guarantee the error records will be written in the order they occurred. The error log entries are also written to an internal storage buffer that is included in the dump.

Using a logrec log stream rather than a logrec data set (SYS1.LOGREC, by default) for each system can streamline logrec error recording.

Console log
Console data that the installation chooses to log.

Hardcopy log
The hardcopy log is a record of the system message traffic that the installation chooses to log, such as messages to and from all consoles, commands and replies entered by the operator. In a dump, these messages are in the master trace. With JES3, the hardcopy log is always written to the SYSLOG. With JES2, the hardcopy log is typically written to the SYSLOG, but can also be written to a console printer, if your installation chooses.

Related information:

- *z/OS MVS Planning: Operations* contains information about OPERLOG and SYSLOG.
- Recording logrec error records in *z/OS MVS Diagnosis: Tools and Service Aids* contains complete information about EREP.
- Error recording on the logrec data set in *z/OS MVS Diagnosis: Reference* lists the incidents and the types of records that can be recorded on the logrec data set for each incident.

Traces

System trace
System trace provides an ongoing record of hardware events and software events occurring during system initialization and operation. The system activates system tracing at initialization, which runs continuously, unless your installation has changed the IBM-supplied system tracing. After system initialization, you can use the TRACE command on a console with master authority to customize system tracing. System trace is formatted in a dump using the IPCS SYSTRACE command.

For complete information, see System trace in *z/OS MVS Diagnosis: Tools and Service Aids*.

Master trace
Master trace maintains a table of all recently issued system messages. This creates a log of external system activity; the other traces log internal system activity. Master trace is activated automatically at system initialization, but you can turn it on or off using the TRACE command. Master Trace is formatted in a dump using the VERBX MTRACE command.

For complete information, see Master trace in *z/OS MVS Diagnosis: Tools and Service Aids*.

Component trace
The component trace service provides a way for z/OS components to collect problem data about events that occur in the component. Each component that uses the component trace service has set up its trace in a way that provides the unique data needed for the component. Component trace is queried and formatted using the IPCS CTRACE command Trace data is commonly used by the IBM Support Center to:

- Diagnose problems in the component
- Check how the component is running

The IBM support center might direct you to use specific component trace options when you need to re-create a problem to gather more diagnostic data.

For complete information, see Component trace in *z/OS MVS Diagnosis: Tools and Service Aids*.

Transaction trace

Transaction trace enables you to debug problems by tracing the path of a work unit running in a single system or across systems in a sysplex environment. Transaction trace provides a consolidated trace of key events for the execution path of application or transaction type work units running in a multi-system application environment.

The essential task of transaction trace is to aggregate data showing the flow of work between components in the sysplex that combine to service a transaction. Transaction trace traces events such as component entry, exit, exceptions and major events such as COMMIT, and ROLLBACK.

Restriction: Do not use transaction trace as a component tracing facility.

For complete information, see Transaction trace in *z/OS MVS Diagnosis: Tools and Service Aids*.

Generalized trace facility (GTF)

GTF traces system and hardware events similar to those in system trace, but also offers the option of an external writer and to write user defined trace events. GTF trace records can be formatted in a dump or trace data set using the IPCS GTFTRACE command.

For complete information, see The Generalized Trace Facility (GTF) in *z/OS MVS Diagnosis: Tools and Service Aids*.

GFS trace (GFS)

GFS trace is a tool that collects information about the use of the GETMAIN, FREEMAIN, or STORAGE macro. You can use GFS trace to analyze the allocation of virtual storage and identify users of large amounts of virtual storage. You must use the generalized trace facility (GTF) to get the GFS trace data output.

For complete information, see GETMAIN, FREEMAIN, STORAGE (GFS) trace in *z/OS MVS Diagnosis: Tools and Service Aids*.

Related information:

- *z/OS DFSMSdfp Diagnosis*
- *z/OS Infoprint Server Messages and Diagnosis*
- For a comprehensive overview of tools and service aids, see the topic on Selecting tools and service aids in *z/OS MVS Diagnosis: Tools and Service Aids*.
- *z/OS Communications Server: IP Configuration Reference*

Dumps

SVC dump

An SVC dump provides a representation of the virtual storage for the system when an error occurs. Typically, a system component requests the dump from a recovery routine when an unexpected error occurs. However, an authorized program or the operator can also request an SVC dump when diagnostic dump data is needed to solve a problem. Complete details are found in SVC dump in *z/OS MVS Diagnosis: Tools and Service Aids*.

Transaction dump

A transaction dump provides a representation of the virtual storage for an address space when an error occurs. Typically, an application requests the dump from a recovery routine when an unexpected error occurs. Complete details are found in Transaction dump in *z/OS MVS Diagnosis: Tools and Service Aids*

Abend dump

An ABEND dump shows the virtual storage predominately for an unauthorized program. To produce a dump when one is requested for an error, a JCL DD statement of SYSUDUMP, SYSABEND or SYSMDUMP must be included in the input job stream. See *z/OS MVS JCL Reference*for more information. An operator can also request an ABEND dump while ending a program, an address space, or canceling a job. There are three types of abend dumps:

- SYSMDUMP – Is an unformatted dump that requires IPCS to view and format. Unformatted dumping is sometimes more efficient because only the storage requested is written to the data set, which means the application can capture diagnostic data and be brought back online faster.
- SYSABEND – The largest of the ABEND dumps, is a pre-formatted dump containing a summary dump for the failing program plus many other areas useful for analyzing processing in the failing program.
- SYSUDUMP – The smallest of the ABEND dumps, containing data and areas only about the failing program.

Complete details are found in Abend dump in *z/OS MVS Diagnosis: Tools and Service Aids*.

SNAP dump

A SNAP dump shows virtual storage areas that a program, while running, requests the system to dump. A SNAP dump, therefore, is written while a program runs, rather than during abnormal end. The program can ask for a dump of as little as a one byte field to as much as all of the storage assigned to the current job step. The program can also ask for some system data in the dump. A SNAP dump is especially useful when testing a program. Complete details are found in SNAP dump in *z/OS MVS Diagnosis: Tools and Service Aids*

Stand-Alone dump

The other tools discussed in this chapter are used to collect data for individual work units on a system or a subset of components on a system. A stand-alone dump is used to collect diagnostic information about the entire system. Stand-alone dumps are not produced by z/OS but by an either the IPCS SADMP dump data set utility or the AMDSADDD REXX utility. After a stand-alone dump is taken, because the system cannot resume usual processing, the IPL is of the stand-alone dump instead of z/OS.

The stand-alone dump program produces a stand-alone dump of storage that is occupied by either:

- A system that is stopped. For example, your installation has a wait state with no processing, so you must capture a stand-alone dump to diagnosis it.

- A stand-alone dump program that failed. Either the stand-alone dump program dumped itself — a self-dump —, or the operator loaded another stand-alone dump program to dump the failed stand-alone dump program.

The stand-alone dump program and the stand-alone dump together form what is known as the stand-alone dump service aid. The term stand-alone means that the dump is performed separately from usual system operations and does not require the system to be in a condition for normal operation. It is essential to perform a store status before taking a stand-alone dump because the program gets loaded over storage that might be needed in the dump.

For more information:
- See the topics on Chapter 3, "Best practices for large stand-alone dump," on page 25.
- See the complete details in Stand-Alone dump in *z/OS MVS Diagnosis: Tools and Service Aids* and in *z/OS MVS IPCS User's Guide*.

IBM Omegamon for z/OS Management Console

The OMEGAMON z/OS Management Console is a monitoring product that includes an interface for z/OS management and is designed to help eliminate, and simplify many z/OS management tasks. The OMEGAMON z/OS Management Console helps deliver real-time, check information provided by the IBM Health Checker for z/OS, and configuration status information for z/OS systems and sysplex resources.

For more information, see IBM OMEGAMON for z/OS Management Console at www.ibm.com/systems/z/os/zos/zmc/.

Sending problem documentation to IBM

There are two tools available to send problem documentation to IBM. z/OS Problem Documentation Upload Utility (PDUU) and AMATERSE each have unique characteristics suitable for the type of documentation you must send to IBM:

Problem Documentation Upload Utility
The z/OS Problem Documentation Upload Utility (PDUU) is the primary utility for sending large volumes of documentation, such as stand-alone dumps, to the IBM FTP site. The encryption capability ensures that the transfer occurs in a secure manner. For complete details, see the topic about Problem Documentation Upload Utility in *z/OS MVS Diagnosis: Tools and Service Aids*.

AMATERSE
AMATERSE is useful for compressing (packing) and unpacking relatively small amounts of service data, but is incompatible with PDUU (output and input), and offers no data transfer or encryption capability. Use AMATERSE to compress and extract problem documentation you send to IBM. There are differences between AMATERSE and the former TRSMAIN utility. AMATERSE is the preferred over TRSMAIN because it is a supported program. For complete details, see the topic about AMATERSE in *z/OS MVS Diagnosis: Tools and Service Aids*.

IBM documentation

There are many types of documentation to aid problem determination. Here are some of the categories:

- Chapter 22, "Diagnosis information for z/OS base elements and features," on page 285, which contains diagnosis material by element or feature name.
- z/OS Internet Library, which contains complete, updated information about all z/OS elements and features: www.ibm.com/servers/eserver/zseries/zos/bkserv/

  Many people find it helpful to search on an individual element or feature using the z/OS elements and features search engine: www.ibm.com/servers/eserver/zseries/zos/bkserv/zshelves9.html
- **z/OS Hot Topics Newsletter**, written by leading z/OS experts, contains hands-on, technical information about z/OS that is not contained in the traditional product libraries: www.ibm.com/servers/eserver/zseries/zos/bkserv/hot_topics.html
- The Techdocs Library, which includes:
  - Flashes that alert you to significant new technical developments and provide guidance on the installation, use and, management of z/OS: www.ibm.com/support/techdocs/atsmastr.nsf/Web/Flashes
  - FAQs to assist you with the installation, use, and management of z/OS: www.ibm.com/support/techdocs/atsmastr.nsf/Web/FAQs
  - White papers, presentations, and more at www.ibm.com/support/techdocs/atsmastr.nsf/Web/Techdocs.
  - Technotes that includes best practices, performance evaluations, recent enhancements and helpful hints and tips: www.ibm.com/support/techdocs/atsmastr.nsf/Web/Technotes
- IBM Redbooks® provide positioning and value guidance, installation and implementation experiences, typical solution scenarios, and step-by-step "how-to" guidelines: www.redbooks.ibm.com/.

Chapter 3. Best practices for large stand-alone dump

This information describes a set of best practices for optimizing stand-alone dump (SADMP) data capture, optimizing problem analysis time, and ensuring that the stand-alone dump is successful at capturing the necessary information for use by IBM Support. In particular, the following areas:

- "Using AutoIPL for stand-alone dumps"
- "Planning a multivolume stand-alone dump data set" on page 26
- "Creating the multivolume SADUMP" on page 27
- "Defining a dump directory for large stand-alone and SVC dumps" on page 27
- "Preparing the dump for further processing with IPCS COPYDUMP" on page 27
- "Compressing data for faster transmission and analysis" on page 28
- "Transmitting dump data to IBM" on page 29
- "Setting up remote access" on page 29
- "Testing your stand-alone dump operations" on page 29
- "Automating the SADMP process" on page 30, which includes "Sample JCL for post-processing" on page 30
- "IBM System Test example" on page 31

This information replaces existing stand-alone dump best practices information previously documented in:

- "z/OS Best Practices: Large stand-alone dump handling," found by searching for TD103286 at Techdocs www.ibm.com/support/techdocs/atsmastr.nsf/Web/Technotes.
- "Every picture tells a story: Best practices for stand-alone dump, " published in the February 2007 issue of the *z/OS Hot Topics Newsletter*. You can access current issues of z/OS Hot Topics Newsletter at www.ibm.com/servers/eserver/zseries/zos/bkserv/hot_topics.html.

Using AutoIPL for stand-alone dumps

You can enable z/OS to automatically trigger a stand alone dump using the automatic IPL (AutoIPL) function. AutoIPL is an automated function, defined in the DIAGxx parmlib member, that the system checks at wait state time. AutoIPL can re-IPL z/OS, or take a SADMP, or take a SADMP and have SADMP re-IPL z/OS when it finishes.

For details including the hard-coded table of wait state and reason codes, the wait state action table (WSAT), which triggers AutoIPL, see:

- The topic about Using the automatic IPL function in *z/OS MVS Planning: Operations*.
- The setting for AutoIPL in the topic about the DIAGxx parmlib member in *z/OS MVS Initialization and Tuning Reference*.

Planning a multivolume stand-alone dump data set

Plan a multivolume stand-alone dump data set that places each volume on a separate DASD volume on a separate control unit. You can achieve the best dump performance when the dump is taken to a multivolume DASD stand-alone dump data set. Stand-alone dump exploits multiple, independent volume paths to accelerate data recording. The dump data set is actually spread across all of the specified volumes, not each volume in succession. They should not be treated as multiple single data sets. See the topic on “Creating the multivolume SADUMP” on page 27.

One of the key performance elements of stand-alone dump is the rate at which data writes to DASD. Modern DASD uses cache in the control unit to improve the performance of write operations. The placement of the multivolume stand-alone dump data set across logical subsystems (LSS) needs to avoid filling the bus or cache within the DASD with the data to be written. When the bus or cache is full of data to be written, the speed of the DASD is reduced to the (slower) speed at which the data can be written to the physical media.

There are significant performance improvements when writing the data to a multivolume stand-alone dump data set, or to specific types of DASD. For more information, review the IBM performance analysis reports flash10143: www.ibm.com/support/docview.wss?uid=tss1flash10143.

When defining your placement of a multivolume stand-alone dump data set, use the following guidelines:

1. Configure each volume on a separate logical subsystem (LSS) to ensure maximum parallel operation. You can achieve the best performance of stand-alone dump when the multivolume data sets have the most separation. That is, separate physical control units and separate channel paths.
2. Configure, if possible, the control units to minimize the occurrence of other activity at the time of the stand-alone dump. For example, DB2® database recovery writing to a local database volume on the same control unit as the stand-alone dump volume can result in slower dump speed and might affect the elapsed time needed to restart an alternative DB2 on an LPAR that is still running.
3. Use FICON-attached DASD volumes, when possible, to yield the best data rates. FICON® channels can deliver much better performance than ESCON® channels. However, with sufficient infrastructure and I/O tuning, an ESCON configuration can still deliver high performance.
4. Dedicate more DASD volumes to SADUMP, up to the maximum of 32 volumes, for better overall performance. IPCS offers a SADMP Dump Data Set Utility, available from the IPCS Utility menu. From the data set utility panel, you can specify whether to define, clear, or reallocate your stand-alone dump data set, specify its name and the volume serial numbers for the SADMP “stripes”. This panel will then start the SADMP allocation program to define the data set that you requested. The volume names, device type, and allocated space are also confirmed. While it is not recommended by IBM, stand-alone dump can also be written to a fast tape subsystem. When directing the SADUMP to a tape drive, the dump only uses a single device and does not prompt for another device, so you cannot switch back to using a DASD device for that stand-alone dump.

Creating the multivolume SADUMP

Use the AMDSADDD utility to define a stand-alone dump data set. Specify a volume list (VOLLIST) in AMDSADDD to designate a list of VOLSERs corresponding to each DASD volume making up the data set. You can allocate a multivolume data set using the specified list of volumes. The device number of the first volume is used to specify the data set to stand-alone dump. Again, each volume should be on a different LSS to ensure parallelism when writing the dump. For a sample job that uses AMDSADDD to generate the SADMP data set, see “IBM System Test example” on page 31. Be sure to catalog your SADMP data set to prevent the possibility of accessing the wrong version of the data set when using *IPCS COPYDUMP* later.

For additional details, see the topic on Using the AMDSADDD utility in *z/OS MVS Diagnosis: Tools and Service Aids*.

Defining a dump directory for large stand-alone and SVC dumps

Choosing the right attributes is the key to facilitating post-processing of large stand-alone dumps and large SVC dumps. IPCS is used to consolidate and extract ASIDs from the dump, format, and analyze the dump. IPCS uses a dump directory to maintain information about the layout and content of the dump. The dump directory is a VSAM data set that you can tune for optimal performance.

You can improve IPCS performance by reducing the number of control interval (CI) splits during initialization and analysis of dumps. To do this, specify the RECORDSIZE parameter in BLSCDDIR (shipped in SYS1.SBLSCLI0). The RECORDSIZE parameter in BLSCDDIR is ‘RECORDSIZE (2560 3072)’ and yields well performing CISIZEs for the data portion of the data set. To allow IPCS to be more efficient in its processing, it is recommended that you delete old dump references from the directory periodically (especially stand-alone dumps).

Note:
1. When IBM System Test uses BLSCDDIR, they specify a CI size of 24,576 and a BUFSPACE of X'100000'.
2. You can tune the RECORDSIZE parameter by observing the number of CI splits using standard VSAM data set analysis techniques, such as the LISTCAT command.

Preparing the dump for further processing with IPCS COPYDUMP

Before you begin: If you are using IPCS on z/OS V1R7, apply PTF UA26080 to fix a problem that causes long IPCS initialization time.

After a stand-alone dump is taken to a multivolume data set, it needs to be post-processed before IPCS or other tools can view it. The IPCS COPYDUMP utility reads and processes the multivolume stand-alone dump faster than IEBGENER, and produces a merged dump ordered by ASID. The COPYDUMP utility processes the dump volumes in parallel, allowing the original dump to be read faster. Using COPYDUMP also helps IBM process the dump more quickly because it eliminates the need to reprocess the dump at IBM. Here is how it works:
1. Use IPCS COPYDUMP to produce a merged dump data set from the multivolume stand-alone dump, and a subset of the original stand-alone dump (ASIDs 1-20). “Sample JCL for post-processing” on page 30 contains a sample batch job.

2. Ensure that the output data set specified to COPYDUMP is DFSMS-striped with at least eight stripes.
3. Catalog the output dump data set to allow IPCS to access it properly.
4. Send the subset dump to IBM using a program like “Sending problem documentation to IBM” on page 22. This is a smaller data set, which takes less time to send through the FTP program to IBM.
5. Keep the merged dump for later use by IBM Support, if necessary.

You can run COPYDUMP to produce a merged version of the entire multivolume stand-alone dump, or to extract a subset of the address spaces contained in the original dump and written to the merged output data set. IBM recommends that you use two COPYDUMP jobs in parallel to produce a full merged dump and a subset merged dump. The subset dump will contain ASIDs 1-20 with the primary system components of the operating system.

IPCS performance is improved when the dump being processed (the COPYDUMP output) is DFSMS-striped. Placing the dump into a data set with at least eight stripes has shown marked improvement in IPCS response (when IBM is analyzing the problem).

A subset of ASIDs can be extracted from the full stand-alone dump into a separate data set and sent to IBM using COPYDUMP. This has been shown to reduce the data transferred by roughly 30% to 40% compared to transmitting a full stand-alone dump. Use the EASYCOPY parameter on COPYDUMP to automatically create a JOBLIST entry with a predefined list of system address space names. The JOBLIST includes the following job names: ALLOCAS, ANTAS000, ANTMAIN, CATALOG, CONSOLE, DEVMAN, DUMPSRV, IEFSCHAS, IOSAS, IXGLOGR, JESXCF, JES2, JES3, and OMVS.

The syntax of the IPCS COPYDUMP command is:

```
COPYDUMP ODS('OUTPUT DATASET NAME')
EASYCOPY
IDS('INPUT DATASET NAME') NOCONFIRM
```

Again, ensure that the specified output data set name supports DFSMS striping.

Compressing data for faster transmission and analysis

Compress dumps before sending the data to IBM using FTP. Beginning in z/OS V1R13, you can use the z/OS Problem Documentation Upload Utility (PDUU) to compress and send the dump to IBM. PDUU also offers encryption. In z/OS V1R9 through z/OS V1R12 or when sending documentation to independent software vendors, use AMATERSE. Otherwise, use TRSMAIN. For AMATERSE and TRSMAIN, you might need to encrypt the resulting data set, so that the data is secure when it arrives at one of the IBM Support staging points (TESTCASE or ECUREP).

- For PDUU, see the topic about z/OS Problem Documentation Upload Utility in *z/OS MVS Diagnosis: Tools and Service Aids*.
- For AMATERSE, see the topic about AMATERSE in *z/OS MVS Diagnosis: Tools and Service Aids*.
- For early releases, download TRSMAIN from the IBM support Web site:
  - https://service.software.ibm.com/s390/support.

- If you require that the data be encrypted prior to sending it to IBM, place the necessary decryption information in the PMR for IBM Support to use. For more information on using IBM Encryption Facility for z/OS, see:
  - www.ibm.com/de/support/ecurep/mvs_encryption.html.

Transmitting dump data to IBM

The z/OS Problem Documentation Upload Utility (PDUU) is the primary utility for sending large volumes of documentation, such as stand-alone dumps, to the IBM FTP site. The encryption capability ensures that the transfer occurs in a secure manner. For complete details, see the topic about Problem Documentation Upload Utility in *z/OS MVS Diagnosis: Tools and Service Aids*.

Setting up remote access

Set up remote access through Remote Screen Viewing Support Facility (RSVSF) or Assist On-Site (AOS) to allow IBM to remotely view your dump in time-critical situations. Remote access products allow you to permit IBM Support personnel to immediately log into an IPCS session and view available documentation with no initial data transfer. Choices include:

- Assist On Site (AOS) available for use worldwide at www.ibm.com/support/assistonsite.
- Remote Screen Viewing Support Facility (RSVSF), contact your service representative for details.
- OnTop available for use in Europe, Northeast Europe, and Southwest Europe, contact your service representative for details.

This should always be the first option in a time-critical situation. Rapid viewing of the documentation has the advantage of allowing IBM Support to itemize or customize any additional documentation they may want to send to IBM for the given situation. If documentation is required to be sent through FTP, the analysis of the current documentation can continue while the requested documentation is in transit. In many cases, sending a subset of the stand-alone dump to IBM can prove sufficient for problem resolution as the complete stand-alone dump is not always required for diagnosis of a given problem.

Testing your stand-alone dump operations

It is critical for your Operations staff to train and practice taking a stand-alone dump so that they are familiar with the procedure, and to ensure that all data sets are set up properly before you run into a critical situation. This includes the process and set up for:

- Taking a SADUMP as part of the standard scheduled shutdown of an LPAR
- Using COPYDUMP to obtain the merged dump and the subset dump.

If the dump resides in a DASD dump data set, IBM recommends that you copy the dump to another data set for IPCS processing and clear (re-initialize) the dump data set using the AMDSADDD or IPCS SADMP dump data set utilities. For more information, see the topic on “Using the AMDSADDD utility” in Using the IPCS Dialog Using the IPCS Dialog in *z/OS MVS IPCS User's Guide*.

The best practice is to rehearse taking a stand-alone dump during scheduled disaster recovery drills. You can also consider practicing when migrating to a new z/OS release, or when moving to a new processor. If you have a test LPAR that you use to train your operations staff, one part of that training might be to take a stand-alone dump following your local procedures and prepare how to react to stand-alone dump messages.

Automating the SADMP process

The following sample JCL can help you automate several best practices. The results are is two "steps" that can be run as background jobs:

1. Use IPCS COPYDUMP to merge the data and produce a single data set to send to IBM. Because the JCL requires invoking IPCS in a background TSO environment, it is not possible to obtain condition code information from the COPYDUMP "step" to determine whether to invoke the preparation step. That means you must manually examine the results of the COPYDUMP step.
2. Use the "Preparation" job, which will compress the output data set produced by COPYDUMP, encrypt the compressed version, and send the final result through FTP to IBM using PUTDOC.

Tip: Beginning with z/OS V1R10, you can use the AutoIPL function to ensure z/OS takes a stand-alone dump when about to load a disabled wait state. For additional details, see the topic about Using the automatic IPL function in *z/OS MVS Planning: Operations*.

Sample JCL for post-processing

Post-processing of a stand-alone dump needs to occur in two steps:

1. Run IPCS COPYDUMP to merge the data and produce a single data set to send to IBM. Examine the output from the run step to ensure that the COPYDUMP ran correctly. This JCL is identified as `=== IPCS COPYDUMP ====`.
2. Run the following JCL, which will terse the resulting (striped) dump data set, encrypt the tersed version, and send it through FTP to IBM using PUTDOC. This JCL is identified as `=== TERSE, ENCRYPT and FTP ====`.

You can tailor the following JCL to process the data sets to be transmitted to the FTP server. Turn off the LINE NUMBERING in following job.

```
=== IPCS COPYDUMP ====
//IPCSCPYD JOB MSGLEVEL=(2,1),....
// CLASS=V,NOTIFY=&SYSUID.,MSGCLASS=H
//*******************************************************************
//* IN    DD IS USED TO POINT TO THE SOURCE OF INPUT WHICH WOULD BE
//*          THE SYS1.SADMP... DATASET
//* OUT   DD IS USED TO POINT TO THE OUTPUT OF THE COPYDUMP
//*       WHERE PPPPP SHOULD BE THE NUMBER OF CYLINDERS FOR PRIMARY
//*             SSSS SHOULD BE THE NUMBER OF CYLINDERS FOR SECONDARY
//*             &DATACLAS SHOULD BE THE DATACLAS
//*             &MGMTCLAS SHOULD BE THE MGMTCLAS
//*             &STORCLAS SHOULD BE THE STORCLAS
//* IPCSDDIR DD DEFINING &SYUID..COPYDUMP.DDIR WITH NON-COMPRESS
//*       DATACLAS
//* COPYDUMP SUBCOMMAND TO REQUEST FIRST 20 ADDRESS SPACES
//*       IF JES OR CATALOG WERE NOT AMONG THE FIRST 20 ADDRESS SPACES
//*       XXX AND YYY SHOULD BE USED FOR THESE TWO SUBSYSTEM ASIDS
//*******************************************************************
//RUN      EXEC PGM=IKJEFT01,REGION=200096K,DYNAMNBR=50
//IPCSPRNT DD SYSOUT=H
//IPCSTOC  DD SYSOUT=H
//IPCSPARM DD DISP=SHR,DSN=SYS1.PARMLIB
//SYSTSPRT DD SYSOUT=H
//IN       DD DISP=SHR,DSN=SYS1.SADMP.....
//OUT      DD DISP=(NEW,CATLOG),DSN=OUTPUT.DATASET.NAME
```

```
//         SPACE=(CYL,(PPPPP,SSSS),RLSE),DATACLAS=&DATACLAS,
//         MGMTCLAS=&MGMTCLAS,STORCLAS=&STORCLAS
//SYSTSIN DD  *
EX 'SYS1.SBLSCLI0(BLSCDDIR)' 'DSN(&SYSUID..COPYDUMP.DDIR) +
   RECORDS(90000) DATACLAS(NOCOMP) MGMTCLAS(DMGDEBUG)'
IPCS NOPARM
COPYDUMP IFILE(IN) OFILE(OUT) ASIDLIST(1:20,XXX,YYY) NOCONFIRM
END
/*
```

```
==== TERSE, ENCRYPT and FTP ====
//TRENCFTP JOB CLASS=I,......
// NOTIFY=&SYSUID.
//JOBLIB  DD   DISP=SHR,DSN=PDS_WITH_TERSE_ENCRYP_PGM
//TERSE    EXEC PGM=TRSMAIN,PARM=PACK
//SYSPRINT DD   SYSOUT=H
//INFILE   DD   DISP=SHR,DSN=SOURCE_OF_DUMP
//OUTFILE  DD   DISP=(NEW,CATLG),
//       DSN=&SYSUID..PMR....TRSD,
//       UNIT=SYSDAL,
//       DATACLAS=COMPRESS,
//       SPACE=(CYL,(PPPPP,SSSS),RLSE)
//DECRYPT  EXEC PGM=FTPENCRD,PARM='PASSCODE',COND=(0,NE)
//SYSOUT   DD   SYSOUT=*
//SYSPRINT DD   SYSOUT=*
//FIN      DD DISP=SHR,DSN=*.TERSE.OUTFILE
//FOUT     DD   DSN=&SYSUID..PMR.....TRSENCRP,
//         DCB=(DSORG=PS,RECFM=FB,LRECL=1024),
//         DISP=(NEW,CATLG),UNIT=SYSDAL,
//         DATACLAS=COMPRESS,
//         SPACE=(CYL,(PPPPP,SSSS),RLSE)
//FTPSTEP  EXEC PGM=FTP,REGION=5000K,
//          PARM='TESTCASE.BOULDER.IBM.COM (EXIT',COND=(00,NE)
//STEPLIB  DD  DISP=SHR,DSN=SYS1.TCPIP.SEZALINK
//*SYSMDUMP DD  SYSOUT=*
//SYSPRINT DD  SYSOUT=H
//OUTPUT   DD  SYSOUT=H
//INPUT    DD  *
ANONYMOUS
YOUR_EMAIL@
cd mvs/toibm
bin
PUT PMR......TRSENCRP PMR.......TRS.ENCRP64
quit
/*
```

IBM System Test example

In a recent set of tests performed by the IBM System z® Product Evaluation Test team, a 12-volume configuration was set up to support a stand-alone dump of a 152 GB real memory system:

- Three Enterprise Storage Server® (ESS) subsystems were used:
  - ESS 2105 F20 - 2 FICON 2 GB CHP, 8 GB cache
  - ESS 2105 mod. 800 - 8 FICON 2 GB CHP, 8 GB cache
  - DS6000™ 1750 mod. 511 - 6 FICON 2GB CHP, 1.3 GB cache.
- Four volumes per CP were defined:
  - Each volume on a unique LSS
  - Each volume as 14902 cylinders
  - The DSTYPE=LARGE attribute was used.

Here is an example of the AMDSADDD JCL that used the DASD configuration:

```
//STEP1    EXEC PGM=IKJEFT01
//SYSTSPRT  DD  SYSOUT=*
//SYSTSIN   DD  *
EXEC 'SYS1.SBLSCLI0(AMDSADDD)' 'DEFINE (SAD041,SAD042,SAD043,SAD044,SA+
D045,SAD046,SAD047,SAD048,SAD049,SAD050,SAD051,SAD052)(PETDUMP.J80L12.+
SADUMP.DSS0.STRIPE) 3390 14902 YES LARGE'
```

Part 2. Runtime Diagnostics

Runtime Diagnostics (component name HZR) can perform many of the same tasks you might typically perform when looking for a failure but done much more quickly and without the need for a storage dump.

Chapter 4. Runtime Diagnostics

Runtime Diagnostics is a base component (HZR) of z/OS that is designed to help you analyze a system that has a potential problem or soft failure. Soft failures are often difficult or impossible to detect and can slowly lead to the degradation of the solution that is using z/OS. To understand soft failures, see the definition in Chapter 7, "Predictive Failure Analysis overview and installation," on page 65.

Runtime Diagnostics does many of the same tasks you might typically do when looking for a failure, such as:
- Reviewing critical messages in the log
- Examining address spaces with high processor usage
- Looking for an address space that might be in a loop
- Evaluating local lock conditions
- Analyzing various types of contention that include ENQ, GRS latch contention, and z/OS UNIX file system latch contention.

In many cases, when Runtime Diagnostics finds a critical message, it does additional analysis based on the job name or other information in the message text. For example, if Runtime Diagnostics identifies an XCF stalled connector message, it performs additional analysis of the identified address space to help narrow down the problem. A key feature of Runtime Diagnostics is its ability to summarize internal processing errors and return the results to you in a message response.

Predictive Failure Analysis (PFA) can also return Runtime Diagnostics report information when activity is absent or unusually low for:
- "PFA_MESSAGE_ARRIVAL_RATE" on page 120
- "PFA_ENQUEUE_REQUEST_RATE" on page 97
- "PFA_SMF_ARRIVAL_RATE" on page 136.

For details, see "How PFA invokes Runtime Diagnostics" on page 67.

This section covers the following topics:
- "How Runtime Diagnostics works"
- "Enabling Runtime Diagnostics" on page 36 and "Reports from Runtime Diagnostics" on page 38
- "Runtime Diagnostics symptoms" on page 39
- "Runtime Diagnostics messages" on page 45
- "Runtime Diagnostics DEBUG options" on page 46

How Runtime Diagnostics works

Note: Runtime Diagnostics is a diagnostic tool to run in your environment when your system experiences symptoms that require its use. Run it when you experience system degradation or if you want to check for potential problems; do not run when the system is operating normally. IBM Service might also request that you run Runtime Diagnostics and report its results.

After you start Runtime Diagnostics (S HZR,SUB=MSTR), you can analyze the home system by entering the following MODIFY (or F) command.

```
MODIFY HZR,ANALYZE
```

or

```
F HZR,ANALYZE
```

Runtime Diagnostics searches for certain messages and message combinations in the operations log (OPERLOG) stream and attempts to identify other system symptoms with minimal dependencies on other system services. By default, Runtime Diagnostics analyzes the home system.

If you want Runtime Diagnostics to analyze a different system, specify the system name on the `MODIFY` command by entering the `SYSNAME` parameter:

```
F HZR,ANALYZE,SYSNAME=SYSB
```

For example, if you are using Runtime Diagnostics on the home system, such as SYSA, all analysis is for the home system. If you must analyze a system other than SYSA, such as SYSB, specify `F HZR,ANALYZE,SYSNAME=SYSB`. When you use this method, Runtime Diagnostics analysis is limited to critical messages in OPERLOG and ENQ information.

Typically when analyzing a system, Runtime Diagnostics runs for less than one minute. It can take more time; it depends on how many systems are in your environment. For example, if Runtime Diagnostics analyzes four systems in a Parallel Sysplex® environment, it must analyze OPERLOG messages from each system in the sysplex to determine which messages are issued by the target system. This analysis takes more time than analyzing a single system where all messages are issued by the target system.

When Runtime Diagnostics finds a problem, it displays a multi-line write-to-operator (WTO) message that lists system error events. The message contains a problem description and a suggested next action for your analysis.

Enabling Runtime Diagnostics

Before z/OS V1R13, Runtime Diagnostics ran as a started task under the master subsystem and had to be started each time you wanted an analysis. Runtime Diagnostics started, ran its analysis, and then ended.

Beginning with z/OS V1R13, Runtime Diagnostics (HZR) ships in the SYS1.PROCLIB data set. You must start Runtime Diagnostics to run as an address space under the master subsystem (`S HZR,SUB=MSTR`). After you start the Runtime Diagnostics address space (HZR), it remains active until you decide to stop it using the `STOP` command.

Requirements:

- You must use the following instructions to run Runtime Diagnostics. For more details, see "How Runtime Diagnostics works" on page 35.
- Using OPERLOG and setting read permissions for the SYSPLEX.OPERLOG is a required to use all the Runtime Diagnostics functions including critical message analysis. Runtime Diagnostics can perform limited analysis without OPERLOG. For details about setting up OPERLOG for your installation, see Chapter 6, "Using OPERLOG," on page 59.

Use the following instructions to start Runtime Diagnostics:

1. Define a user ID for the HZR started task by following the steps your installation has in place for its security policy. For example, you might use:

   ```
   ADDUSER HZR OWNER(*owner*)DFLTGRP(*dfltgrp*) NOPASSWORD
   ```

2. Define a profile for HZR in the RACF® STARTED class with the user ID from 1. For example, you might specify:

   ```
    RDEFINE STARTED HZR.HZR STDATA(USER(HZR) GROUP(*dfltgrp*) TRUSTED(YES))
    SETROPTS CLASSACT(STARTED) RACLIST(STARTED)
   ```

   Note: On the RDEFINE command, you must specify HZR as the job name. During START processing, HZR initialization checks the job name and fails the START command if HZR is not present.

3. Ensure the hzrproc (HZR) points to the program name PGM=HZRINIT; not PGM=HZRIMAIN as it did if you previously ran Runtime Diagnostics in V1R12. (The hzrproc (HZR) ships in the SYS1.PROCLIB data set.)

4. Ensure Runtime Diagnostics restarts on IPL by updating the COMMNDxx parmlib member with the HZR procedure. Do the same any system automation your installation uses to start and restart major system address spaces.

   ```
   COM='S HZR,SUB=MSTR'
   ```

   For more information about the COMMNDxx member of parmlib, see *z/OS MVS Initialization and Tuning Reference*.

5. After you start the HZR address space, enter the MODIFY command to use Runtime Diagnostics. For example:

   ```
   MODIFY HZR,ANALYZE
   ```

   The system returns either message HZR0200I or HZR0201I. The output shows SUMMARY: SUCCESS as in Figure 2 on page 38.

Note: The system rejects the START command if you try to start Runtime Diagnostic with any name other than HZR. For example, if you change the name from HZR to HZR780, you must specify the JOBNAME parameter on the start command as follows:

```
S HZR780,SUB=MSTR,JOBNAME=HZR
```

The JOBNAME parameter assigns the name HZR to the started address space as opposed to HZR780.

To view the HZROUT data set without stopping HZR, you must specify DISP=SHR. For example:

```
LRECL=121,BLKSIZE=0,RECFM=FB,DISP=SHR
```

HZR ships in the SYS1.PROCLIB data set with HZROUT specified as DD DUMMY. You can modify HZR to direct HZROUT to a sequential data set. If you are using only the WTO message output (as written to the hardcopy log), a copy of the Runtime Diagnostics message goes to DDNAME HZROUT. Specify BLKSIZE=0 for the system choose an optimum block size for DASD. If you set the block size to a number other than zero, ensure that it is an even multiple of 121.

Alternatively, if you are using only the WTO message output, you can omit the HZROUT DD from the PROC. If you decide to omit HZROUT, a message displays a warning that DDNAME HZROUT is missing and therefore no output is written to HZROUT.

Running Runtime Diagnostics with mixed releases of z/OS

To enable Runtime Diagnostics to run on z/OS V1R12, set up a separate PROC statement for each image that is running the lower release. For example, you have z/OS V1R13 installed with Runtime Diagnostics, but you also have a V1R12 system with Runtime Diagnostics. When you want to analyze the V1R12 system, enter the following statement on the V1R12 system:

```
S hzrproc,SUB=MSTR,JOBNAME=HZR
```

Note: On z/OS V1R12, the hzrproc points to program name PGM= HZRIMAIN. On the z/OS V1R13 system, the hzrproc points to program name PGM=HZRINIT.

Reports from Runtime Diagnostics

Runtime Diagnostics reports system symptoms it finds in the `EVENTS:` portion of the message report. "Runtime Diagnostics symptoms" on page 39 contains more examples and explanations of symptoms.

```
f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 974
SUMMARY: SUCCESS
 REQ: 001 TARGET SYSTEM: SY1      HOME: SY1      2010/12/21 - 11:30:57
 INTERVAL:  60 MINUTES
  EVENTS:
  FOUND: 05 - PRIORITIES: HIGH:05  MED:00  LOW:00
  TYPES: CF:04
  TYPES: HIGHCPU:01
----------------------------------------------------------------------
```

Figure 2. Status message that reports Runtime Diagnostics success

For Runtime Diagnostics to perform message analysis, you must ensure that the system is running OPERLOG and that read permissions are set for the SYSPLEX.OPERLOG. (Review "Enabling Runtime Diagnostics" on page 36 in "Steps for setting up OPERLOG" on page 59). If you do not connect to OPERLOG, a system logger message and possibly a RACF (or equivalent security product) message displays. See the example shown in Figure 3 on page 39 as message **ICH408I** and **IXG231I**.

Runtime Diagnostics reports when some of its processing fails (unable to complete processing for one or more events) as **QUALIFIED SUCCESS** in the **SUMMARY** portion of the report. Notice, in Figure 3 on page 39, how message HZR0200I explains what part of the processing was unsuccessful under the field **PROCESSING FAILURES.** In this case, Runtime Diagnostics did not have the proper Security Server RACF authority to connect to OPERLOG to examine messages. Yet Runtime Diagnostics continues its analysis for other soft failures. In this example, Runtime Diagnostics found a **HIGHCPU** error even though it was unable to connect to OPERLOG.

```
F HZR,ANALYZE
ICH408I JOB(HZR) STEP(HZR) SYSPLEX.OPERLOG CL(LOGSTRM ) 312
  INSUFFICIENT ACCESS AUTHORITY
  ACCESS INTENT(READ)  ACCESS ALLOWED(NONE)
IXG231I IXGCONN REQUEST=CONNECT TO LOG STREAM SYSPLEX.OPERLOG DID NOT
SUCCEED FOR JOB HZR.  RETURN CODE: 00000008  REASON CODE: 0000080D
DIAG1: 00000008  DIAG2: 00000000  DIAG3: 03010000  DIAG4: 00000000
HZR0200I RUNTIME DIAGNOSTICS RESULT 318
SUMMARY: QUALIFIED SUCCESS - SOME PROCESSING FAILED
 REQ: 001 TARGET SYSTEM: SY1      HOME: SY1      2011/04/22 - 10:06:10
 INTERVAL:  60 MINUTES
 EVENTS:
  FOUND: 01 - PRIORITIES: HIGH:01  MED:00  LOW:00
  TYPES: HIGHCPU:01
  PROCESSING FAILURES:
   OPERLOG....IXGCONN REQ=CONNECT ERROR.......RC=00000008 RS=0000080D
----------------------------------------------------------------------
EVENT 01: HIGH - HIGHCPU      - SYSTEM: SY1      2011/04/22 - 10:06:11
ASID CPU RATE:98%     ASID:002E   JOBNAME:IBMUSERX
STEPNAME:STEP1    PROCSTEP:         JOBID:JOB00051 USERID:IBMUSER
JOBSTART:2011/04/22 - 09:48:49
  ERROR: ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MIGHT BE LOOPING.
 ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
```

Figure 3. Status message that reports qualified success with events

In Figure 4, Runtime Diagnostics was unable to connect to OPERLOG to examine messages, yet it continues its analysis for other soft failures. The status message shows Runtime Diagnostics`Found: 00` additional errors (none).

```
HZR0200I RUNTIME DIAGNOSTICS RESULT  FRAME LAST   F      E   SYS=N68
SUMMARY: QUALIFIED SUCCESS - SOME PROCESSING FAILED
 REQ: 002 TARGET SYSTEM: N68      HOME: N68      2011/04/20 - 10:15:30
 INTERVAL:  60 MINUTES
 EVENTS:
  FOUND: 00  - PRIORITIES: HIGH:00  MED:00  LOW:00
  PROCESSING FAILURES:
   ENQ........ISGECA API ERROR................RC=00000004 RS=00000000
  PROCESSING BYPASSED:
   OPERLOG....OPERLOG IS NOT ACTIVE.
----------------------------------------------------------------------
```

Figure 4. Status message that reports qualified success with no events

Runtime Diagnostics symptoms

This section covers the types of problem symptoms Runtime Diagnostics can detect.

Critical message analysis
Runtime Diagnostics reads through the last hour of OPERLOG looking for critical messages. If any are found, Runtime Diagnostics lists the critical message as an error event. For a subset of critical messages, Runtime Diagnostics does additional analysis based on the message identifier and the content of the message. If less than one hour of message content is available in OPERLOG, message analysis is done for the messages available. For more details, see Figure 10 on page 44.

ENQ contention checking
Runtime Diagnostics provides a point in time check of ENQ contention equivalent to issuing the `D GRS,AN,WAITER` command. It compares the list of job names that are waiters with a hardcoded list of IBM-supplied address

spaces to determine whether any are waiters. If Runtime Diagnostics finds ENQ contention, it issues an error event within message HZR0200I stating the job name of the waiter for ENQ resource.

Here is the list of IBM-supplied address spaces:

Table 4. ENQ checking: IBM-supplied address spaces

IBM-supplied address spaces				
• *MASTER*	• ALLOCAS	• ANTMAIN	• ANTAS000	• BPXOINT
• CEA	• CONSOLE	• DEVMAIN	• DFHSM	• DUMPSRV
• GRS	• IOSAS	• IEFSCHAF	• IXGLOGR	• JES2
• JES2XCF	• JES2AUX	• JES2MON	• LLA	• OMVS
• PCAUTH	• RACF	• RASP	• SMS	• SMSPDSE
• SMSPDSE1	• SMSVSAM	• TRACE	• VLF	• WLM
• XCFAS	• ZFS			

```
SY1  F HZR,ANALYZE
   SY1  HZR0200I RUNTIME DIAGNOSTICS RESULT 989
   SUMMARY: SUCCESS
    REQ: 004 TARGET SYSTEM: SY1      HOME: SY1      2012/07/23 - 14:29:44
    INTERVAL:  60 MINUTES
    EVENTS:
     FOUND: 01 - PRIORITIES: HIGH:01  MED:00  LOW:00
     TYPES: ENQ:01
   ----------------------------------------------------------------------
   EVENT 01: HIGH - ENQ          - SYSTEM: SY1      2012/07/23 - 14:29:44
   ENQ WAITER  - ASID:0009 - JOBNAME:CONSOLE  - SYSTEM:SY1
   ENQ BLOCKER - ASID:000E - JOBNAME:MAINJOB  - SYSTEM:SY1
   QNAME: SYSZMCS
   RNAME: SYSMCS#MCS
     ERROR: ADDRESS SPACES MIGHT BE IN ENQ CONTENTION.
    ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE BLOCKING JOBS AND
    ACTION: ASIDS.
   ----------------------------------------------------------------------
```

Figure 5. Runtime Diagnostics ENQ contention report

To resolve contention, determine if the blocking job is running properly or if you must cancel it. The topic on Serialization summary in *z/OS MVS Diagnosis: Reference* contains the ENQ names with the issuing component.

Find additional contention information in the following topics:

- To identify modules, components, and products, see the topic on Identifying modules, components, and products in *z/OS MVS Diagnosis: Reference*.
- To understand ENQ contention and additional analysis steps, see the topic on Contention management in *z/OS MVS Planning: Global Resource Serialization*.
- For System Logger, see the topic on Associating latch contention with a logger TCB or WEB in *z/OS MVS Diagnosis: Reference*.

CPU analysis

Runtime Diagnostics provides a point in time check of any address space that is using more than 95% of the capacity of a single CPU. High CPU usage might indicate that the address space is in a loop (see Figure 6 on page 41). The analysis is a one second sample interval based on the capacity of a single CPU within the LPAR. It is possible for the usage to be reported greater than 100% if the address space has multiple TCBs and several are each using a high percentage of the CPU capacity.

```
f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 568
SUMMARY: SUCCESS
 REQ: 003 TARGET SYSTEM: SY1      HOME: SY1      2010/12/21 - 13:45:49
 INTERVAL:  60 MINUTES
 EVENTS:
  FOUND: 02 - PRIORITIES: HIGH:02  MED:00  LOW:00
  TYPES: HIGHCPU:01
  TYPES: LOCK:01
----------------------------------------------------------------------
EVENT 01: HIGH - HIGHCPU      - SYSTEM: SY1      2010/12/21 - 13:45:50
ASID CPU RATE:99%     ASID:002E   JOBNAME:IBMUSERX
STEPNAME:STEP1    PROCSTEP:         JOBID:JOB00045 USERID:IBMUSER
JOBSTART:2010/12/21 - 11:22:51
  ERROR: ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MIGHT BE LOOPING.
 ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
EVENT 02: HIGH - LOCK         - SYSTEM: SY1      2010/12/21 - 13:45:50
HIGH LOCAL LOCK SUSPENSION RATE - ASID:000A JOBNAME:WLM
STEPNAME:WLM      PROCSTEP:IEFPROC  JOBID:++++++++ USERID:++++++++
JOBSTART:2010/12/21 - 11:15:08
  ERROR: ADDRESS SPACE HAS HIGH LOCAL LOCK SUSPENSION RATE.
 ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
```

Figure 6. Runtime Diagnostics HIGHCPU and LOCK report

More details about CPU activity are found in the following publications:
- *z/OS RMF User's Guide*

Local lock suspension
Runtime Diagnostics provides a point in time check of local lock suspension for all address spaces. For the local lock suspension, Runtime Diagnostics calculates the amount of time an ASID is suspended waiting for the local lock. When an ASID is suspended more than 50% of the time waiting for a local lock, Runtime Diagnostics reports an event. For an example, see Figure 6.

Loop detection
Runtime Diagnostics looks through all tasks in all address spaces to determine whether a task is looping. Runtime Diagnostics examines various system information for indicators of consistent repetitive activity that are typical when a task is in a loop. When both a HIGHCPU event and a LOOP event (shown in Figure 7 on page 42) list the job name, the task in the job is likely in a loop. The normal corrective action is to cancel the job name listed.

```
f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 581
SUMMARY: SUCCESS
 REQ: 004 TARGET SYSTEM: SY1      HOME: SY1      2010/12/21 - 13:51:32
 INTERVAL:  60 MINUTES
 EVENTS:
  FOUND: 04 - PRIORITIES: HIGH:04  MED:00  LOW:00
  TYPES: HIGHCPU:01
  TYPES: LOOP:01 ENQ:01 LOCK:01
----------------------------------------------------------------------
EVENT 01: HIGH - ENQ          - SYSTEM: SY1      2010/12/21 - 13:51:32
ENQ WAITER  - ASID:0038 - JOBNAME:IBMUSER2 - SYSTEM:SY1
ENQ BLOCKER - ASID:002F - JOBNAME:IBMUSER1 - SYSTEM:SY1
QNAME: TESTENQ
RNAME: TESTOFAVERYVERYVERYVERYLOOOOOOOOOOOOOOOOOOOOOONGRNAME1234567...
  ERROR: ADDRESS SPACES MIGHT BE IN ENQ CONTENTION.
 ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE BLOCKING JOBS AND
 ACTION: ASIDS.
----------------------------------------------------------------------
EVENT 02: HIGH - HIGHCPU      - SYSTEM: SY1      2010/12/21 - 13:51:33
ASID CPU RATE:99%     ASID:002E   JOBNAME:IBMUSERX
STEPNAME:STEP1    PROCSTEP:         JOBID:JOB00045 USERID:IBMUSER
JOBSTART:2010/12/21 - 11:22:51
  ERROR: ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MIGHT BE LOOPING.
 ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
EVENT 03: HIGH - LOOP         - SYSTEM: SY1      2010/12/21 - 13:51:14
ASID:002E   JOBNAME:IBMUSERX   TCB:004FF1C0
STEPNAME:STEP1    PROCSTEP:         JOBID:JOB00045 USERID:IBMUSER
JOBSTART:2010/12/21 - 11:22:51
  ERROR: ADDRESS SPACE MIGHT BE IN A LOOP.
 ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
EVENT 04: HIGH - LOCK         - SYSTEM: SY1      2010/12/21 - 13:51:33
HIGH LOCAL LOCK SUSPENSION RATE - ASID:000A JOBNAME:WLM
STEPNAME:WLM      PROCSTEP:IEFPROC  JOBID:++++++++ USERID:++++++++
JOBSTART:2010/12/21 - 11:15:08
  ERROR: ADDRESS SPACE HAS HIGH LOCAL LOCK SUSPENSION RATE.
 ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
```

Figure 7. Runtime Diagnostics LOOP and HIGHCPU report

z/OS UNIX latch contention

Runtime Diagnostics gathers information about z/OS UNIX file system contention. When z/OS UNIX latch contention or waiting threads exist for greater than five minutes in z/OS UNIX, the z/OS UNIX file system latch contention and waiting threads record (as shown in Figure 8 on page 43) displays at the console.

```
F HZR,ANALYZE
HZR0200I RUNTIME DIAGNOSTICS RESULT
SUMMARY: SUCCESS
 REQ: 009 TARGET SYSTEM: SY1      HOME: SY1      2010/12/21 - 14:24:29
 INTERVAL:  60 MINUTES
 EVENTS:
  FOUND: 01 - PRIORITIES: HIGH:01  MED:00  LOW:00
  TYPES: OMVS:01
----------------------------------------------------------------------
EVENT 01: HIGH - OMVS         - SYSTEM: SY1      2010/12/21 - 14:24:29
ASID:000E - JOBNAME:OMVS
MOUNT LATCH WAITERS: 1
FILE SYSTEM LATCH WAITERS: 0
XSYS AND OTHER THREADS WAITING FOR z/OS UNIX: 1
  ERROR: z/OS UNIX MIGHT HAVE FILE SYSTEM LATCH CONTENTION.
 ACTION: D OMVS,W,A TO INVESTIGATE z/OS UNIX FILE SYSTEM LATCH
 ACTION: CONTENTION, ACTIVITY AND WAITING THREADS. USE YOUR SOFTWARE
 ACTION: MONITORS TO INVESTIGATE BLOCKING JOBS AND ASIDS.
----------------------------------------------------------------------
```

Figure 8. HZR event for the z/OS UNIX file system latch contention and waiting threads record

If all the counts are zero, the record does not display. Follow the instructions listed in the `ACTION` statement in the event record by issuing `D OMVS,W,A` command, which returns the ASID and job names of any latch waiters.

For more information about diagnosing and resolving latch contention, see the topics on z/OS UNIX System Services and in *z/OS MVS Diagnosis: Reference*. For zFS file system contention, see *z/OS Distributed File Service zFS Administration*.

GRS latch contention

The following example shows the Runtime Diagnostics event record that summarizes latch contention. When any of the counts are greater than zero, Runtime Diagnostics displays a summary record. Follow the `ACTION` statement in the summary to determine the source of the contention and further actions.

```
HZR0200I RUNTIME DIAGNOSTICS RESULT 928
SUMMARY: SUCCESS
 REQ: 002 TARGET SYSTEM: SY1      HOME: SY1      2010/12/21 - 14:32:01
 INTERVAL:  60 MINUTES
 EVENTS:
  FOUND: 02 - PRIORITIES: HIGH:02  MED:00  LOW:00
  TYPES: LATCH:02
----------------------------------------------------------------------
EVENT 01: HIGH - LATCH        - SYSTEM: SY1      2010/12/21 - 14:32:01
LATCH SET NAME: SYSTEST.LATCH_TESTSET
LATCH NUMBER:3         CASID:0039  CJOBNAME:TSTLATCH
TOP WAITER - ASID:0039 - JOBNAME:TSTLATCH - TCB/WEB:004E2A70
TOP BLOCKER- ASID:0039 - JOBNAME:TSTLATCH - TCB/WEB:004FF028
  ERROR: ADDRESS SPACES MIGHT BE IN LATCH CONTENTION.
 ACTION: D GRS,AN,LATCH,DEP,CASID=0039,LAT=(SYSTEST.L*,3),DET
 ACTION: TO ANALYZE THE LATCH DEPENDENCIES. USE YOUR SOFTWARE
 ACTION: MONITORS TO INVESTIGATE BLOCKING JOBS AND ASIDS.
----------------------------------------------------------------------
EVENT 02: HIGH - LATCH        - SYSTEM: SY1      2010/12/21 - 14:32:01
LATCH SET NAME: SYSTEST.LATCH_TESTSET
LATCH NUMBER:3         CASID:003B  CJOBNAME:TSTLATC2
TOP WAITER - ASID:003B - JOBNAME:TSTLATC2 - TCB/WEB:004E2A70
TOP BLOCKER- ASID:003B - JOBNAME:TSTLATC2 - TCB/WEB:004FF028
  ERROR: ADDRESS SPACES MIGHT BE IN LATCH CONTENTION.
 ACTION: D GRS,AN,LATCH,DEP,CASID=003B,LAT=(SYSTEST.L*,3),DET
 ACTION: TO ANALYZE THE LATCH DEPENDENCIES. USE YOUR SOFTWARE
 ACTION: MONITORS TO INVESTIGATE BLOCKING JOBS AND ASIDS.
----------------------------------------------------------------------
```

Figure 9. HZR event for the GRS latch contention event record

Additional Runtime Diagnostics analysis

For more information about the message descriptions, see IXC messages in *z/OS MVS System Messages, Vol 10 (IXC-IZP).*

IXC101I, IXC105I, IXC418I

Runtime Diagnostics compares messages IXC105I and IXC418I to determine whether they were issued for the same system as the IXC101I. If so, Runtime Diagnostics lists the activity in the IXC101I event. For example:

```
HZR0200I RUNTIME DIAGNOSTICS RESULT
SUMMARY: SUCCESS
 REQ: 001 TARGET SYSTEM: SY1      HOME: SY1      2009/06/09 - 10:34:28
 INTERVAL:  60 MINUTES
 EVENTS:
  FOUND: 02 - PRIORITIES: HIGH=02  MED=00  LOW=00
  TYPES: XCF=02
----------------------------------------------------------------------
EVENT 01: HIGH - XCF          - SYSTEM: SY1      2009/06/09 - 10:34:10
IXC101I SYSPLEX PARTITIONING IN PROGRESS  FOR SY3
  ERROR: MESSAGE IXC105I WAS ALSO ISSUED FOR SAME SYSNAME
 ACTION: LOOK FOR AND CORRECT ANY PROBLEMS WITH THE ETR CLOCK,
 ACTION: SIGNALING PATHS, OR COUPLE DATA SET.
----------------------------------------------------------------------
EVENT 02: HIGH - XCF          - SYSTEM: SY1      2009/06/09 - 10:34:10
IXC105I SYSPLEX PARTITIONING HAS COMPLETED FOR SY3
  ERROR: XCF REMOVED A SYSTEM FROM THE SYSPLEX.
 ACTION: LOOK FOR AND CORRECT ANY PROBLEMS WITH THE ETR CLOCK,
 ACTION: SIGNALING PATHS, OR COUPLE DATA SET.
----------------------------------------------------------------------
```

Figure 10. Runtime Diagnostics critical message analysis

IXL013I
: Runtime Diagnostics compares the structure name, job name, ASID, and connector name for multiple `IXL013I` messages. Runtime Diagnostics lists only the last message that contains a match of all four fields.

IXC431I
: Runtime Diagnostics compares the stalled identifiers for multiple `IXC431I` messages. It lists the last `IXC431I` message that contains the stalled identifier and analyzes the job name to determine whether it is the waiter for any ENQ contention. The stalled identifiers display as `ID: s#.r#` in the message description. For example,

```
STALLED AT sdate stime ID: s#.r#
```

IXC246E
: Runtime Diagnostics examines IOS messages, occurring 1 minute before and 1 minute after the system issues `IXC246E`. It looks for IOS messages that contain the same *devnum* (device number for the data set). If Runtime Diagnostics finds additional IOS messages with the same *devnum*, it lists the messages with the `IXC246E` events.

IXC467I
: Runtime Diagnostics does not list this message as an error event if the reason is `system partitioning`.

IXC585E
: Runtime Diagnostics compares the structure name and physical structure version for multiple IXC585E messages. It lists only the last message that contains a match of both fields.

Runtime Diagnostics messages

This section covers the following topics:
- “Understanding the messages Runtime Diagnostics issues”
- “Test messages ignored by Runtime Diagnostics” on page 46

Runtime Diagnostics uses a set of critical z/OS system messages for its analysis. See Chapter 5, “Messages that Runtime Diagnostics analyzes,” on page 51.

For the complete list of messages that Runtime Diagnostics issues, see the topic on HZR messages in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

Understanding the messages Runtime Diagnostics issues

If `HZR0201I` message returns when you enter the `F HZR,ANALYZE` command, there are no error events present on the system the command was run against. For example:

```
HZR0201I SUCCESS. TIME (2009/06/09 - 10:25:01). NO RUNTIME DIAGNOSTICS EVENTS WERE FOUND FOR SYSTEM: SY1
```

In this section, a target system is a system not equal to the home system. When Runtime Diagnostics analyzes a target system, not the home system, it can find critical messages and perform ENQ analysis. Runtime Diagnostics cannot do any processing that requires control block analysis. Consequently, when analyzing a target system, Runtime Diagnostics cannot find:
- z/OS UNIX file system latch contention

- GRS latch contention
- Loop detection
- CPU analysis
- Local lock suspension.

The following output shows an example of what you might receive when analyzing a target system:

```
HZR0200I RUNTIME DIAGNOSTICS RESULT 593
SUMMARY: SUCCESS - NO EVENTS FOUND
 REQ: 001 TARGET SYSTEM: SYS3     HOME: SY1      2010/12/21 - 14:52:50
 INTERVAL:  60 MINUTES
 EVENTS:
  FOUND: 00 - PRIORITIES: HIGH:00  MED:00  LOW:00
  PROCESSING BYPASSED:
   OMVS.......SPECIFIED TARGET SYSTEM IS NOT THE HOME SYSTEM.
   LATCHES....SPECIFIED TARGET SYSTEM IS NOT THE HOME SYSTEM.
   LOOP.......SPECIFIED TARGET SYSTEM IS NOT THE HOME SYSTEM.
   HIGHCPU....SPECIFIED TARGET SYSTEM IS NOT THE HOME SYSTEM.
   LOCK.......SPECIFIED TARGET SYSTEM IS NOT THE HOME SYSTEM.
----------------------------------------------------------------------
```

Figure 11. Message output for target system

Test messages ignored by Runtime Diagnostics

The Geographically Dispersed Parallel Sysplex (GDPS®) integrates Parallel Sysplex technology and remote copy technology to enhance application availability and improve disaster recovery. Runtime Diagnostics ignores the following messages when GDPS issues them as test messages:

- IXC102A
- IXC105I
- IXC256A

Runtime Diagnostics DEBUG options

Purpose

Restriction: Do not enable the debug option unless under the direction of IBM support.

If IBM Service determines they need more Runtime Diagnostics data, they might request that you specify the debug options. The `MODIFY` *`HZR`*`,ANALYZE,SYSNAME=`*`XXXXXXX`*`,DEBUG=(`*`XXXXXXX`*`,`*`XXXXXXX`*`)` command initiates an SVC dump to collect addition problem data.

Note: If the DEBUG option is requesting when there is no event detected, only the Runtime Diagnostics address space is dumped. If the DEBUG option is requested when an event is detected, both the HZR address space and the specific address spaces that are identified in the events are dumped.

Tip: Before using this option, ensure that the system is able to take an SVC dump. To determine which dump options are currently in effect, use the **DISPLAY DUMP** command.

Format

```
MODIFY HZR,ANALYZE,[SYSNAME=SYSNAME]([,DEBUG=ALL)|(LOOP|NOLOOP),(ENQ|NOENQ),
(LOCK|NOLOCK),(HIGHCPU|NOHIGHCPU),(MSGS|NOMSGS),(OMVS|NOOMVS),(LATCH|NOLATCH)])
```

Options

SYSNAME
The target system that Runtime Diagnostics analyzes. By default, all analysis is conducted against the home system.

Note: When the target system is not the home system, Runtime Diagnostics processes only the following debug options:
- ALL
- ENQ
- NOENQ
- MSGS
- NOMSGS

DEBUG
Indicates conditions under which Runtime Diagnostics initiates an SVC dump of the HZR address space and the address spaces that are associated with events displayed in message HZR0200I. Up to 15 address spaces can be reported in the dump. The debug option controls only when the system dump occurs; the debug option does not provide analysis of Runtime Diagnostics processing.

When entering two or more debug options, use parenthesis except with `ALL`, which is mutually exclusive. For example:

```
F HZR,ANALYZE,DEBUG=ALL
F HZR,ANALYZE,DEBUG=(LOOP,NOENQ)
```

Remember: Use the debug option only when under the direction of IBM Service.

ALL
Runtime Diagnostics initiates an SVC dump of the HZR address spaces of 15 address spaces and the 15 address spaces associated with each LOOP, HIGHCPU, ENQ, LOCK, OMVS and LATCH event that is displayed in message HZR0200I.

Note: The keyword ALL supersedes all other keywords. For Example, if you specify both All and MSG on the ANALYZE command, `F HZR,ANALYZE,DEBUG=(ALL,MSGS)`, MSGS is ignored.

LOOP
Runtime Diagnostics initiates an SVC dump of the HZR address space, along with a maximum of 15 address spaces that are associated with each *LOOP* event that displays in the `HZR0200I` message. If any data spaces exist for the address spaces dumped, they are also included in the dump.

NOLOOP
When no LOOP events are found during the ANALYZE request, Runtime Diagnostics dumps the HZR address space.

ENQ
Runtime Diagnostics initiates an SVC dump of the HZR address space along with a maximum of 15 address spaces that are associated with each *ENQ* event that displays in the `HZR0200I` message. If any data spaces exist for the address spaces dumped, they are also included in the dump.

NOENQ
: When no ENQ events are found during the ANALYZE request, Runtime Diagnostics dumps the HZR address space.

LOCK
: Runtime Diagnostics initiates an SVC dump of the HZR address space along with a maximum of 15 address spaces that are associated with each *LOCK* event that displays in the `HZR0200I` message. If any data spaces exist for the address spaces dumped, they are also included in the dump.

NOLOCK
: When no *LOCK* events are found during the ANALYZE request,

HIGHCPU
: Runtime Diagnostics initiates an SVC dump of the HZR address space along with a maximum of 15 address spaces that are associated with each *HIGHCPU* event that displays in the `HZR0200I` message. If any data spaces exist for the address spaces dumped, they are also included in the dump.

NOHIGHCPU
: When no *HIGHCPU* events are found during the ANALYZE request, Runtime Diagnostics dumps the HZR address space.

MSGS
: Runtime Diagnostics dumps the HZR address space when MESSAGE events are found during the ANALYZE request.

NOMSGS
: Runtime Diagnostics dumps the HZR address space when no MESSAGE events are found during the ANALYZE request.

OMVS
: Runtime Diagnostics initiates an SVC dump of the HZR address space, along with a maximum of 15 address spaces that are associated with each *OMVS* event that displays in the `HZR0200I` message. If any data spaces exist for the address spaces dumped, they are also included in the dump.

NOOMVS
: When no *OMVS* events are found during the ANALYZE request, HZR dumps the HZR address space.

LATCH
: Runtime Diagnostics initiates an SVC dump of the HZR address space along with a maximum of 15 address spaces that are associated with each *LATCH* event that displays in the `HZR0200I` message. If any data spaces exist for the address spaces dumped, they are also included in the dump.

NOLATCH
: When *NO LATCH* events are found, Runtime Diagnostics dumps the HZR address space .

Examples

Example:
Runtime Diagnostics shows a job consuming a high amount of CPU, but your monitors are not showing the job that is consuming high CPU. If

instructed by IBM Service, add the following debug option to the `MODIFY` command the next time Runtime Diagnostics runs:

```
F HZR,ANALYZE,DEBUG=HIGHCPU
```

If Runtime Diagnostics highlights the job with a *HIGHCPU* event, it initiates an SVC dump of relevant storage areas to assist IBM Service in diagnosing analysis processing.

Chapter 5. Messages that Runtime Diagnostics analyzes

BPX message analysis

This topic covers the messages Runtime Diagnostics uses for its analysis.

For more information on BPX messages for z/OS UNIX System Services, see *z/OS MVS System Messages, Vol 3 (ASB-BPX)*.

BPXF006I **A FILE SYSTEM WITH FILESYSTYPE** *type* **FAILED TO INITIALIZE. IT TERMINATED DURING INITIALIZATION**

Explanation: If prompted for restart, fix the problem and respond. If no prompt, the system will run without the physical file system.

BPXF020I **FILE SYSTEM** *name* **MAY BE DAMAGED. RETURN CODE =** *return_code*, **REASON CODE =** *reason_code*

Explanation: Access to files within the file system might still be possible. If an sdump was captured, provide it to IBM support.

BPXF029E **ROOT FILE SYSTEM** *name* **WAS NOT MOUNTED. RETURN CODE =** *return_code*, **REASON CODE =** *reason_code*

Explanation: Correct the problem in *bpxprmxx* or the superuser can enter correct data using `TSO/E mount` with "/" as the mountpoint.

BPXF076I **FILE SYSTEM INIT DELAYED BY ACTIVITY ON ANOTHER SYSTEM.**

Explanation: See the subsequent BPXF041I message in OPERLOG. Issue `D GRS,LATCH,C` on all systems to investigate latch contention.

BPXF083I **THE INDICATED FILE SYSTEM IS UNUSABLE UNTIL IT IS UNQUIESCED** *<name>***QUIESCING SYSTEM=***<sysname>* **PID=***<pid>* **JOB=***<jobname>* **LATCH=***<latchnum>*

Explanation: An authorized user might be able to unquiesce the file system from the ISPF shell. See the message documentation in *z/OS MVS System Messages, Vol 3 (ASB-BPX)*.

BPXF215E **ACCESS TO THE z/OS UNIX COUPLE DATA SET IS NOT AVAILABLE.**

Explanation: Review the error codes and correct the access problem. Issue the `SETXCF` couple command to enable the couple data set.

BPXF216E **FILE SYSTEM PARTITION CLEANUP IS DELAYED DUE TO** *text*

Explanation: See the subsequent BPXF041I message in OPERLOG. Issue `D GRS,LATCH,C` on all systems to investigate latch contention.

BPXF217E **FILE SYSTEM PARTITION CLEANUP FAILED DUE TO** *text*

Explanation: Issue `F BPXOINIT,FILESYS=D,EXCEPTION` to identify impacted file systems. Unmount and remount if file system does not recover.

BPXI026I **THE ETCINIT JOB COULD NOT BE STARTED.** *system_call* **RETURN CODE** *return_code* **REASON CODE** *reason_code*

Explanation: Creation for process failed for `/etc/init` or `/usr/sbin/init`.

Examine the return and reason codes. See the message documentation in *z/OS MVS System Messages, Vol 3 (ASB-BPX)*.

BPXI027I **THE ETCINIT JOB ENDED IN ERROR, EXIT STATUS** *exit_status*

Explanation: Examine the exit status code. See the message documentation in *z/OS MVS System Messages, Vol 3 (ASB-BPX)*.

BPXI031E **BPXOINIT FAILED TO INITIALIZE. RETURN CODE** *return_code* **REASON CODE** *reason_code*

Explanation: Examine the return and reason codes. Correct the error. A re-IPL of the system is required to start z/OS UNIX.

BPXI036E **UNIX SYSTEM SERVICES ARE NOT AVAILABLE**

Explanation: Correct the conditions that caused the failure. A re-IPL of the system is required to start z/OS UNIX.

BPXI043E MOUNT TABLE LIMIT HAS REACHED *limperc%* **OF ITS CURRENT CAPACITY OF** *limtot*

Explanation: See the message documentation in *z/OS MVS System Messages, Vol 3 (ASB-BPX)*.

BPXI060I *jobname* **RUNNING IN ADDRESS SPACE** *asid* **IS BLOCKING SHUTDOWN OF OMVS**

Explanation: Evaluate stopping the indicated job. The indicated job must be stopped to allow z/OS UNIX shutdown to complete.

BPXI062I *jobname* **RUNNING IN ADDRESS SPACE** *asid* **IS PREVENTING THE SHUTDOWN OF OMVS FROM COMPLETING**

Explanation: The indicated job is blocking z/OS UNIX shutdown. Evaluate stopping the indicated job. The indicated job must be stopped to allow z/OS UNIX shutdown to complete.

BPXI068I *jobname* **RUNNING IN ADDRESS SPACE** *asid* **IS USING** *text*

Explanation: Evaluate stopping the indicated job. The indicated job must be stopped to allow z/OS UNIX shutdown to complete

BPXI076E LATCH CONTENTION EXISTS THAT MUST BE RESOLVED PRIOR TO SHUTDOWN

Explanation: Issue `D GRS,C` to determine the nature of the contention. Evaluate canceling or forcing the address space that is causing contention.

BPXI084E OMVS SHUTDOWN IS STALLED IN FILE SYSTEM TERMINATION

Explanation: z/OS UNIX shutdown delayed while terminating file systems. Get an SVC dump of z/OS UNIX and its associated data spaces.

BPXM048I BPXOINIT FILESYSTEM SHUTDOWN INCOMPLETE. *notshutdown* **FILESYSTEM(S) ARE STILL OWNED BY THIS SYSTEM.** *mounted* **FILESYSTEM(S) WERE MOUNTED DURING THE SHUTDOWN PROCESS**

Explanation: Issue `D OMVS,F` to identify file systems that did not move or unmount. See the message documentation in *z/OS MVS System Messages, Vol 3 (ASB-BPX)*.

BPXP007E STARTING PHYSICAL FILE SYSTEM *pfsname* **IN ADDRESS SPACE** *spacename*

Explanation: Get an SVC dump of the indicated address space, z/OS UNIX and its associated data spaces. Provide the dump to IBM support.

IEA message analysis

For more information on IEA messages for the communication and console services, see *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IEA230E WTOR BUFFER SHORTAGE. 80% FULL

Explanation: Enter `D R,R` to display outstanding WTOR messages. Reply to them. Enter `K M,RLIM=nnnn` to increase the RLIM buffer limit.

IEA231A SEVERE WTOR BUFFER SHORTAGE. 100% FULL

Explanation: Enter `D R,R` to display outstanding WTOR messages. Reply to them. Enter `K M,RLIM=nnnn` to increase the RLIM buffer limit.

IEA359E BUFFER SHORTAGE FOR RETAINED ACTION MESSAGES - 80% FULL

Explanation: Enter `D R,L` to display action messages, respond to them, or delete them using `K C,X,ID-ID (X = A,E OR CE)` where ID is the message numbers.

IEA360A SEVERE BUFFER SHORTAGE FOR RETAINED ACTION MESSAGES - 100% FULL

Explanation: Enter `D R,L` to display action messages, respond to them, or delete them using `K C,X,ID-ID (X = A,E OR CE)` where ID is the message numbers.

IEA404A SEVERE WTO BUFFER SHORTAGE - 100% FULL

Explanation: Enter `D C,B` to display WTO backlog; enter `K Q` to clear message queue; enter `K M,MLIM=nnnn` to increase the buffer limit for MLIM (maximum number of WTO messages allowed in the system).

IEA405E WTO BUFFER SHORTAGE - 80% FULL

Explanation: Enter `D C,B` to display WTO backlog. Enter `K Q` to clear message queue. Enter `K M,MLIM=nnnn` to increase the buffer limit for MLIM (maximum number of WTO messages allowed in the system).

IEA406I **WTO BUFFER SHORTAGE RELIEVED**

IEA611I **{COMPLETE|PARTIAL} DUMP ON** *dsname*

Explanation: Use IPCS to diagnose the problem and determine the action for a partial dump, see the message documentation.

IEA793A **NO SVC DUMP DATA SETS AVAILABLE FOR DUMPID=***dumpid* **FOR JOB (*MASTER*). USE THE DUMPDS COMMAND OR REPLY D TO DELETE THE CAPTURED DUMP**

Explanation: Enter `DD ADD,SMS=class` to add SMS classes, enter `DD ADD,VOL=VOLSER` to add DASD volumes, or reply `D` to delete the captured dump.

IEA799I **AUTOMATIC ALLOCATION OF SVC DUMP DATA SET FAILED DUMPID=***dumpid* **REQUESTED BY JOB (***jobname***)** *reason-text reason text2*

Explanation: Enter `D D` to view allocation status. Enter `DD ADD,VOL=VOLSER` to add dump resources.

IEA911E **{COMPLETE|PARTIAL} DUMP ON SYS1.DUMP***nn* **JOB (***jobname***) FOR ASIDS(***id,id,...***) [REMOTE DUMPS REQUESTED | REMOTE DUMP FOR SYSNAME:** *sysname***] INCIDENT TOKEN:***incident-token* **[SDRSN =** *vvvvvvvv wwwwwwww xxxxxxxx zzzzzzzz***]** **[***reason-text***] [ERRORID = SEQ***yyyyyy* **CPU***zz* **ASID***asid* **TIME***hh.mm.ss.f***]** **[TSOID =** *tsoid***] [ID =** *uuuuuuuuuu***DUMPid=***dumpid* **REQUESTED BY**

Explanation: Use IPCS to diagnose the problem and determine the action. For a partial dump, see the message documentation.

IEE message analysis

For more information about IEE messages for MVS, see *z/OS MVS System Messages, Vol 7 (IEB-IEE)*.

IEE012A **NO LONGER SAVING MESSAGES FOR HARDCOPY, LOGLIM REACHED.**

Explanation: Issue `V SYSLOG,HARDCPY` to activate `SYSLOG. K M,LOGLIM=nnnnn` to increase the LOGLIM value.

IEE601E **PROCESSOR (y) IS IN AN EXCESSIVE DISABLED SPIN LOOP WAITING FOR** *event* **HELD BY PROCESSOR (x). ACR IS ALREADY ACTIVE. SPIN WILL CONTINUE.**

Explanation: Reply `U`, `ABEND` or `TERM` to message IEE331A.

IEE711I **[SYSTEM UNABLE TO DUMP|SYSTEM DUMP NOT TAKEN. reason]**

Explanation: The system did not write the requested SVC dump. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IEE766E **BUFFER SHORTAGE FOR SYSTEM LOG - 60% FULL**

Explanation: `K M,LOGLIM=nnnnnn` to increase the WTL buffers.

IEE767A **SEVERE BUFFER SHORTAGE FOR SYSTEM LOG - 100% FULL**

Explanation: `K M,LOGLIM=nnnnnn` to increase the WTL buffers.

IEE786I **THE PAGEADD COMMAND ENCOUNTERED A FAILURE**

Explanation: Issue `R nn,U` to message IEE787A to continue PAGEADD processing. Issue `R nn,END` to message IEE787A to end PAGEADD processing.

IEE986E **SMF HAS USED** *nn%* **OF AVAILABLE BUFFER SPACE**

Explanation: Issue `D SMF` to check the status of the SMF data sets. Use the SMF dump program (IFASMFDP) to make a data set available for use.

IOS message analysis

For more information about IOS messages, see *z/OS MVS System Messages, Vol 9 (IGF-IWM)*.

IOS078I I/O REQUEST HAS TIMED OUT

Explanation: Run EREP to dump data from SYS1.LOGREC and provide it to IBM Support.

IOS078I I/O REQUEST HAS TIMED OUT

Explanation: Run EREP to dump data from SYS1.LOGREC and provide it to IBM Support.

IOS431I THE INDICATED SYSTEM HOLDS THE RESERVE ON THE DASD DEVICE.

Explanation: Issue `D GRS,DEV=dev` to investigate and resolve the contention. See the message documentation in *z/OS MVS System Messages, Vol 9 (IGF-IWM)*.

IOS1078I I/O REQUEST HAS TIMED OUT

Explanation: Run EREP to dump data from SYS1.LOGREC and provide it to IBM support.

IOS1079I I/O REQUEST HAS TIMED OUT

Explanation: Run EREP to dump data from SYS1.LOGREC and provide it to IBM support.

IRA message analysis

For more information about IRA messages for System Resource Manager, see *z/OS MVS System Messages, Vol 9 (IGF-IWM)*.

IRA100E SQA SHORTAGE

Explanation: Determine largest users of sqa/csa storage. Get an SVC dump of those users for analysis. Include SQA/CSA/RGN in the dump.

IRA101E CRITICAL SQA SHORTAGE

Explanation: Get an SVC dump of common storage to analyze its growth and determine if SQA/CSA allocation is adequate.

IRA103I SQA/ESQA HAS EXPANDED INTO CSA/ECSA BY xxxxx PAGES

Explanation: Evaluate the system requirement for sqa storage. Increase the size of the SQA parameter in the IEASYSXX parmlib member.

IRA200E AUXILIARY STORAGE SHORTAGE

Explanation: Issue PA PAGE=DSNAME to add auxiliary storage. Evaluate canceling jobs indicated by messages IRA206I and IRA210E.

IRA201E CRITICAL AUXILIARY STORAGE SHORTAGE

Explanation: Issue PA PAGE=DSNAME to add auxiliary storage. Evaluate canceling any jobs indicated by messages ira206i and ira210e.

IRA206I *uuuuuuuu* **ASID** *aaaa* **FRAMES** *ffffffffff* **SLOTS** *ssssssssss* % **OF AUX** *nn.n*

Explanation: Get an SVC dump of the indicated ASIDs for analysis. Evaluate canceling one or more of the indicated ASIDs.

IRA210E *uuuuuuuu* **ASID** *aaaa* **SET NON DISPATCHABLE Frames+Slots** *vvvvvvvvvv* **RATE** *rrrrrr*

Explanation: Issue PA PAGE=DSNAME to add auxiliary storage. Evaluate canceling the indicated job.

IRA211I *uuuuuuuu* **ASID** *aaaa* **SET NON DISPATCHABLE Frames+Slots** *vvvvvvvvvv* **RATE** *rrrrrr*

Explanation: Issue PA PAGE=DSNAME to add auxiliary storage. Evaluate canceling the indicated job.

IRA220I CRITICAL AUXILIARY SHORTAGE*text*

Explanation: The *text* is:

```
! ## ! USER     ! ASID ! PAGES      ! SLOTS      !
+----+----------+------+------------+------------+
! ii ! uuuuuuuu ! aaaa ! xxxxxxxxxx ! zzzzzzzzzz !
! ii ! uuuuuuuu ! aaaaS! xxxxxxxxxx ! zzzzzzzzzz !
! ii ! uuuuuuuu ! aaaaN! xxxxxxxxxx ! zzzzzzzzzz !
```

Issue `PA PAGE=DSNAME` to add auxiliary storage. Evaluate canceling a specific consumer by replying to message IRA221D.

IRA400E *return-code*, **PAGEABLE STORAGE SHORTAGE**

Explanation: Evaluate canceling jobs identified by messages IRA403E and IRA404I.

IRA401E *return-code*, **CRITICAL PAGEABLE STORAGE SHORTAGE**

Explanation: Evaluate canceling jobs identified by messages IRA403E and IRA404I.

IXC message analysis

For more information on IXC messages for Cross System Coupling Facility (XCF) , see *z/OS MVS System Messages, Vol 10 (IXC-IZP)*.

IXC101I **SYSPLEX PARTITIONING IN PROGRESS FOR** *sysname* **REQUESTED BY** *jobname* **REASON:** *reason*

Explanation: See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC105I **A SYSTEM HAS BEEN REMOVED FROM THE SYSPLEX.**

Explanation: See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC244E **XCF CANNOT USE {PRIMARY | ALTERNATE} SYSPLEX COUPLE DATA SET** *dsname*, **{ON VOLSER** *volser*, **| VOLSER N/A,}** *text*

Explanation: Check data set name and VOLSER to ensure they are correct.

IXC246E *typename* **COUPLE DATA SET** *dsname* **ON VOLSER volser, DEVN***devnum*, **HAS BEEN EXPERIENCING I/O DELAYS FOR** *delaysec* **SECONDS.**

Explanation: See messages IOS078I and IOS1078I for the same device number and see the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC255I **UNABLE TO USE DATA SET** *dsname* **AS THE {PRIMARY|ALTERNATE} FOR** *typename*: *text* **[RELEVANT***typename* **COUPLE DATA SET FORMAT INFORMATIONPRIMARY FORMAT LEVEL:** *fmtlevel* **FORMAT KEYWORDS:** *fmtinfo* **ALTERNATE FORMAT LEVEL:** *fmtlevel* **FORMAT KEYWORDS:** *fmtinfo***]**

Explanation: See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC256A **REMOVAL OF {PRIMARY|ALTERNATE} COUPLE DATA SET** *dsname* **FOR** *typename* **CANNOT FINISH THE {ACTION|COMPLETE} PHASE UNTIL THE FOLLOWING SYSTEM(S) ACKNOWLEDGE THE REMOVAL:** *syslist*

Explanation: Missing response delaying the removal of a couple data set. If the condition persists, see the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC259I **I/O ERROR ON DATA SET** *dsname* **FOR** *typename*, **VOLSER** *volser*, *modname*,*post-code*,*text*

Explanation: Reply COUPLE=xx to message IXC207A. (xx = the suffix of the COUPLExx parmlib member to be used by XCF initialization.)

IXC267E **PROCESSING WITHOUT AN ALTERNATE COUPLE DATA SET FOR** *typename*. **ISSUE SETXCF COMMAND TO ACTIVATE A NEW ALTERNATE.**

Explanation: No alternate couple data set defined for indicated function. Issue `SETXCF COUPLE,TYPE=typename,ACOUPLE=DSNAME` to activate a new alternate couple data set.

IXC406I **THIS SYSTEM IS CONNECTED TO ETR NET ID=***xx*. **THE OTHER ACTIVE SYSTEMS IN THE SYSPLEX ARE USING ETR NET ID=***yy*. **EFFECTIVE CLOCK VALUES ARE NOT CONSISTENT.**

Explanation: Correct any ETR problem and retry with COUPLExx parmlib member or correct any improperly defined ETR time offsets.

IXC427A **SYSTEM** *sysname* **HAS NOT UPDATED STATUS SINCE** *hh:mm:ss* **BUT IS SENDING XCF SIGNALS. XCF SYSPLEX FAILURE MANAGEMENT WILL REMOVE** *sysname* **IF NO SIGNALS ARE RECEIVED WITHIN A** *interval* **SECOND INTERVAL**

Explanation: Determine and resolve any contention issues with the sysplex couple data set. Reply to message IXC426D.

IXC430E **SYSTEM** *sysname* **HAS STALLED XCF GROUP MEMBERS**

Explanation: Issue D XCF,G to see groups with stalled members. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC431I *text*

Explanation: Issue `D XCF,G,grpname,membername` for more information. For the complete message text, see the IXC431I messages in *z/OS MVS System Messages, Vol 10 (IXC-IZP)*

IXC432I *text*

Explanation: The indicated XCF group member is no longer stalled. No action needed.

IXC440E **SYSTEM** *hurtsys* **IMPACTED BY STALLED XCF GROUP MEMBERS ON SYSTEM** *stallsys*

Explanation: Issue `D XCF,G` to see groups with stalled members. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC446I **SYSTEM** *sysname* **IS IN MONITOR-DETECTED STOP STATUS BUT IS SENDING XCF SIGNALS. SFM WILL TAKE SSUM ACTION AT** *actiontime* **IF SYSTEM REMAINS IN THIS STATE.**

Explanation: The indicated system has not updated its system status. Investigate the XCF couple data sets for contention or poor performance.

IXC467I *command dir pathname* **RSN:** *text*

Explanation: XCF IS TRYING TO RESTORE THE INDICATED PATH TO SERVICE. SEE THE MESSASGE DOCUMENTATION.

Explanation: The indicated system is waiting for system reset. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC512I **POLICY CHANGE IN PROGRESS FOR CFRM TO MAKE** *polname* **POLICY ACTIVE.** *numpend* **POLICY CHANGE(S) PENDING.**

Explanation: Issue D XCF,STR to show structures pending actions. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC518I **SYSTEM** *sysname* **NOT USING COUPLING FACILITY** *type*.*mfg*.*plant*.*sequence* **PARTITION:** *partition side* **CPCID:** *cpcid* **NAMED** *cfname* **REASON:** *text*

Explanation: The indicated coupling facility is unusable. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC519E **COUPLING FACILITY DAMAGE RECOGNIZED FOR COUPLING FACILITY** *type*.*mfg*.*plant*.*sequence* **PARTITION:** *partition side* **CPCID:** *cpcid* **NAMED** *cfname*

Explanation: Run EREP to dump data from SYS1.LOGREC and gather XCF/XES CTRACE records and provide them to IBM support.

IXC522I *rebuildtype* **FOR STRUCTURE** *strname* **IS BEING STOPPED** *action* **DUE TO** *reason* [*codetype stopcode*]

Explanation: See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC538I **DUPLEXING REBUILD OF STRUCTURE** *strname* **WAS NOT INITIATED BY MVS. REASON:** *reason*

Explanation: Duplexing rebuild start request of structure not performed. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC552I **DUPLEX REBUILD NEW STRUCTURE** *strname* **WAS ALLOCATED IN A COUPLING FACILITY THAT IS NOT FAILURE ISOLATED FROM THE OLD STRUCTURE.**

Explanation: Review your coupling facility configuration. When possible duplex failure isolation is strongly encouraged.

IXC553E **DUPLEXING REBUILD NEW STRUCTURE** *strname* **IS NOT FAILURE ISOLATED FROM THE DUPLEXING REBUILD OLD STRUCTURE.**

Explanation: Review your coupling facility configuration. When possible duplex failure isolation is strongly encouraged.

IXC573I *phase* **PROCESSING DURING A SYSTEM-MANAGED** *process* **FOR STRUCTURE** *strname* **ENCOUNTERED AN ERROR. ERROR DATA:** *reason* [*reldata1 reldata2 reldata3 reldata4*] **AUTO VERSION:** *procid1 procid2*

Explanation: Error encountered during rebuild or duplex rebuild process. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC585E **STRUCTURE** *strname* **IN COUPLING FACILITY** *cfname*, **PHYSICAL STRUCTURE VERSION** *physver1 physver2*, **IS AT OR ABOVE STRUCTURE FULL MONITORING THRESHOLD OF** *thresh%*. **ENTRIES: IN-USE:** *nnnnnnnn*, **TOTAL:** *pppppppp, pct%* **FULL [ELEMENTS: IN-USE:** *nnnnnnnn*, **TOTAL:** *pppppppp, pct%* **FULL] [EMCS: IN-USE:** *nnnnnnnn*, **TOTAL:** *pppppppp, pct%* **FULL]**

Explanation: Issue `D XCF,STR,STRNAME=strname` to get structure information. Iincrease structure size or take action against application.

IXC615I **GROUP** *grpname* **MEMBER** *membername* **JOB** *jobname* **ASID** *asidtext*

Explanation: XCF terminated the group member to resolve critical problem. Restart the affected application, subsystem, or system. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC631I **GROUP** *grpname* **MEMBER** *membername* **JOB** *jobname* **ASID** *asid***STALLED, IMPACTING SYSTEM** *sysname* **{WHICH IS IN PARTITIONING }**

Explanation: Issue `D XCF,G,grpname,membername` to get more information. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC633I *text* **GROUP gnme MEMBER mnme JOB jnme ASID asid{DEEMED | CONFIRMED} IMPAIRED AT ipdate iptime ID: s#.r#LAST MSGX: sgdate sgtime sgexit STALLED sgwork PENDINGQLAST GRPX: grdate grtime grexit STALLED grwork PENDINGQLAST STAX: stdate sttime stexit STALLED**

Explanation: Indicated XCF group member is not operating normally. See message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC635E **SYSTEM** *sysname* **HAS IMPAIRED XCF GROUP MEMBERS**

Explanation: The indicated system has impaired group members. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC700E **SYSPLEX COUPLE DATA SET LIMIT REACHED, FUTURE REQUESTS MAY BE REJECTED.** *text*

Explanation: Issue `D XCF,COUPLE` to review maximum values in the sysplex couple data set. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXC800I **ELEMENTS FROM TERMINATED SYSTEM** *sysname* **NOT RESTARTED.** *text*

Explanation: Indicated system had jobs that could not be restarted. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXL message analysis

For more information on IXL messages for Cross System Extended Services (XES), see *z/OS MVS System Messages, Vol 10 (IXC-IZP)*.

IXL010E **NOTIFICATION RECEIVED FROM COUPLING FACILITY** *type.mfg.plant.sequence* **PARTITION:** *partition side* **CPCID:** *cpcid* **NAMED** *cfname cfservrecord*

Explanation: Run EREP to dump data from SYS1.LOGREC and provide it to IBM support.

IXL013I *requesttype* **REQUEST FOR STRUCTURE** *structure-name* **FAILED. JOBNAME:** *jobname* **ASID:** *asid* **CONNECTOR NAME:** *connector-name* **IXLCONN RETURN CODE:** *return-code*, **REASON CODE:** *reason-code errortype* **CONADIAG***diagn: diagvalue*

Explanation: See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

Note: Only the last occurrence of message IXL013I is displayed when structure name, jobname, ASID, and connector name match.

IXL015I *strtype* **ALLOCATION INFORMATION FOR STRUCTURE** *structure-name* **CONNECTOR NAME:** *connector-name* **CFNAME ALLOCATION STATUS/FAILURE REASON------ -------------------------------cfname text [diag]**

Explanation: Connector statistical information for indicated connector. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXL0130 **CONNECTOR STATISTICS FOR LOCK STRUCTURE** *structure-name***, CONNECTOR** *connector-name: n*

Explanation: Connector statistical information for indicated connector. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXL040E **CONNECTOR NAME:** *connector-name***, JOBNAME:** *jobname***, ASID:** *asid* **HAS** *text***.** *process* **FOR STRUCTURE** *structure-name* **CANNOT CONTINUE. MONITORING FOR RESPONSE STARTED:** *mm/dd/yyyy hh:mm:ss***. DIAG:** *x*

Explanation: Evaluate job and gather XES CTRACE for specific connector terminate non-responsive job. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXL041E **CONNECTOR NAME:** *connector-name***, JOBNAME:** *jobname***, ASID:** *asid* **HAS NOT RESPONDED TO THE** *event* **FOR SUBJECT CONNECTION:** *subject-connector-name***.** *process* **FOR STRUCTURE** *structure-name* **CANNOT CONTINUE. MONITORING FOR RESPONSE STARTED:** *mm/dd/yyyy hh:ss:mm***. DIAG:** *x*

Explanation: Evaluate job and gather XES CTRACE for specific connector; terminate non-responsive job. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXL044I **COUPLING FACILITY** *cfname* **HAS EXPERIENCED** *IfccCount* **INTERFACE CONTROL CHECKS ON CHPID** *chpid* **DURING THE LAST** *interval* **SECONDS.**

Explanation: Run EREP to dump data from SYS1.LOGREC and provide it to IBM support.

IXL045E **[REBUILD] CONNECTOR NAME:** *connector-name***, JOBNAME:** *jobname***, ASID:** *asid* **FOR STRUCTURE** *structure-name* **MAY BE ENCOUNTERING DELAYS DUE TO LIMITED XES SRB SCHEDULING.**

Explanation: Run EREP to dump data from SYS1.LOGREC and gather XES CTRACE and system log and provide it to IBM support. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXL049E **HANG RESOLUTION ACTION FOR CONNECTOR NAME:** *conname* **TO STRUCTURE** *strname***, JOBNAME:** *jobname***, ASID:** *asid***:** *actiontext*

Explanation: See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXL158I **PATH** *chpid* **IS NOW NOT-OPERATIONAL TO CUID:** *cuid* **COUPLING FACILITY** *type***.***mfg***.***plant***.***sequence* **PARTITION:** *partition side* **CPCID:** *cpcid*

Explanation: Indicated channel not operational. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXL159E **COUPLING SUPPORT FACILITY HARDWARE ERROR DETECTED.**

Explanation: Run EREP to dump data from SYS1.LOGREC and provide it to IBM Support.

IXL160E **CF REQUEST TIME ORDERING: REQUIRED AND NOT-ENABLED** *text reason*

Explanation: Coupling facility hardware configuration problem. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

IXL162E **CF REQUEST TIME ORDERING: REQUIRED AND WILL NOT BE ENABLED** *text reason*

Explanation: Coupling facility hardware configuration problem. See the message documentation in *z/OS MVS System Messages, Vol 6 (GOS-IEA)*.

Chapter 6. Using OPERLOG

The operations log (OPERLOG) is a log stream that uses system logger to record and merge communications about programs and system functions from each system in a sysplex. Use OPERLOG as your hardcopy medium when you need a permanent log about operating conditions and maintenance for all systems in a sysplex.

Restriction: OPERLOG is designed to run in a Parallel Sysplex where the log stream is shared among the systems in a CF structure. If you have a basic sysplex, do not attempt to configure OPERLOG. Even if you set up a DASD log stream for OPERLOG on each system, only one system can connect to its OPERLOG log stream at any time because the name SYSPLEX.OPERLOG is hardcoded in the Couple Data Set.

Determining hardcopy medium settings

Use the following step to determine how your installation's hardcopy medium is set up:

- Enter the `DISPLAY CONSOLES,HARDCOPY` command. The output displays the following information:
  - The hardcopy medium as SYSLOG, OPERLOG, or both
  - The criteria defined by your installation for selecting messages for the hardcopy message set
  - The number of messages waiting to be placed on the hardcopy medium

If you have already defined the log stream as SYSPLEX.OPERLOG in either the data administrative utility or in the IXGINVNT macro, use the `V OPERLOG,HARDCPY` to assign OPERLOG as the hardcopy medium. You can assign both `OPERLOG` and `SYSLOG` by issuing the command separately.

To define a log stream using the system logger services, see “Setting up OPERLOG.”

Setting up OPERLOG

The following instructions for setting up OPERLOG are a summary of the details found in “Systems Programmer's Guide to: z/OS System Logger” available from IBM Redbooks (http://www.ibm.com/redbooks). For complete details about defining the log stream, see Preparing to use system logger applications in *z/OS MVS Setting Up a Sysplex*.

Steps for setting up OPERLOG

Before you begin: You must define the logger subsystem.

Perform the following steps to set up OPERLOG.

1. Define the hardcopy device as OPERLOG in the HARDCOPY statement of the CONSOL*xx* parmlib member. You can change this setting using the `V OPERLOG,HARDCPY` command. Specify `V OPERLOG,HARDCOPY,OFF` to turn off OPERLOG.

2. Define that the correspondent coupling facility structure in the CFRM policy. For example,

```
//OPERLOG JOB CLASS=A,MSGCLASS=A
//POLICY EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
   DATA TYPE(CFRM)
   STRUCTURE NAME(OPERLOG)
         SIZE(40448)
         INITSIZE(40448)
         PREFLIST(FACIL01,FACIL02)
```

3. Activate the CFRM policy using the `START,POLICY,TYPE=CFRM,POLNAME=polname` command, or the COUPLE*xx* parmlib member.
4. Define the log stream to the LOGR policy. The following example is for illustrative purposes only; you must follow the recommendations in *z/OS MVS Setting Up a Sysplex* and *z/OS MVS Programming: Assembler Services Guide*:

```
//OPERLOG JOB CLASS=A,MSGCLASS=A
//POLICY EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
 DATA TYPE(LOGR)
 DEFINE STRUCTURE NAME(OPERLOG)
  LOGSNUM(1)
  MAXBUFSIZE(4092)
  AVGBUFSIZE(512)
 DEFINE LOGSTREAM NAME(SYSPLEX.OPERLOG)
  STRUCTNAME(OPERLOG)
  LS_DATACLAS(LOGR4K)
  HLQ(IXGLOGR)
  LS_SIZE(1024)
  LOWOFFLOAD(0)
  HIGHOFFLOAD(80)
  STG_DUPLEX(NO)
  RETPD(30)
  AUTODELETE(No)
```

5. Create the security definitions for RACF (or for the equivalent security product). In the following example, the SYSPLEX.OPERLOG of the LOGSTRM resource CLASS is given READ permission, which allows everyone to browse the operations log and *userid1* has UPDATE access level, which allows *userid1* to delete records from the log stream. That is, the user ID associated with the job running the IEAMDBLG program. For example:

```
RDEFINE LOGSTRM SYSPLEX.OPERLOG UACC(READ)
PERMIT SYSPLEX.OPERLOG CLASS(LOGSTRM) ID(userid1) ACCESS(UPDATE)
SETROPTS CLASSACT(LOGSTRM)
```

   Note: This example is for illustrative purposes only. Follow the guidelines for your installation.
6. After you activate OPERLOG, you must manage the way in which records are handled.

   SYS1.SAMPLIB contains a sample program, IEAMDBLG, to read log blocks from the OPERLOG log stream and convert them to SYSLOG format. The program is an example of how to use the services of the MVS system logger to retrieve and delete records from the OPERLOG stream. It reads the records created in a given time span, converts them from Message Data Block (MDB) format to hardcopy log format (HCL or JES2 SYSLOG), and writes the SYSLOG-format records to a file. It also has an option to delete from the stream all the records created prior to a given date. When you use the delete option, a suggestion is to first copy the records on alternate media, and then conditionally delete them on a separate JCL step to ensure that you have a copy of the data before deleting. If you do not run them on two separate conditional steps, deletion occurs simultaneously with copy without any guarantee that the copy process was successful.

For additional details about handling log data, see the topic on “Managing log data: How much? For how long?” in *z/OS MVS Setting Up a Sysplex*.

You know you are done when you enter the `DISPLAY CONSOLES,HARDCOPY` command and see `OPERLOG`.

Part 3. Predictive Failure Analysis

Soft failures are abnormal yet allowable behaviors that can slowly lead to the degradation of the operating system. To help eliminate soft failures, use Predictive Failure Analysis (PFA). PFA is intended to detect abnormal behavior early enough to allow you to correct the problem before it affects your business. PFA uses remote checks from IBM Health Checker for z/OS to collect data about your installation, and then uses machine learning to analyze this historical data to identify abnormal behavior. It warns you by issuing an exception message when a system trend might cause a problem. To help you correct the problem, it identifies a list of potential issues. PFA can invoke Runtime Diagnostics to analyze and report insufficient metric activity for specific checks and provide the next action that you can take to avoid a problem.

Chapter 7. Predictive Failure Analysis overview and installation

This chapter contains the following information:

- "Avoiding soft failures"
- "Overview of Predictive Failure Analysis"
  - "How PFA works with a typical remote check" on page 66
  - "How PFA interacts with IBM Health Checker for z/OS" on page 67
  - "How PFA invokes Runtime Diagnostics" on page 67
  - "Migration considerations for PFA" on page 69
  - "How PFA uses the ini file" on page 71
- "Installing PFA" on page 71
  - "Installing PFA in a z/OS UNIX shared file system environment" on page 75
  - "Updating the Java path" on page 76

Avoiding soft failures

Unlike typical problems or hard failures that have a clear start and a clear cause, soft failures are caused by abnormal, but allowable behavior. Because the cause of the problem is dependent on a certain sequence or combination of events that are unique and infrequent, a solution is often difficult to determine. Multiple atypical, but legal actions performed by components on the z/OS image cause most soft failures. By design, most components of z/OS are stateless and are therefore unable to detect soft failures caused by atypical behavior.

A classic example is the exhaustion of common storage usage. A low priority, authorized task obtains common storage, but obtains significantly more common storage than usual. Then, a critical authorized system component fails while attempting to obtain a normal amount of common storage. Although the problem occurs in the second critical component, this second component is actually the victim. The first component caused the problem and is considered the villain. Soft failures usually occur in four generic areas:

- Exhaustion of shared resources
- Recurring or recursive failures often caused by damage to critical control structures
- Serialization problems such as classic deadlocks and priority inversions
- Unexpected state transition

z/OS has developed Predictive Failure Analysis (PFA) to help eliminate these soft failures.

Overview of Predictive Failure Analysis

Predictive Failure Analysis (PFA) is designed to predict potential problems with your systems. PFA extends availability by going beyond failure detection to predict problems before they occur. PFA provides this support using remote checks from IBM Health Checker for z/OS to collect data about your installation. Using this data, PFA constructs a model of the expected or future behavior of the z/OS images, compares the actual behavior with the expected behavior, and if the behavior is abnormal, PFA issues a health check exception. PFA uses a z/OS UNIX System Services (z/OS UNIX) file system to manage the historical and problem data that it collects.

Here is an LPAR view of the PFA components:

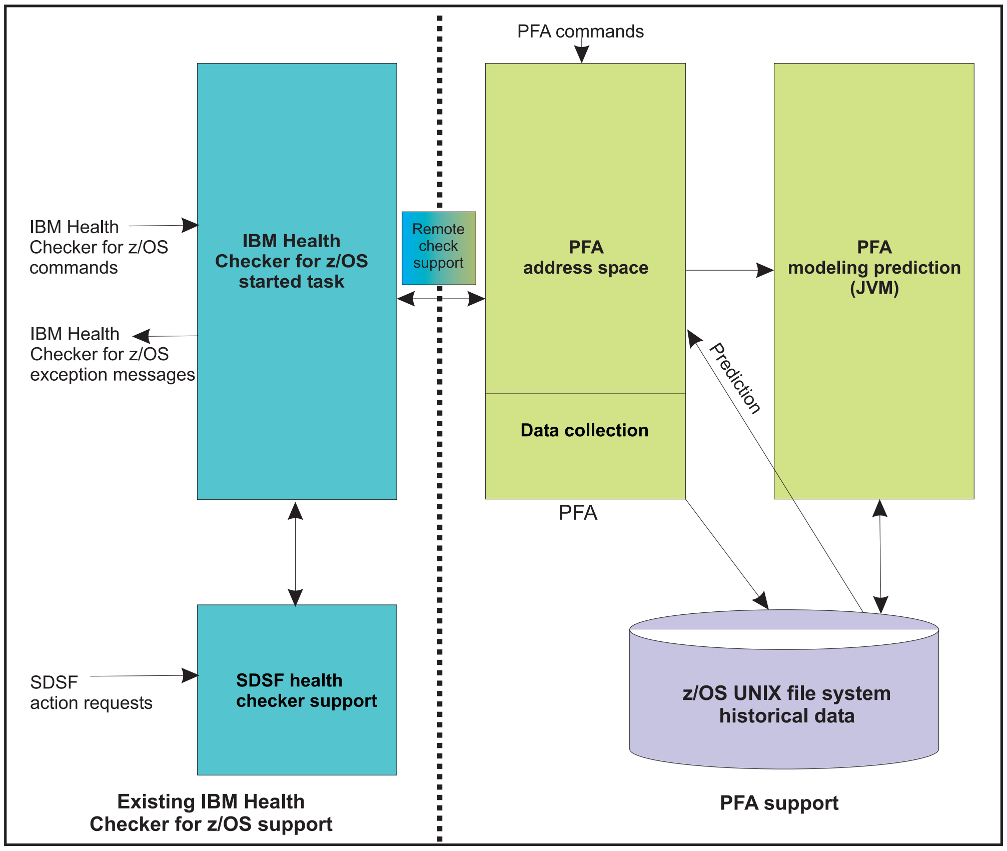

PFA commands
IBM Health Checker for z/OS commands
IBM Health Checker for z/OS exception messages
IBM Health Checker for z/OS started task
Remote check support
PFA address space
PFA modeling prediction (JVM)
Prediction
Data collection
PFA
SDSF action requests
SDSF health checker support
z/OS UNIX file system historical data
Existing IBM Health Checker for z/OS support
PFA support

Figure 12. LPAR view of the PFA components

PFA creates report output in the following ways:

- In a z/OS UNIX file that stores the list of suspect tasks. The individual checks contain descriptions of the directory and file names.
- In an IBM Health Checker for z/OS report that is displayed by z/OS System Display and Search Facility (SDSF) and the message buffer.
- Your installation can also set up IBM Health Checker for z/OS to send output to a log stream. After you set it up, you can use the HZSPRINT utility to view PFA check output in the message buffer or in the log stream. For complete details, see Using the HZSPRINT utility in *IBM Health Checker for z/OS User's Guide*.

How PFA works with a typical remote check

PFA_COMMON_STORAGE_USAGE is a remote check that evaluates the common storage use of each system. PFA, running in its own address space, periodically collects common storage area (CSA + SQA) data from the system on which the check is running. The check writes the CSA usage data, at intervals, to a z/OS UNIX file. The check identifies a list of common storage users that are abnormal and that might contribute to exhausting common storage. PFA issues an exception message to alert you if there is a potential common storage problem and provides a list of suspect tasks. You can then examine the list and stop the cause of the potential problem or move critical work off the LPAR.

How PFA interacts with IBM Health Checker for z/OS

When PFA issues an exception, the PFA check does not continue to issue exceptions to the console until the check determines a new exception must be issued or the exception resolves. For some checks, the new exception is always issued after a new model occurs. For other checks, the data must change significantly or the exception message must be different. For all checks, the check continues to run at the defined interval making the latest exception report data available using the CK panel in SDSF.

How PFA invokes Runtime Diagnostics

Runtime Diagnostics is an MVS utility (component HZR) that can perform some of the same tasks you might manually perform when looking for a the cause of a hung address space as well as other tasks. See Chapter 4, “Runtime Diagnostics,” on page 35 for complete details about Runtime Diagnostics.

PFA can invoke Runtime Diagnostics to analyze and report insufficient metric activity from the PFA_ENQUEUE_REQUEST_RATE check, PFA_MESSAGE_ARRIVAL_RATE check, and PFA_SMF_ARRIVAL_RATE check. For details and examples, see:
- “PFA_ENQUEUE_REQUEST_RATE” on page 97
- “PFA_MESSAGE_ARRIVAL_RATE” on page 120
- “PFA_SMF_ARRIVAL_RATE” on page 136.

Note: PFA requires the Runtime Diagnostic address space (HZR) to be active on the system or systems running these checks for Runtime Diagnostics to detect the insufficient metric activity.

When PFA issues a check exception because metric activity is unusually low, the IBM Health Checker for z/OS report includes information from Runtime Diagnostics. The Runtime Diagnostics information in the report points to the specific job or address space and provides the next action you can take. The additional Runtime Diagnostic output can help you quickly determine your next course of action and possibly help you avoid additional problems.

The following is an example of the Runtime Diagnostics output that might appear in the message arrival rate check when PFA determined the tracked jobs had a lower than expected message arrival rate (for AIH206E):

```
.
.
.
Persistent address spaces with low rates:
                                      Predicted Message
                   Message              Arrival Rate
Job                Arrival
Name     ASID         Rate       1 Hour      24 Hour        7 Day
________ ____ ____________ ____________ ____________ ____________
JOBS4    0027         1.17        23.88        22.82        15.82
JOBS5    002D         0.30         8.34        11.11        12.11

Runtime Diagnostics Output:

Runtime Diagnostics detects a problem in job: JOBS4

EVENT 06: HIGH - HIGHCPU - SYSTEM: SY1 2011/06/12 - 13:28:46
ASID CPU RATE: 96% ASID: 0027 JOBNAME: JOBS4
STEPNAME: STEPA PROCSTEP: STEPA JOBID: STC00042 USERID: ++++++++
JOBSTART: 2011/06/12 - 13:28:35
ERROR: ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MAY BE LOOPING.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
EVENT 07: HIGH - LOOP - SYSTEM: SY1 2011/06/12 - 13:28:46
ASID: 0027 JOBNAME: JOBS4 TCB: 004E6850
STEPNAME: STEPA PROCSTEP: STEPA JOBID: STC00042 USERID: ++++++++
JOBSTART: 2011/06/12 - 13:28:35
ERROR: ADDRESS SPACE APPEARS TO BE IN A LOOP.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
```

Figure 13. Runtime Diagnostics report within the PFA message arrival rate check

The following is an example of the Runtime Diagnostics output that might appear in the SMF arrival rate check when PFA determines the tracked jobs exception report for jobs that had a lower than expected SMF arrival rate (for AIH208E):

```
.
.
.
Persistent address spaces with low rates:

                                        Predicted SMF
                  SMF                    Arrival Rate
Job            Arrival
Name     ASID     Rate       1 Hour      24 Hour        7 Day
________ ____ ____________ ____________ ____________ ____________
TRACKED4 0027         0.20        23.88        22.82        15.82
TRACKED5 0034         0.01        12.43        11.11         8.36

Runtime Diagnostics Output:

Runtime Diagnostics detected a problem in job: TRACKED4

EVENT 06: HIGH - HIGHCPU - SYSTEM: SY1 2011/06/12 - 13:28:46
ASID CPU RATE: 96% ASID: 0027 JOBNAME: TRACKED4
STEPNAME: STEPA PROCSTEP: STEPA JOBID: STC00042 USERID: ++++++++
JOBSTART: 2011/06/12 - 13:28:35
ERROR: ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MAY BE LOOPING.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
EVENT 07: HIGH - LOOP - SYSTEM: SY1 2011/06/12 - 13:28:46
ASID: 0027 JOBNAME: TRACKED4 TCB: 004E6850
STEPNAME: STEPA PROCSTEP: STEPA JOBID: STC00042 USERID: ++++++++
JOBSTART: 2011/06/12 - 13:28:35
ERROR: ADDRESS SPACE APPEARS TO BE IN A LOOP.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
```

Figure 14. Runtime Diagnostics report within the SMF arrival rate check

Migration considerations for PFA

This section covers the following topics:

- "Using the *migrate* or *new* parameters when running AIRSHREP.sh"
- "How PFA uses the ini file" on page 71

Find complete details about installing PFA in "Steps for installing PFA" on page 72.

Using the *migrate* or *new* parameters when running AIRSHREP.sh

When migrating from z/OS V1R12 to V2R1, you are required to run AIRSHREP.sh with either the *new* or *migrate* option. When migrating from z/OS V1R13 to V2R1, you are not required to run AIRSHREP.sh; however, if you want to delete all previous release data, run AIRSHREP.sh with the *new* option.

When running the install script AIRSHREP.sh from the home directory PFA is using or when using the sample JCL for batch provided in SYS1.SAMPLIB, provide one of the following parameters:

migrate: Use the *migrate* parameter to preserve PFA history data from the previous release. The *migrate* option is recommended for all installations that previously used PFA and are migrating from z/OS V1R12.

| *new*: Use the *new* parameter if you are installing PFA for the first time or if you
| want to delete everything from previous releases and start PFA with empty
| directories.

If you do not append the *migrate* or *new* parameter or specify the parameter incorrectly, the script fails.

If you specify the *migrate* parameter when running the install script:

1. The script creates the directory structures for all checks that have not been previously installed on your system.
2. The script copies the first ini file found from one of the existing check directories to the `/etc/PFA/` directory starting with the `pfa_directory/PFA_COMMON_STORAGE_USAGE/` check. By copying an existing ini file, the Java configuration from previous installations of PFA for an existing check is automatically applied. If you do not want this Java configuration for all the checks, you can create the ini files on a per-check basis. For more information, see "How PFA uses the ini file" on page 71.
3. The script deletes the ini file from each of the existing check directories.
4. The script preserves the existing `EXCLUDED_JOBS` file for the message arrival rate or enqueue request rate check. If you did not previously define the `EXCLUDED_JOBS` files for the checks, the script creates the EXCLUDED_JOBS file in corresponding check config directory. For example, the following files are preserved:
   - `pfa_directory/PFA_MESSAGE_ARRIVAL_RATE/config` directory to exclude JES* jobs with generic system name *.
   - `pfa_directory/PFA_ENQUEUE_REQUEST_RATE/config` directory to exclude NETVIEW and *MASTER*.

   The EXCLUDED_JOBS file must exist in the local `/config` directory for the check on each LPAR to which it applies. The generic system name allows the file to be copied from one partition to other partitions without any changes. For additional details, see Table 5 on page 87.

If you specify the *new* parameter when running the install script:

1. The script deletes the existing check directories and creates a new directory structure for all the checks.
2. The script copies the ini file from the `/usr/lpp/bcp/samples/PFA/` directory to the `/etc/PFA/` directory.
3. The script creates the `EXCLUDED_JOBS` file the in corresponding check config directory. For example, the following files are created:
   - PFA_MESSAGE_ARRIVAL_RATE check to exclude JES* jobs with generic system name *
   - PFA_ENQUEUE_REQUEST_RATE check to exclude NETVIEW and *MASTER*.

   The EXCLUDED_JOBS files must exist in the local `/config` directory for the check on each LPAR to which it applies. The generic system name allows the file to be copied from one partition to other partitions without any changes. For additional details, see Table 5 on page 87.

Note: If the system does not find the `/etc/PFA` directory when running the install script, AIRSHREP, the ini file is copied to each check directory (for both new and migrate option).

How PFA uses the ini file

PFA processing uses the ini file from the check directory if one exists otherwise, it uses the ini file from `/etc/PFA`.

Note: The system creates `/etc/PFA` during the installation of z/OS. If your installation takes steps to delete the `/etc/PFA` directory that is created during installation, ensure you retain it to take advantage of the new PFA function. If the system does not find the `/etc/PFA` directory when running the install script, AIRSHREP, the ini file is copied to each check directory (for both new and migrate option). For more information, see the topic on Migrate /etc and /var system control files in *z/OS Migration*.

PFA can use the single ini file for all checks in the `/etc/PFA` directory, which means you only have to update and maintain one ini file. If you prefer a specific check uses a different level of Java than what is specified in the `/etc/PFA/ini` directory, provide an ini file in the check directory for the check. For example, create an ini file in the `pfa/PFA_MESSAGE_ARRIVAL_RATE/` directory if you want to use a different level of Java for the PFA_MESSAGE_ARRIVAL_RATE check.

If the path to the JDK for your installation is not the same as the path in the ini file in `/etc/PFA/` and in the checks' directories (if they exist) or if you installed the PFA Java code in a location other than the default path, you must update each ini file after running the install script for PFA. For more information, see "Updating the Java path" on page 76.

Installing PFA

Before you begin: Before installing PFA in your environment, you must initialize z/OS UNIX and install the following products on your system:

- IBM 31-bit SDK for z/OS Java Technology Edition, Version 6.0.0. For more information about Java, see Java Standard Edition website at www.ibm.com/systems/z/os/zos/tools/java/.

  Restriction: PFA does not support IBM 64-bit SDK for z/OS Java Technology Edition.
- IBM Health Checker for z/OS. You must be familiar with the set up for IBM Health Checker for z/OS. Most of the setup for PFA involves security definitions that are similar to the setup for any other started task and remote check. You must ensure that both PFA and IBM Health Checker for z/OS have access to the necessary resources including z/OS UNIX. For IBM Health Checker for z/OS details, see Setting up IBM Health Checker for z/OS in *IBM Health Checker for z/OS User's Guide*.

Guidelines:

1. The examples this procedure shows are for illustrative purposes only. Replace the example parameters with the correct specifications for your environment.
2. This documentation uses `/pfa` as the generic term for the home directory of the PFA user ID.
3. To use the PFA_COMMON_STORAGE_USAGE check, you must ensure your system is using the following DIAGxx parmlib member options: `VSM TRACK CSA(ON) SQA(ON)`.

For more information about using DIAGxx, see *z/OS MVS Initialization and Tuning Reference* and *z/OS MVS Initialization and Tuning Guide*. In addition, each check has specific guidelines. For example, the PFA_JES_SPOOL_USAGE check works only for JES2.

4. The z/OS File System (zFS) is a z/OS UNIX file system that contains files and directories that can be accessed with z/OS UNIX application programming interfaces (APIs) and supports access control lists (ACLs). In this documentation, all references to the z/OS UNIX file system assume that you are using zFS. For complete zFS details, see *z/OS Distributed File Service zFS Administration*.
5. If z/OS UNIX is shut down for any reason, restart PFA.

Steps for installing PFA

Use the following steps to set up PFA with RACF and z/OS UNIX:

1. Define additional DASD storage for PFA. The recommended **/var/pfa** file system is zFS. Requirements for the total space for the PFA file system for each LPAR is 300 cylinders primary; 50 cylinders secondary on a 3390 device.
2. Create a user ID to define the location in the z/OS UNIX file system that stores the PFA data and connects the PFA user ID to an existing or new RACF group.

   Guideline: This documentation uses *pfauser* as the generic term for the PFA user ID. Certain installation tasks require UID of 0.

   If you are using PFA in a sysplex that shares file systems for z/OS UNIX, use a unique directory for each LPAR so that the event data that PFA writes to the file system is stored separately for each system. For details, see “Installing PFA in a z/OS UNIX shared file system environment” on page 75.

   a. Create a new user ID to own the PFA. For example, *pfauser*. The PFA user ID must be unique; do not use the same user ID that is assigned to the IBM Health Checker for z/OS.

   b. Ensure */etc/PFA* has the same security settings as *pfauser* or *pfauser* owns */etc/PFA*.

   c. Define the PFA started task by creating a RACF profile for the *pfauser* with the following items:

      - OMVS segment with a UID parameter (for example, *omvs(uid(7))*)
      - Home directory (for example, *home(/pfa)*)
      - PROGRAM pathname of /bin/sh (for example, *program(/bin/sh)*)

   Examples:

   - This example shows how you can define and connect a new user to RACF. Ensure that you replace the parameters with the correct settings for your installation.

     ```
     ADDUSER pfauser
        OMVS(UID(7) HOME(/pfa) PROGRAM(/bin/sh)) PASSWORD(sys1)
     ADDGROUP OMVSGRP OMVS(GID(46))
     CONNECT pfauser GROUP(OMVSGRP)
     ```

   - This example shows how you can change the information in a user's RACF profile:

     ```
     altuser pfauser omvs(uid(7) shared home(/pfa) program(/bin/sh))
     ```

   - For information about Security Server RACF, see the ADDUSER and ADDGROUP sections in *z/OS Security Server RACF Command Language Reference*.

- For information about defining z/OS UNIX users to RACF, see the topic on Steps for defining z/OS UNIX users to RACF in *z/OS UNIX System Services Planning*.

3. Add the PFA task to the STARTED class table in RACF and refresh, if necessary. For example:

```
SETROPTS GENERIC(STARTED)
RDEFINE STARTED PFA.* STDATA(USER(pfauser) GROUP(OMVSGRP))
SETROPTS CLASSACT(STARTED)
SETROPTS RACLIST(STARTED)
```

   If you have already activated RACLIST in the STARTED class, the last statement is:

```
SETROPTS RACLIST(STARTED) REFRESH
```

   For more information, see the following information:
   - *z/OS UNIX System Services Planning* and *z/OS Security Server RACF Security Administrator's Guide*.
   - The RDEFINE and SETROPTS sections in *z/OS Security Server RACF Command Language Reference*.
4. Copy the sample PFA procedure, AIRPROC, from SYS1.SAMPLIB to the PFA member of SYS1.PROCLIB data set. If SMP/E does not write the executable code in the z/OS UNIX file system to `PARM='path=(/usr/lpp/bcp)'`, change the PARM value in AIRPROC to the path in which you store the executable code.
5. Run the install script AIRSHREP.sh, either from of the user ID that owns the PFA data and PFA started task (for example, /pfa) or create a batch job using the JCL file, AIRINJCL, provided in SYS1.SAMPLIB, with one of the following parameters:
   - **migrate**: Use the *migrate* parameter to preserve existing PFA data from the prior release. The *migrate* option is recommended.
   - **new**: Use the *new* parameter if you are installing PFA for the first time or if you want to delete everything from prior releases and start PFA with clean directories.

   To run the install script from your home directory:
   a. From OMVS command line, make the current directory the home directory for the PFA user: `cd /pfa`
   b. Using the appropriate parameter, run the install script using either `/usr/lpp/bcp/AIRSHREP.sh new` or `/usr/lpp/bcp/AIRSHREP.sh migrate`

   For example:

```
cd /pfa
/usr/lpp/bcp/AIRSHREP.sh migrate
```

   To run the install script in batch:
   a. Copy the sample batch job AIRINJCL from SYS1.SAMPLIB.
   b. Update the directory path in your copy of AIRINJCL with the home directory of the user ID that owns the PFA data and started task.
   c. Select the appropriate parameter for migration on `PARM=` JCL statement.
   d. Submit the JCL to run the install script.

   Update Java configuration

   PFA can use the single ini file for all checks in the `/etc/PFA` directory, which means you only have to update and maintain one ini file. If you prefer a specific check uses a different level of Java than what is specified in the

`/etc/PFA/ini` directory, provide an ini file in the check directory for the check. For example, create an ini file in the `pfa_directory/PFA_MESSAGE_ARRIVAL_RATE/` if you want to use a different level of Java for the PFA_MESSAGE_ARRIVAL_RATE check.

If the path to the JDK for your installation is not the same as the path in the ini file in `/etc/PFA/` and in the check directories (if they exist) or if you installed the PFA Java code in a location other than the default path, you must update each ini file after running the install script for PFA. For more information, see "Updating the Java path" on page 76.

Note: Read the topic on "Using and configuring supervised learning" on page 86 to decide whether you need to use the EXCLUDED_JOBS file.

6. Allow the appropriate people access to the PFA results in SDSF and the z/OS UNIX file system. (Both systems use standard security controls.)
7. Verify that common storage tracking (CSA Tracker) is active and the SMF exits (in SMFPRMxx.) are defined.
   - For information about activating and reviewing data provided by CSA Tracker, see the topic about "Using the common storage tracking function" in *z/OS MVS Initialization and Tuning Guide*.
   - For information about defining SMF exits, see the topic about SMFPRMxx in *z/OS MVS Initialization and Tuning Reference*.
8. Update the COMMNDxx parmlib member, and any system automation your installation has defined, with the PFA procedure to ensure that PFA restarts on IPL as shown here:
   ```
   START pfa_procname
   ```
   Make sure that you define the IBM Health Checker for z/OS procedure in COMMND*xx*. See Start IBM Health Checker for z/OS in *IBM Health Checker for z/OS User's Guide*.

   Important: When updating your COMMNDxx parmlib member, remember to update any system automation your installation uses to start and restart major system address spaces.
9. Update your WLM Service Class policy for PFA to be the same priority that your installation uses for monitoring products like RMF. For more information about defining service classes, see "Defining service classes and performance goals" in *z/OS MVS Planning: Workload Management*.
10. Increase the MAXFILEPROC settings in BPXPRMxx if your current settings are too low. The MAXFILEPROC sets the maximum number of file descriptors that a single process can have open concurrently, such as all open files, directories, sockets, and pipes. By limiting the number of open files that a process can have, you limit the amount of system resources a single process can use at one time. You can also dynamically change the MAXFILEPROC setting using the `SETOMVS` command. For example:
    ```
    SETOMVS PID=123,MAXFILEPROC=value
    ```
    References:
    - See the topic on Steps for changing the process limits for an active process in *z/OS UNIX System Services Planning*.
    - To determine if the MAXFILEPROC value is set too low, the IBM Health Checker for z/OS provides a check, USS_MAXSOCKETS_MAXFILEPROC. For details, see the check USS_MAXSOCKETS_MAXFILEPROC in *IBM Health Checker for z/OS User's Guide*.
11. Customize your system settings for PFA:

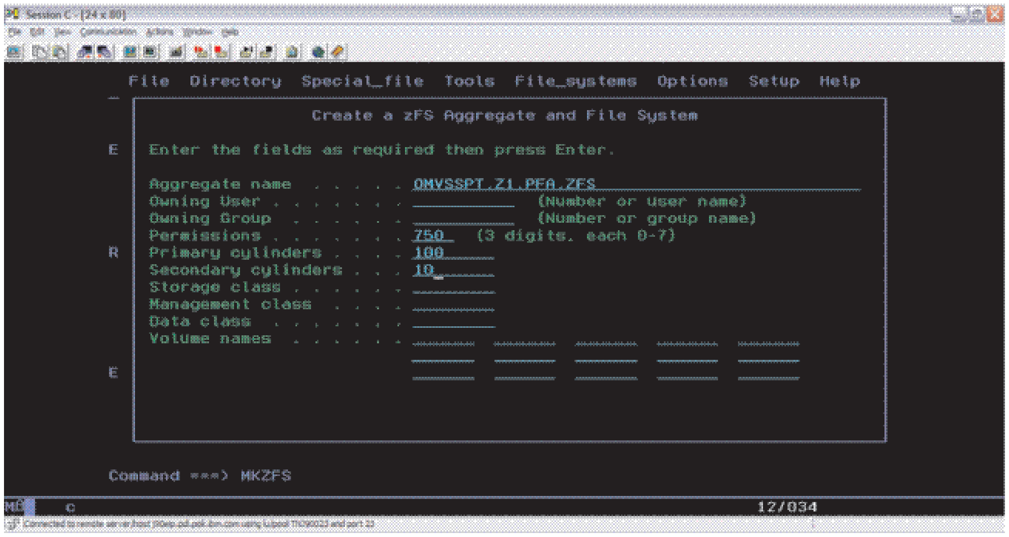
a. Update your system automation to look for PFA exception messages. For complete details, see the topic about Approaches to automation with IBM Health Checker for z/OS in *IBM Health Checker for z/OS User's Guide*.
b. Follow the guidelines for correcting problems by reviewing the check-specific best practice.
c. After the checks have been running for a while, you might decide that the default parameters are not appropriate in your environment. You can customize the PFA checks using the check-specific parameters. For details, see the check-specific parameters.

Installing PFA in a z/OS UNIX shared file system environment

In this procedure you create a z/OS UNIX file system that is shared among members of the sysplex with directories that are local to the LPAR. This procedure uses zSeries File System (zFS) because it is the strategic file system. This procedure enables you to define one started task and user ID that permits PFA to write files to a system-unique directory.

Requirement: The user ID that performs this installation must have a UID of 0. In some installations, this might mean your z/OS UNIX Administrator.

1. Define the file systems as one for each LPAR. You can use the TSO ISHELL (ISPF shell) panel to define and format the zFS file system. In this example, the Integration Test team used `OMVSSPT.Z1.PFA.ZFS` as the example system.



```
     File  Directory  Special_file  Tools  File_systems  Options  Setup  Help
    _ ┌──────────────────────────────────────────────────────────────────────┐
      │               Create a zFS Aggregate and File System                 │
      │                                                                      │
    E │ Enter the fields as required then press Enter.                       │
      │                                                                      │
      │ Aggregate name  . . . . . OMVSSPT.Z1.PFA.ZFS                         │
      │ Owning User . . . . . . . __________  (Number or user name)          │
      │ Owning Group  . . . . . . __________  (Number or group name)         │
      │ Permissions . . . . . . . 750   (3 digits, each 0-7)                 │
    R │ Primary cylinders . . . . 100_____                                   │
      │ Secondary cylinders . . . 10______                                   │
      │ Storage class . . . . . . ________                                   │
      │ Management class  . . . . ________                                   │
      │ Data class  . . . . . . . ________                                   │
      │ Volume names  . . . . . . ______  ______  ______  ______  ______     │
      │                           ______  ______  ______  ______  ______     │
    E │                           ______  ______  ______  ______  ______     │
      │                                                                      │
      └──────────────────────────────────────────────────────────────────────┘

    Command ===> MKZFS
```

Figure 15. Defining the file system

For more information about using ISHELL, see the topic "Invoking the ISPF shell" in *z/OS UNIX System Services User's Guide*.

2. After defining the file systems for each LPAR, define a symbolic link (also called a symlink) to the sysplex root. From the root directory, enter the following command (the user issuing cd and ln commands require a UID of 0):

```
cd /
ln -s \$SYSNAME/pfa pfa
```

This results in a home directory of /*systemname*/`pfa`.

Guideline: The home directory of the user ID that owns the started task is where PFA expects the directories and data created by the install process to reside. PFA writes the historical data it needs to make predictions to the directories created by the installation process.

3. Create the PFA directory in each of the system directories by entering the following commands for each of your system names:

   For example, the command for system Z1 is: `mkdir /Z1/pfa`:

   ```
   mkdir /Z1/pfa
   mkdir /Z2/pfa
   mkdir /Z3/pfa
   mkdir /Z4/pfa
   ```

4. Create the new file systems (one for each system) and mount them at the appropriate system mount point. For example, for `OMVSSPT.Z1.PFA.ZFS`, the mount point is `z1/pfa`:

   ```
   OMVSSPT.Z1.PFA.ZFS
   OMVSSPT.Z2.PFA.ZFS
   OMVSSPT.Z3.PFA.ZFS
   OMVSSPT.Z4.PFA.ZFS
   ```

   Here is an example of the file system attributes:
   See the topic “Managing the z/OS UNIX file system” in *z/OS UNIX System*

File System Attributes

```
         File System Attributes

File system name:
OMVSSPT.Z1.PFA.ZFS
Aggregate name:
OMVSSPT.Z1.PFA.ZFS
Mount point:
/Z1/pfa
                                                 More:     +

Status . . . . . . . . : Available
File system type . . . : ZFS
Mount mode . . . . . . : R/W
Device number  . . . . : 28412
Type number  . . . . . : 1
DD name  . . . . . . . :
Block size . . . . . . : 1024
Total blocks . . . . . : 720000
Available blocks . . . : 561716
```

   Services Planning.

5. Place an entry in the SYS1.PARMLIB(BPXPRMXX) member to mount the new file systems during IPL. (If you do not want to wait until the next IPL, you can manually mount these file systems.) Use the UNMOUNT attribute on the BPXPRMXX parmlib member to unmount the file system when OMVS or the LPAR is taken down. The file system mount point is `/sysname/pfa` for example:

   ```
   SYS1.PARMLIB(BPXPRM00)
     MOUNT FILESYSTEM('OMVSSPT.&SYSNAME..PFA.ZFS') TYPE(ZFS)
       MODE(RDWR) MOUNTPOINT('/&SYSNAME./pfa') UNMOUNT
   ```

Updating the Java path

You must update the ini file in the `/etc/PFA/` directory and ini files in check directories for PFA processing to locate the Java code on your system. For more information about how PFA determines which ini file to use, see “Migration considerations for PFA” on page 69.

The `JAVAPATH=` line must be set to the SMP/E installation path for the Java code for PFA. If you installed the Java code for PFA elsewhere, you must change this line to the path where the PFA for Java is installed. The default is `/usr/lpp/bcp`. For example:

```
JAVAPATH= /usr/lpp/bcp
```

Update the PATH= and LIBPATH= statements in the ini file for each check to point to the executable code needed for JNI calls. The following example 77 shows the typical Java 6.0 path for PFA:

```
PATH= /usr/lpp/java/J6.0/lib/s390/classic:/usr/lpp/java/J6.0/lib/s390
LIBPATH= /usr/lpp/java/J6.0/lib/s390:/usr/lpp/java/J6.0/lib/s390/classic:/lib:/usr/lib:
```

The update to the `PATH=` and `LIBPATH=` lines is dependent of the level of Java that your installation uses. In Java 6.0, these lines typically point to `$JAVA_HOME/lib/s390/classic`. The value for the $JAVA_HOME variable is typically as follows for Java 6.0:

```
/usr/lpp/java/J6.0 (/usr/lpp/java/J6.0/lib/s390/classic)
```

Chapter 8. Managing PFA checks

You can use the MODIFY system command to display and update Predictive Failure Analysis (PFA) checks as well as the IBM Health Checker for z/OS commands. To help you understand how to manage these differences, this section contains the following topics:

- "Understanding how to modify PFA checks" describes the unique differences PFA checks have from traditional checks.
- "MODIFY PFA, DISPLAY" on page 81 describes the `MODIFY` *`pfa`*`,DISPLAY` command in detail.
- "MODIFY PFA, UPDATE" on page 84 describes the `MODIFY` *`pfa`*`,UPDATE` command in detail.
- "Using and configuring supervised learning" on page 86 describes how to use the EXCLUDED_JOBS file to help PFA avoid false positives.

For the IBM Health Checker for z/OS commands, see *IBM Health Checker for z/OS User's Guide*.

You can make installation updates to PFA checks that persist across check refreshes and restart IBM Health Checker for z/OS by activating IBM Health Checker for z/OS policies. You might do this if some check default values are not suitable for your environment or configuration. For complete details, see the topic on Creating IBM Health Checker for z/OS policies in *IBM Health Checker for z/OS User's Guide*.

Restriction: The IBM Health Checker for z/OS debug commands are not the same debug parameter that PFA checks use. For details, see "Understanding how to modify PFA checks."

Understanding how to modify PFA checks

PFA checks work differently than traditional checks. Although you can modify PFA checks using the IBM Health Checker for z/OS commands and policies (as described in *IBM Health Checker for z/OS User's Guide*), modifications to the PFA checks are unique in the following ways:

- **How to quiesce a PFA check**:

  Use the MODIFY command to quiesce a PFA check. If the check is to permanently quiesce until PFA restarts, use the MODIFY command to delete the check. Because PFA checks are remote health checks, you must restart PFA to add a previously deleted check. If the check is to be quiesced now, but restarted later while PFA is still running, use the MODIFY command to deactivate the check. Deactivating a check stops the comparisons from occurring, but to stop collection and modeling you must set the COLLECTINACTIVE parameter to 0. To stop collections, modeling, and comparisons for an individual check so that the check can be reactivated without restarting PFA, do the following:

  1. First , stop PFA from collecting and modeling data by setting the COLLECTINACTIVE parameter to zero. For example:
     ```
     f hzsproc,update,check(ibmpfa,pfa_logrec_arrival_rate),
     parm('collectinactive(0)')
     ```
  2. Next, deactivate the check in IBM Health Checker for z/OS. For example:
     ```
     f hzsproc,deactivate,check(ibmpfa,pfa_logrec_arrival_rate)
     ```

- **Use the MODIFY *pfa* DISPLAY command to display the check-specific parameters:**

  You can use the `MODIFY pfa,DISPLAY` command to display the cumulative set of modified, check-specific parameters that are currently in use by a PFA check. See "MODIFY PFA, DISPLAY" on page 81 for more information.
- **Use the MODIFY *pfa* UPDATE command to read specific PFA configuration files:**

  You can use the `MODIFY` *`pfa`*`,UPDATE` command to read certain configuration files that a check supports. See "MODIFY PFA, UPDATE" on page 84.
- **Using the MODIFY *hzsproc* command to modify individual PFA check parameters:**

  You can use the MODIFY *hzsproc* command to modify individual parameters of PFA checks. When specifying the PARM parameter on `f hzsproc,update` for PFA checks, you do not have to specify all check-specific parameters. The parameters that are not specified are not changed. If the parameters were not previously modified, the values remain the default values. If the parameters were previously modified, the previously specified values remain.

  The following example of the MODIFY *hzsproc* command sets the `debug` parameter for PFA to ON.

  ```
  f hzsproc,update,check(IBMPFA,PFA_COMMON_STORAGE_USAGE),parm('debug(1)')
  ```

  The rest of the check-specific parameters retain their default values as follows:

  ```
  CHECK SPECIFIC PARAMETERS:
         COLLECTINT                 : 15
         MODELINT                   : 360
         COLLECTINACTIVE            : 1=ON
         DEBUG                      : 1=ON
         THRESHOLD                  : 2
  ```

  If the check is displayed using IBM Health Checker for z/OS interfaces, the user supplied parameters are displayed as either:

  ```
  USER SUPPLIED PARAMETERS:  debug(1)
  ```

  or

  ```
  CHECK PARM: debug(1)
  ```

  Then, when the following command is entered:

  ```
  f hzsproc,update,check(IBMPFA,PFA_COMMON_STORAGE_USAGE),parm('collectint(10) modelint(60)')
  ```

  the values for the check change to:

  ```
  CHECK SPECIFIC PARAMETERS:
         COLLECTINT                 : 10
         MODELINT                   : 60
         COLLECTINACTIVE            : 1=ON
         DEBUG                      : 1=ON
         THRESHOLD                  : 5
  ```

  If the check is displayed again using IBM Health Checker for z/OS interfaces, the user-supplied parameters that are displayed are the last ones specified on the MODIFY command as follows:

  ```
  USER SUPPLIED PARAMETERS:  collectint(10) modelint(60)
  ```

  or

  ```
  CHECK PARM: collectint(10) modelint(60)
  ```

This is not the cumulative list of parameters currently in use by the check. Therefore, to display the current parameter values being used by the check, use the `f pfa,display,checks,detail` command as listed in "MODIFY PFA, DISPLAY."

- **The debug parameter is a PFA check-specific parameter.**

  The debug parameter is a check-specific parameter and not the same debug parameter as the one in IBM Health Checker for z/OS. The debug parameter within IBM Health Checker for z/OS applies only to the phase of performing the check when the interval is reached and not to all other phases, such as data collection and modeling that are done internally within the PFA checks. Therefore, the debug parameter within IBM Health Checker for z/OS has no meaning to PFA and is ignored.

  To set the debug parameter for PFA checks, specify it as a parm as in the following example:

  ```
  f hzsproc,update,check(IBMPFA,PFA_CHECK_NAME),parm('debug(1)')
  ```

  Each PFA check contains more information about its check-specific parameter.

MODIFY PFA, DISPLAY

Purpose

`MODIFY` *`pfa`*`,DISPLAY` (`f pfa,display`) issues messages with information specified as different options (listed in "Parameters").

When displaying PFA checks using the IBM Health Checker for z/OS commands, the user-supplied parameters list contains only the parameters that were specified on the last update command not the cumulative set of modified parameters. Therefore, you must use the `MODIFY pfa,DISPLAY` command to display the check-specific parameters that are currently used by a PFA check. See "Understanding how to modify PFA checks" on page 79 for more information.

Format

```
DISPLAY
 {
 [CHECKS [,filters] [,SUMMARY] | ,DETAIL]]
 |
 [filters [,SUMMARY] | ,DETAIL]]
 |
 [STATUS]
 }
```

Parameters

CHECKS
CHECKS displays information about PFA checks.

filters Filters specify which check or checks you want to take an action against. You can specify the wildcard character * for filters in the last position of the filter. An asterisk (*) represents any string having a length of zero or more characters.

Filters must be specified in one of the following formats:

```
CHECKS,CHECK=(check_name)
or
CHECK=(check_name)
```

`check_name` specifies the 1- through 32-character check name.

SUMMARY
PFA issues message AIR013I with summary information about the specified checks. See "PFA DISPLAY examples" on page 83. For each check matching the specified filter, the following information is returned:

- Check name
- Indicator of whether the check is eligible to run at the next interval (ACTIVE(ENABLED)) in IBM Health Checker for z/OS
- The last successful collection time
- The last successful model time

SUMMARY is the default value; it does not need to be specified in the command.

DETAIL
PFA issues message AIR018I with detailed information about the specified checks. See "PFA DISPLAY examples" on page 83. For each check matching the specified filter, the following information is returned:

- Check name
- Indicator of whether the check is eligible to run at the next interval (ACTIVE(ENABLED)) in IBM Health Checker for z/OS
- Total number of collections attempted
- Total number of successful collections
- The last time the collection ran
- The last successful collection time
- The next collection time
- Total number of models attempted
- Total number of successful models
- The last time the model ran
- The last successful model time
- The next model time
- The current settings of the check-specific parameters for this check:
  - The collection interval in minutes
  - The model interval in minutes
  - Indicator of whether to collect data and model data even if the check is not eligible to run (ACTIVE(ENABLED)) in IBM Health Checker for z/OS
  - Indicator if the check is generating additional diagnostic information
  - Any other parameters that are supported for this check
- If this check supports excluded jobs, the list of jobs currently being excluded for this check.

STATUS
PFA issues message AIR017I with general status information for PFA. The following information is returned:

- The number of checks registered to PFA
- The number of PFA checks eligible to run (ACTIVE(ENABLED)) in IBM Health Checker for z/OS
- The number of collections currently queued
- The number of models currently queued
- The number of JVM terminations that have occurred since PFA started

Flags

None.

Error conditions

None.

Version

All releases.

PFA DISPLAY examples

Example of DISPLAY STATUS message output

```
IR017I 10.31.32 PFA STATUS

NUMBER OF CHECKS REGISTERED    : 5
NUMBER OF CHECKS ACTIVE        : 5
COUNT OF COLLECT QUEUE ELEMENTS: 0
COUNT OF MODEL QUEUE ELEMENTS  : 0
COUNT OF JVM TERMINATIONS      : 0
```

The DISPLAY STATUS message output displays in response to the following commands:

- `f pfa,display`
- `f pfa,display,status`

Example of DISPLAY SUMMARY message output

```
AIR013I 10.09.14 PFA CHECK SUMMARY

                                  LAST SUCCESSFUL     LAST SUCCESSFUL
CHECK NAME                ACTIVE   COLLECT TIME          MODEL TIME
PFA_COMMON_STORAGE_USAGE  YES    04/05/2010 10.01    04/05/2010 08.16
PFA_ENQUEUE_REQUEST_RATE  YES    04/05/2012 09.15    04/05/2012 06.32
PFA_LOGREC_ARRIVAL_RATE   YES    04/05/2010 09.15    04/05/2010 06.32
PFA_MESSAGE_ARRIVAL_RATE  YES    04/05/2010 08.15    04/05/2010 05.32
PFA_SMF_ARRIVAL_RATE      YES    04/05/2010 08.05    04/05/2010 05.02
```

The DISPLAY SUMMARY message output displays in response to the following commands:

- `f pfa,display,checks` - results in all checks
- `f pfa,display,checks,check=(check_name)` - results in one check
- `f pfa,display,checks,check=(PFA_*)` - can result in > 1 check
- `f pfa,display,checks,check=(*),summary` - results in all checks
- `f pfa,display,check(check_name)` - results in one check
- `f pfa,display,check(PFA_*)` - can result in > 1 check
- `f pfa,display,check(*),summary` - results in all checks

Example of DISPLAY DETAIL message output

```
AIR018I 02.22.54 PFA CHECK DETAIL

CHECK NAME:  PFA_MESSAGE_ARRIVAL_RATE
    ACTIVE                          : YES
    TOTAL COLLECTION COUNT          : 5
    SUCCESSFUL COLLECTION COUNT     : 5
    LAST COLLECTION TIME            : 04/05/2010 10.18.22
    LAST SUCCESSFUL COLLECTION TIME : 04/05/2010 10.18.22
    NEXT COLLECTION TIME            : 04/05/2010 10.33.22
    TOTAL MODEL COUNT               : 1
    SUCCESSFUL MODEL COUNT          : 1
    LAST MODEL TIME                 : 04/05/2010 10.18.24
    LAST SUCCESSFUL MODEL TIME      : 04/05/2010 10.18.24
    NEXT MODEL TIME                 : 04/05/2010 11.18.24
    CHECK SPECIFIC PARAMETERS:
       COLLECTINT                   : 15
       MODELINT                     : 60
       COLLECTINACTIVE              : 1=YES
       DEBUG                        : 0=NO
       STDDEV                       : 10
       TRACKEDNMIN                   : 0
       EXCEPTIONMIN                  : 1
EXCLUDED JOBS:
      NAME     SYSTEM   DATE ADDED       REASON ADDED
      JES2     *        03/11/2010 15:15 Excluded JES* jobs on ALL.
```

The DISPLAY DETAIL message output displays in response to the following commands:

- `f pfa,display,checks,detail` - results in all checks
- `f pfa,display,checks,check=(name),detail` - results in one check
- `f pfa,display,checks,check=(check_na*),detail` - can result in > 1 check
- `f pfa,display,checks,check=(*),detail` - results in all checks
- `f pfa,display,check(check_name),detail` - results in one check
- `f pfa,display,check(check_na*),detail` - can result in > 1 check
- `f pfa,display,check(*),detail` - results in all checks

MODIFY PFA, UPDATE

Purpose

Use the `MODIFY` *pfa*`,UPDATE` (`f pfa,update`) command to read configuration files that an individual check supports and store the values for future PFA processing.

You must use the `MODIFY pfa,update` command before changes made to check-specific configuration files are used by PFA processing.

Format

```
UPDATE
{
[CHECKS [,filters] [,EXCLUDED_JOBS]]
|
[filters [,EXCLUDED_JOBS]]
}
```

Parameters

CHECKS
CHECKS reads the configuration file requested for every check that supports that configuration file. This parameter is optional.

filters Filters specify which check or checks you want to take an action against. This parameter is optional. You can specify the wildcard character * for filters in the last position of the filter. An asterisk (*) represents any string having a length of zero or more characters.

Filters must be specified in one of the following formats:

```
CHECKS,CHECK=(check_name)
or
CHECK=(check_name)
```

`check_name` specifies the 1- through 32-character check name.

EXCLUDED_JOBS
PFA reads the EXCLUDED_JOBS file and updates its internal configuration to use the values in this file. **EXCLUDED_JOBS** is the default value; it does not need to be specified in the command. You can optionally add exclusions. For details, see "Using and configuring supervised learning" on page 86.

Defaults

- CHECKS – all checks are updated.
- EXCLUDED_JOBS – is the default value.

Processing

For each check specified that supports EXCLUDED_JOBS, the EXCLUDED_JOBS file (if it exists for the check) is read from the checks' /config directory and stored in memory for the PFA address space to use in its processing.

Usage

Use this command when PFA configuration needs to be updated, but the update is not a part of the IBM Health Checker for z/OS configuration and parameters.

Restrictions

None.

Authorization

The user running the MODIFY PFA,UPDATE command must be authorized to write to the EXCLUDED_JOBS file for the check.

Flags

None.

Error conditions

None.

Messages

- AIR024I
- AIR025I
- AIR027I
- AIR028I

- AIR029I
- AIR030I
- AIR031I

For the complete message text, see the topic on AIR messages in *z/OS MVS System Messages, Vol 1 (ABA-AOM)*.

Version

All releases beginning with z/OS V1R12.

Examples

- f pfa,update
  - Reads the EXCLUDED_JOBS file (if it exists) for every check that supports it and stores the new values to use for future processing.
- f pfa,update,checks,EXCLUDED_JOBS
  - Reads the EXCLUDED_JOBS file (if it exists) for every check that supports it and stores the new values to use for future processing.
- f pfa,update,check(pfa_m*) or f pfa,update,check(pfa_m*),EXCLUDED_JOBS
  - Reads the EXCLUDED_JOBS file (if it exists) for every check that starts with “pfa_m” and that supports it and stores the new values to use for future processing.

Context

- Load module: AIRAMPVT
- Entry Point: AIRA1INI

Using and configuring supervised learning

The “supervised” learning service can help you avoid false positives by excluding certain data that PFA uses when making predictions of future behavior. To minimize the impact to check performance, only use EXCLUDED_JOBS for the conditions that cause you the most inconvenience. Instead, use other tuning parameters for the check such as `STDDEV`. A sample EXCLUDED_JOBS file ships in the `/usr/lpp/bcp/samples/PFA` directory. It is named EXCLUDED_JOBS and includes an example comment line. You can modify the file using the OEDIT command, and then use the `f pfa,update` command to have PFA read in the contents of the file and start to use it in during processing. Supervised learning applies to the following checks:

- “PFA_ENQUEUE_REQUEST_RATE” on page 97
- “PFA_LOGREC_ARRIVAL_RATE” on page 113
- “PFA_MESSAGE_ARRIVAL_RATE” on page 120
- “PFA_JES_SPOOL_USAGE” on page 107: For example, the JES spool usage check shows a certain job to have high spool usage because the job is frequently restarted. You want to exclude this job from check processing.

After PFA is installed, you can optionally use the following instructions to use the supervised learning support:

1. Create the EXCLUDED_JOBS file in the `/config` directory for each check for which you want jobs excluded and that supports supervised learning. You can copy the sample from `/usr/lpp/bcp/samples/PFA/EXCLUDED_JOBS`.
2. Add the jobs you want to be excluded.

3. If PFA has already been started, run the `f pfa,update` command to cause the EXCLUDED_JOBS file to be read for all checks.
4. If you have modified the STDDEV parameter for a check because it was receiving too many exceptions and you are now excluding the jobs that caused the exception, consider reducing the STDDEV parameter for those checks.

   Note: There is an excluded job for the message arrival rate check. The PFA_MESSAGE_ARRIVAL_RATE: CONSOLE exclusion is hardcoded. Data for the job is not included in the check-specific processing.

The checks that support supervised learning use a z/OS UNIX file with the name EXCLUDED_JOBS in the `/config` directory for the check. This directory is read at check initialization (when PFA starts) and when the `modify PFA,update` command is issued for the check. For example, find the list of excluded jobs for the PFA_MESSAGE_ARRIVAL_RATE in the `/pfa_directory/ PFA_MESSAGE_ARRIVAL_RATE/config/EXCLUDED_JOBS file`.

If you are using PFA in a z/OS UNIX shared file system environment, each LPAR in the sysplex has a local `pfa_directory` in which a directory exists for each check. For each LPAR in the sysplex that needs to exclude jobs, the `EXCLUDED_JOBS` file must exist in the `/config` directory of the checks for which jobs should be excluded. To generically specify the system on which to exclude the job, use a system name containing a wildcard in the system name field in the EXCLUDED_JOBS file. You can then copy the file from the `/config` directory for a check on one LPAR to the `/config` directory for the same check on another LPAR.

The PFA EXCLUDED_JOBS file format is a simple, comma-separated value format as shown in Table 5. The row ends when a new line character is reached. Each field ends when a comma is reached although not all data is stored in memory.

Table 5. PFA EXCLUDED_JOBS file format

Field	Length	Format of field
Job name	8 characters maximum	• The name of the job to exclude and is required. • Both the job name and the system name must match at run time in order for the job to be excluded on this system. • Wildcard characters are allowed. – The * character is allowed in any position and denotes one or more characters. – The ? character is allowed in any position and denotes one character.
System name	8 characters maximum	• This field identifies the system to which this excluded job applies. • This field is required. • Both the job name and the system name must match at run time in order for the job to be excluded on this system. • If this excluded job applies to all LPARs in the sysplex, specify *. • If the exclusion applies to multiple systems, but not to all systems in the sysplex, use wildcard characters: – The * character is allowed in any position and denotes one or more characters. – The ? character is allowed in any position and denotes one character.

Table 5. PFA EXCLUDED_JOBS file format (continued)

Field	Length	Format of field
Date and/or time of adding	A maximum of 16 characters are stored in memory	This field is a character string that provides usability so you can see when you added the exclusion. There is no specific format and is not used by the code other than to display it when you request it. • The length is 16 characters to fit the date and time in a numerical format such as: `06/10/2009 03:45` • If you do not require this field, you can skip it by only specifying the comma delimiter. • If the field is longer than 16 characters, only the first 16 characters are stored in memory. No error message is issued for this situation because the text is still available in the file.
Reason for adding	A maximum of 35 characters are stored in memory	This field is merely a character string and provided for usability so you can see the reason for excluding the job or address space. There is no specific format and is not used by the code other than to display it when you request it. • If this field is not required, it can be omitted. • If it is longer than 35 characters, only the first 35 characters are stored in memory. No error message is issued for this situation because the text is still available in the file. • The `f pfa,display` output displays the first 29 characters. All 35 characters are written to the config log. You can type more characters and they remain in the file, but only the first 35 are in memory.

Example: Valid input rows

```
JES2,*,06/10/2009 03:45:35,Exclude JES2 on all systems
TEST*,SYS1,Skip all of my test jobs on SYS1
DUMPSRV,*
CONSOLE,SYS1,06/10/2009 03:45:35
CONSOLE,SYS2,06/10/2009 03:45:35
```

Example: Input rows that are not valid:

```
JES234567,*,,This name was too long.
,,06/10/2009,Name and system are required
```

Note:

1. When processing encounters a row that is not valid, it disregards the row and issues an error message.
2. The code checks that the maximum number of characters for the job name and the system name are 8.
3. When processing encounters a duplicate job name and system name combination, it uses the first occurrence, ignores the subsequent occurrence, and issues a message.
4. The EXCLUDED_JOBS file supports comment lines. If a line starts with `/*` or `#`, processing ignores the line. No ending comment is necessary however, the comment lines as well as all lines must end in a new line character ‘15’x.

Chapter 9. Predictive Failure Analysis checks

Predictive Failure Analysis (PFA) provides the following remote checks:

- "PFA_COMMON_STORAGE_USAGE"
- "PFA_ENQUEUE_REQUEST_RATE" on page 97
- "PFA_JES_SPOOL_USAGE" on page 107
- "PFA_LOGREC_ARRIVAL_RATE" on page 113
- "PFA_MESSAGE_ARRIVAL_RATE" on page 120
- "PFA_SMF_ARRIVAL_RATE" on page 136

PFA_COMMON_STORAGE_USAGE

Description:

The check is looking to see if there is a potential for storage to be exhausted in the upcoming predictive failure analysis (PFA) model interval. PFA analyzes the following storage locations:

- common storage area (CSA)
- system queue area (SQA)
- extended common storage area (ECSA)
- extended system queue area (ESQA)
- CSA + SQA
- ECSA + ESQA

The PFA_COMMON_STORAGE_USAGE check detects three classes of common storage exhaustion:

- Spike
- Leak
- Creep

If PFA detects that there is a potential for the exhaustion of common storage, PFA issues exception message AIRH101E and provides a list of suspect tasks in the report. During the analysis, this check writes the common storage usage data at intervals to a z/OS UNIX System Services file in comma-separated value (.csv) format. The check identifies a list of users of common storage that might contribute to exhausting common storage. If deeper analysis is necessary, PFA also provides files that contain additional diagnostic information that you can examine. See "Best practice:."

PFA also issues the following informational messages:

- AIRH102I
- AIRH103I
- AIRH132I

Reason for check:

If the system runs out of common storage, jobs and started tasks experience abends.

Best practice:

The best practice is to predict common storage problems before they occur, determine the cause of the problem, and take the appropriate action.

When IBM Health Checker for z/OS issues exception message AIRH101E, PFA has predicted that the amount of storage allocated to the common storage area is in jeopardy of being exhausted. Use the following steps to determine the appropriate action:

1. Examine the Common Storage Usage Prediction Report issued with the exception message. This report contains the total current usage and predictions for each of the six storage locations: CSA, SQA, ECSA, ESQA, CSA+SQA, and ECSA+ESQA. It also contains up to ten “users” each of CSA, SQA, ECSA, and ESQA whose usage has changed the most in the last model interval. The cause of the problem is most likely within this list of users. See “Output:” on page 92 for the example report.
2. If the cause of the problem is not obvious from the common storage usage report, you can obtain additional information in the csadata and the csaAlldata files, or from other checks that list the top users of storage such as the checks owned by IBMVSM (VSM_CSA_THRESHOLD and VSM_SQA_THRESHOLD). The files are text files in comma-separated value (.csv) format and contain the historical data on the usage for each interval. You can export the files into any spreadsheet-type program.
3. Determine which type of common storage problem is occurring by examining the symptoms, and then correct the behavior:
   - **Spike:** A piece of code uses more and more of the common storage area with usage growing linearly or exponentially over time. If the problem is caused by a spike, **the csaAlldata file** contains one or more users that are in the last few intervals and that consume a significant and measurable amount of common storage.

     Determine if the job causing the spike can be stopped, canceled, or slowed without affecting the overall system behavior.
   - **Leak:** A piece of code returns some but not all of the storage, which results in more usage of the common storage area over time. If the problem is caused by a leak, look for the contributor that is on the list multiple times, but not in every interval.

     Determine if the job causing the leak can be stopped, canceled, or slowed down without affecting the overall system behavior.
   - **Creep:** The common storage area usage grows slowly reflecting the overall system usage, which means there is no individual user of CSA responsible for the storage exhaustion. If there is no job or address space that appears to be using an excessive or unusual amount of common storage, the amount of work being done by the LPAR is probably causing the usage of common storage to creep.

     Determine if the amount of work being sent to this LPAR can be reduced.

   Note: Because of the random variation in common storage usage that typically occurs and the PFA check collects and models data at defined intervals, PFA is unable to detect all leaks, spikes, and creeps.
   - PFA is sometimes unable to detect a leak or creep that is less than 750 bytes per second.
   - PFA cannot detect rapid growth that occurs on a machine time frame such as within a collection interval.
   - PFA cannot detect common storage exhaustion caused by fragmentation.

z/OS releases the check applies to:
z/OS V1R10 and later.

Type of check:
Remote

Restrictions
Ensure your system is using the following DIAGxx parmlib member options: `VSM TRACK CSA(ON) SQA(ON)`

Parameters accepted:
Yes, as follows:

Table 6. PFA_COMMON_STORAGE_USAGE check parameters

Parameter name	Default value	Minimum Value	Maximum Value	Description
collectint	15 minutes	1	360	This parameter determines the time (in minutes) to run the data collector that determines the amount of common storage being used. The default is 15 minutes (15).
modelint	720 minutes	4	1440	This parameter determines how often (in minutes) you want the system to analyze the data and construct a new common storage usage model or prediction. Note that, even when you set a value larger than 360, PFA performs the first model at 360 minutes (6 hours). By default, PFA analyzes the data and constructs a new model every 720 minutes (12 hours). The model interval must be at least four times larger than the collection interval. If necessary modeling occurs more frequently.
threshold	2 percent	1	100	The percentage of the capacity of each area predicted to produce the capacity value to use in comparisons. The threshold can be used to reduce false positive comparisons. Setting the threshold too high might cause exhaustion problems to be undetected. The default is 2 percent (2).
collectinactive	1 (on)	0 (off)	1 (on)	Defines whether data is collected and modeled even if the check is not eligible to run (is not ACTIVE(ENABLED)) in IBM Health Checker for z/OS.
debug	0 (off)	0 (off)	1 (on)	This parameter (an integer of 0 or 1) is used at the direction of IBM service to generate additional diagnosic information for the IBM Support Center. This debug parameter is used in place of the IBM Health Checker for z/OS policy. The default is off (0).

To determine the status of the common storage usage check, issue `f pfa,display,check(pfa_common_storage_usage),detail`. See for the complete command example. The following is an example of the output written to message AIR018I in SDSF user log (ULOG):

```
F PFA,DISPLAY,CHECK(PFA_COMMON_STORAGE_USAGE),DETAIL
 AIR018I 16:20:21 PFA CHECK DETAIL
 CHECK NAME: PFA_COMMON_STORAGE_USAGE
     ACTIVE                          : YES
     TOTAL COLLECTION COUNT          : 5
     SUCCESSFUL COLLECTION COUNT     : 5
     LAST COLLECTION TIME            : 09/01/2008 10:18:22
     LAST SUCCESSFUL COLLECTION TIME : 09/01/2008 10:18:22
     NEXT COLLECTION TIME            : 09/01/2008 10:33:22
     TOTAL MODEL COUNT               : 1
     SUCCESSFUL MODEL COUNT          : 1
     LAST MODEL TIME                 : 09/01/2008 10:18:24
     LAST SUCCESSFUL MODEL TIME      : 09/01/2008 10:18:24
     NEXT MODEL TIME                 : 09/01/2008 22:18:24
```

```
CHECK SPECIFIC PARAMETERS:
    COLLECTINT                    : 15
    MODELINT                      : 720
    COLLECTINACTIVE               : 1=ON
    DEBUG                         : 0=OFF
    THRESHOLD                     : 2
```

User override of IBM values:

The following example shows keywords you can use to override check values either on a POLICY statement in the HZSPRMxx parmlib member or on a MODIFY command. See Chapter 8, “Managing PFA checks,” on page 79. You can copy and modify this statement to override the check defaults:

```
UPDATE CHECK(IBMPFA,PFA_COMMON_STORAGE_USAGE)
           ACTIVE
           SEVERITY(MEDIUM)
           INTERVAL(00:01)
      PARMS=('COLLECTINT(15)','MODELINT(720)','THRESHOLD(2)',
      'COLLECTINACTIVE(1)','DEBUG(0)')
           DATE(20071101)
      REASON('Common storage usage is nearing the user defined threshold.')
```

Verbose support:

The check provides additional details in verbose mode. You can put a check into verbose mode either using the UPDATE,filters,VERBOSE=ON parameters on the MODIFY command or on a POLICY statement on an HZSPRMxx parmlib member.

Debug support:

The DEBUG parameter in IBM Health Checker for z/OS is ignored by this check. Rather, the debug parameter is a PFA check specific parameter. The IBM Health Checker for z/OS debug commands are not the same debug parameter that PFA checks use. For details, see “Understanding how to modify PFA checks” on page 79.

Reference:

For more information about PFA, see the topic on “Overview of Predictive Failure Analysis” on page 65.

Messages:

This check issues the following exception messages:

- AIRH101E

For additional message information, see the topics:

- AIRH messages in *z/OS MVS System Messages, Vol 1 (ABA-AOM)*.
- AIR messages in *z/OS MVS System Messages, Vol 1 (ABA-AOM)*.

SECLABEL recommended for MLS users:

SYSLOW

Output:

The common storage usage output report:

```
Common Storage Usage Prediction Report

Last successful model time     :  07/09/2009 11:08:44
Next model time                :  07/09/2009 23:12:44
Model interval                 :  720
Last successful collection time:  07/09/2009 11:10:52
Next collection time           :  07/09/2009 11:25:52
Collection interval            :  15


                                            Capacity When  Percentage
Storage     Current Usage  Prediction     Predicted      of Current
Location    in Kilobytes   in Kilobytes   in Kilobytes   to Capacity

__________  _____________  _____________  _____________  ____________
*CSA                 2796           3152           2956           95%
SQA                   455            455           2460           18%
CSA+SQA              3251           3771           5116           64%
ECSA               114922         637703         512700           22%
ESQA                 8414           9319          13184           64%
ECSA+ESQA          123336         646007         525884           23%


 Address spaces with the highest increased usage:

 Job           Storage      Current Usage        Predicted Usage
 Name          Location     in Kilobytes         in Kilobytes

 __________    ________     _______________      _______________
 JOB3          *CSA                    1235                 1523
 JOB1          *CSA                     752                  935
 JOB5          *CSA                     354                  420
 JOB8          *CSA                     152                  267
 JOB2          *CSA                      75                   80
 JOB6          *CSA                      66                   78
 JOB15         *CSA                      53                   55
 JOB18         *CSA                      42                   63
 JOB7          *CSA                      36                   35
 JOB9          *CSA                      31                   34

* = Storage locations that caused the exception.
```

Figure 16. Common storage usage prediction report

Note: In accordance with the IBM Health Checker for z/OS messaging guidelines, the largest generated output length for decimal variable values up to 2147483647 (X'7FFFFFF') is 10 bytes. When any PFA report value is greater than 2147483647, it displays using multiplier notation with a maximum of six characters. For example, if the report value is 2222233333444445555, PFA displays it as 1973P (2222233333444445555 ÷ 1125899906842) using the following multiplier notation:

Table 7. Multiplier notation used in values for PFA reports

Name	Sym	Size
Kilo	K	1,024
Mega	M	1,048,576
Giga	G	1,073,741,824
Tera	T	1,099,511,627,776
Peta	P	1,125,899,906,842

- **Last successful model time:** The date and time of the last successful model for this check. The predictions on this report were generated at that time.

- **Next model time:** The date and time of the next model. The next model will recalculate the predictions.
- **Model interval:** The value in the configured MODELINT parameter for this check. If PFA determines new prediction calculations are necessary, modeling can occur earlier.
- **Last successful collection time:** The date and time of the last successful data collection for this check.
- **Next collection time:** The date and time of the next collection.
- **Collection interval:** The value in the configured COLLECTINT parameter for this check.
- **Storage Location:** The storage location for the values in the row of the report. The location can be one of the following:
  - CSA
  - SQA
  - ECSA
  - ESQA
  - CSA + SQA
  - ECSA + ESQA

  An asterisk (*) printed prior to the storage location indicates that location is the storage location that caused the exception.

  When storage is expanded from SQA or ESQA to CSA or ECSA, an additional message prints on the report, exceptions for the original location are suppressed, and the storage is included in the CSA and ECSA current usage and predictions appropriately.
- **Current Usage in Kilobytes:** The amount of storage used in kilobytes in this storage location when the check was run. The predicted usage for *SYSTEM* jobs is calculated, but no attempt is made to calculate the current usage for *SYSTEM* jobs. Therefore, `UNAVAILABLE` is printed for the current usage of *SYSTEM* jobs.
- **Predicted Usage in Kilobytes:** The prediction of the usage in this storage location for the end of the model interval.
- **Capacity When Predicted in Kilobytes:** The total defined capacity for this storage location (both used and unused) at the time the prediction was made.
- **Percentage of Current to Capacity:** The percent of storage used in kilobytes in this storage location as compared to the capacity available.
- **Address spaces with the highest increased usage:** The address spaces whose storage usage for each individual storage location recently increased the most. The report is sorted by predicted usage within each storage location. This list is only printed if the check issues an exception or the debug parameter is on. The number of jobs printed can vary. An asterisk printed prior to the storage location indicates that is the storage location that caused the exception. If debug is off, the only storage locations printed are those that caused the exception.

  Note: If the SQA expands into the CSA, the CSA usage and predictions include the storage taken from the CSA as SQA and PFA no longer performs comparisons for the SQA. Similarly, if the ESQA expands into the ECSA, the ECSA usage and predictions include the storage taken from the ECSA as ESQA and PFA no longer performs comparisons for the ESQA.

Directories

When you install PFA_COMMON_STORAGE_USAGE, the shell script creates the following directories that hold the executable program, log, error, data store, intermediate, and results files.

Note: The content and names for these files are subject to change and cannot be used as programming interfaces; these files are documented only to provide help in diagnosing problems with PFA.

pfa_directory

This directory contains all the PFA checks and is pointed to by the home directory of the started task. The following files only contain data if messages are generated by the JVM:

- java.stderr (generated by JVM)
- java.stdout (generated by JVM)

pfa_directory/PFA_COMMON_STORAGE_USAGE/data

The directory for common storage usage that holds data and modeling results.

Results files:

- systemName.prediction - The predictions generated by modeling for the six storage locations. This file is used as input to the code that compares the predicted usage with the amount of current usage. The following example shows the common storage usage prediction report in .csv format, which is written to the systemName.prediction file:

```
A/TOTAL,22910,23484
B/TOTAL,763,763
C/CSA  ,316,316
E/ECSA ,14832,14836
Q/ESQA ,8078,8644
S/SQA  ,447,447
```

  Storage location: The location where the storage was allocated. The possible values are:

  – `A/TOTAL`: total above the line common storage (ECSA+ESQA)
  – `B/TOTAL`: total below the line common storage (CSA+SQA)
  – `C/CSA`: common storage area (CSA).
  – `E/ECSA`: extended common storage area (ECSA).
  – `Q/ESQA`: extended system queue area (ESQA).
  – `S/SQA`: system queue area (SQA).
  – `22910`: The current usage when predicted in kilobytes.
  – `23484`: The prediction in kilobytes

- systemName.prediction.html - This file contains an .html report version of the data found in the systemName.prediction file.
- systemName.diag - The predictions for the address spaces whose common storage usage increased the most since the last model. This file is not updated unless debug is on or an exception occurred. This file is used as input to the code that writes the top predicted users on the report.
- systemName.diag.html - The file contents for systemName.diag in .html report format as follows:

- **User of Common Storage:** This is the identification of the user of common storage. It consists of the address space name, ASID, and PSW.
- **Instance Count:** The number of records with this user that were factored into the prediction model.
- **Current Estimated Common Storage Used:** The current amount of common storage used by this user in the last collection interval included in this model.
- **Prediction Look Forward Seconds:** The number of seconds the prediction should project into the future.
- **Predicted Common Storage Usage:** The predicted amount of common storage usage for this user.

Data store files:

- systemName.csaAll.timestamp - The csaAll files contain usage in a collection interval for all address spaces. The usage is categorized by the six locations of common storage tracked by this check.
- systemName.csaSumAll.timestamp - The csaSumAll files summarize the data in the csaAll files. After five days, the csaAll data is averaged and compressed to one file each day, and then time-stamped with the start of that day. The data then moves to a csaSumAll file and csaAll files are deleted.
- systemName.csaTotals.timestamp - The csaTotals files contain the usage of common storage in a collection interval for the six storage locations tracked by this check.
- systemName.csaSumTotals.timestamp - The csaSumTotals files summarize the data in the csaTotals files. After five days, the csaTotals data is averaged and compressed to one file each day, and then time-stamped with the start of that day. The data then moves to a csaSumTotals file and csaTotals files are deleted.

Intermediate files:

- systemName.csadata - The input to modeling in CSV format. The csadata file has one entry per location from the csaTotals files for each collection interval.
- systemName.mapmvs - Convert PSW execution address to module name.
- systemNameMAPREQF.OUT - Contains the location of the module.
- systemName.csaAlldata -- The input to modeling the address spaces whose usage has increased the most in the model interval. This file is in CSV format.
- systemName.csaSumAllX.timestamp - This file is used during summarization of the csaAll files.
- systemName.csaSumTotalsX.timestamp -- This file is used during summarization of the csaTotals files.

This directory contains all the relevant files that are copied from the check's data directory to use when investigating exceptions issued by this check at the timestamp provided in the directory name. Additional information is written to these log files when DEBUG(1).

- systemName.cart.log - The log file generated by modeling code that contains the execution details of modeling code.

- systemNamemapcsa.log- The log file generated by intermediate code that builds the files that are input to modeling with details about code execution.
- systemNameCONFIG.LOG - The log file containing the configuration history for the last 30 days for this check.
- systemNameCOLLECT.LOG - The log file used during data collection.
- systemNameMODEL.LOG - The log file used during portions of the modeling phase.
- systemNameRUN.LOG - The log file used when the check runs.
- systemName.launcher.log - The log file generated by launcher code.
- systemName.tree - This file is generated by the modeling code. It contains information about the model tree that is built based on collected common storage usage data.

pfa_directory/PFA_COMMON_STORAGE_USAGE/EXC_timestamp
This directory contains all the relevant data for investigating exceptions issued by this check at the timestamp provided in the directory name. PFA keeps directories only for the last 30 exceptions. Therefore at each exception, if more than 30 exception directories exist, the oldest directory is deleted so that only 30 exceptions remain after the latest exception is added.

- systemNameREPORT.LOG - The log file containing the same contents as the IBM Health Checker for z/OS report for this exception as well as other diagnostic information issued during report generation.

pfa_directory/PFA_COMMON_STORAGE_USAGE/config
This directory contains the configuration files for the check.

PFA_ENQUEUE_REQUEST_RATE

Description:

| The PFA_ENQUEUE_REQUEST_RATE check detects damage to an address
| space or system by using the number of enqueue requests per CPU millisecond
| used as the tracked metric. If PFA detects that the enqueue request rate is lower than expected, PFA calls Runtime Diagnostics to detect if an address space is hung. If PFA detects that the enqueue request rate is higher than expected, PFA calls Runtime Diagnostics to detect if there is a damaged address space. By detecting these conditions early, you can correct the problem before it causes the system to hang or crash.

The enqueue request rate check issues an exception for the following types of comparisons:

- tracked jobs
- total system

To perform comparisons, the PFA_ENQUEUE_REQUEST_RATE check requires enough data to exist such that current predictions are available for two time ranges.
After the PFA_ENQUEUE_REQUEST_RATE check issues an exception, it does not perform the next comparison type. To avoid skewing the enqueue request rate, PFA ignores the first hour of enqueue data after IPL and the last hour of enqueue data prior to shutdown. In addition, PFA attempts to track the same persistent address spaces that it tracked prior to IPL or PFA restart if the same persistent address spaces are still active. Read the topic about persistent jons in

"PFA_MESSAGE_ARRIVAL_RATE" on page 120 to understand how the PFA_ENQUEUE_REQUEST_RATE check determines the top twenty persistent jobs.

By default, an EXCLUDED_JOBS file containing the address spaces `NETVIEW` and `*MASTER*` on all systems is created during installation. Therefore, if you have not made any modifications to the EXCLUDED_JOBS file, these jobs are excluded. See "Using and configuring supervised learning" on page 86 for more information.

Guidelines

- If you change the maximum number of concurrent ENQ, ISGENQ, RESERVE, GQSCAN and ISGQUERY requests or change system-wide defaults using the SETGRS command or through GRSCNFxx parmlib, delete the files in the PFA_ENQUEUE_REQUEST_RATE/data directory to ensure PFA is collecting relevant information.
- PFA never calls Runtime Diagnostics if PFA detects something is too high. When PFA detects that the enqueue request rate is higher than expected, PFA issues an exception indicating that an address space or the system might be damaged.

Note: This check supports supervised learning. See the topic on "Using and configuring supervised learning" on page 86.

| **Reason for check:**

The objective of this check is to determine if an LPAR or address space is damaged or hung by using the number of enqueues per CPU millisecond as the tracked metric.

Best practice:

The best practice is to analyze the message and reports issued by PFA to determine what is causing the increase or decrease in the enqueue request rate.

z/OS releases the check applies to:

z/OS V1R13 and later.

Type of check:

Remote

Parameters accepted:

Yes, as follows:

Table 8. PFA_ENQUEUE_REQUEST_RATE check parameters

Parameter name	Default value	Minimum Value	Maximum Value	Description
collectint	1 Minute	1	360	This parameter determines how often (in minutes) to run the data collector that retrieves the current enqueue request rate.
modelint	720 Minutes	60	1440	This parameter determines how often (in minutes) you want the system to analyze the data and construct a new enqueue request rate model or prediction. By default, PFA analyzes the data and constructs a new model every "default value" minutes. The model interval must be at least four times larger than the collection interval. Note that, even when you set a value larger than 360, PFA performs the first model at 360 minutes (6 hours). By default, PFA analyzes the data and constructs a new model every 720 minutes (12 hours).

Table 8. PFA_ENQUEUE_REQUEST_RATE check parameters (continued)

Parameter name	Default value	Minimum Value	Maximum Value	Description
stddev	10	2	100	This parameter is used to specify how much variance is allowed between the actual enqueue request rate per amount of CPU and the expected enqueue request rate. It determines if the actual enqueue request rate has increased beyond the allowable upper limit and how much variance is allowed across the time range predictions. If you set the STDDEV parameter to a smaller value, an exception issues when the actual enqueue request rate is closer to the expected enqueue request rate and the predictions across the time ranges are consistent. If you set the STDDEV parameter to a larger value, an exception issues when the actual enqueue request rate is significantly greater than the expected enqueue request rate even if the predictions across the different time ranges are inconsistent.
collectinactive	1 (on)	0 (off)	1 (on)	Defines whether data is collected and modeled even if the check is not eligible to run, not ACTIVE(ENABLED), in IBM Health Checker for z/OS.
trackedmin	3	0	1000	This parameter defines the minimum enqueue request rate required for a persistent job in order for it to be considered a top persistent job that should be tracked individually.
exceptionmin	1	0	1000	This parameter is used when determining if an exception should be issued for an unexpectedly high enqueue request rate. For tracked jobs, this parameter defines the minimum enqueue request rate and the minimum predicted enqueue request rate required to cause a too high exception. For the total system comparison, this parameter defines the minimum enqueue request rate required to cause a too high exception.
checklow	1	0	1	Defines whether Runtime Diagnostics is run to validate that a low enqueue request rate is caused by a problem. If this value is off, PFA does not issue exceptions for conditions in which the enqueue request rate is unexpectedly low.
stddevlow	4	2	100	This parameter is used to specify how much variance is allowed between the actual enqueue request rate per amount of CPU, and the expected enqueue request rate, when determining if the actual rate is unexpectedly low. • If you set the STDDEVLOW parameter to a smaller value, an exception is issued when the actual enqueue request rate is closer to the expected enqueue request rate. • If you set the STDDEVLOW parameter to a larger value, an exception is issued when the actual enqueue request rate is significantly lower than the expected enqueue request rate.
limitlow	3	1	100	This parameter defines the maximum enqueue request rate allowed when issuing an exception for an unexpectedly low number of enqueues.
debug	0 (off)	0 (off)	1 (on)	This parameter (an integer of 0 or 1) is used at the direction of IBM service to generate additional diagnostic information for the IBM Support Center. This debug parameter is used in place of the IBM Health Checker for z/OS policy. The default is off (0).

To determine the status of the enqueue request rate check, issue `f pfa,display,check(PFA_ENQUEUE_REQUEST_RATE),detail`. For the command example and more details, see . The following example shows the output written to message AIR018I in SDSF:

```
AIR018I 02:22:54 PFA CHECK DETAIL

CHECK NAME:  PFA_ENQUEUE_REQUEST_RATE
    ACTIVE                          : YES
    TOTAL COLLECTION COUNT          : 5
    SUCCESSFUL COLLECTION COUNT     : 5
    LAST COLLECTION TIME            : 02/05/2009 10:18:22
    LAST SUCCESSFUL COLLECTION TIME : 02/05/2009 10:18:22
    NEXT COLLECTION TIME            : 02/05/2009 10:19:22
    TOTAL MODEL COUNT               : 1
    SUCCESSFUL MODEL COUNT          : 1
    LAST MODEL TIME                 : 02/05/2009 10:18:24
    LAST SUCCESSFUL MODEL TIME      : 02/05/2009 10:18:24
    NEXT MODEL TIME                 : 02/05/2009 22:18:24
    CHECK SPECIFIC PARAMETERS:
       COLLECTINT                   : 1
       MODELINT                     : 720
       COLLECTINACTIVE              : 1=ON
       DEBUG                        : 0=OFF
       STDDEV                       : 10
       TRACKEDMIN                   : 3
       EXCEPTIONMIN                 : 1
       CHECKLOW                     : 1=ON
       STDDEVLOW                    : 4
       LIMITLOW                     : 3
```

User override of IBM values:

The following shows keywords you can use to override check values on either a POLICY statement in the HZSPRMxx parmlib member or on a MODIFY command. This statement can be copied and modified to override the check defaults:

```
UPDATE CHECK(IBMPFA,PFA_ENQUEUE_REQUEST_RATE)
            ACTIVE
            SEVERITY(MEDIUM)
            INTERVAL(ONETIME)
      PARMS=('COLLECTINT(1)','MODELINT(720)','STDDEV(10)','DEBUG(0)',
            'COLLECTINACTIVE(1)','EXCEPTIONMIN(1)','TRACKEDMIN(3)')
            'CHECKLOW(1)','STDDEVLOW(4)','LIMITLOW(3)'
            DATE(20080330)
      REASON('The enqueue request rate is higher than expected
              which can indicate a damaged address space.')
```

The enqueue request rate check is designed to run automatically after every data collection. Do not change the INTERVAL parameter.

Verbose support:

The check provides additional detail in verbose mode. You can put a check into verbose mode using the UPDATE,filters,VERBOSE=ON parameters on either the MODIFY command or in a POLICY statement in an HZSPRMxx parmlib member.

Debug support:

The DEBUG parameter in IBM Health Checker for z/OS is ignored by this check. Rather, the debug parameter is a PFA check specific parameter. For details, see “Understanding how to modify PFA checks” on page 79.

Reference:

For more information about PFA, see the topic on “Overview of Predictive Failure Analysis” on page 65.

Messages:

The output is a enqueue request rate prediction report that corresponds to the message issued. PFA generates one of the following reports:

- AIRH190E - Enqueue request rate lower than expected exception
- AIRH192E - Enqueue request rate higher than expected exception
- AIRH210E - Total system enqueue request rate higher than expected exception
- AIRH211E - Total system enqueue request rate system lower than expected exception
- AIRH216I - Runtime Diagnostic output

For complete message information, see the topics on:

- AIRH messages in *z/OS MVS System Messages, Vol 1 (ABA-AOM)*.
- AIR messages*z/OS MVS System Messages, Vol 1 (ABA-AOM)*.

SECLABEL recommended for MLS users:

SYSLOW

Output:

The output is a variation of the enqueue request rate prediction report. The values found in the enqueue request prediction file are as follows:

Tracked jobs exception report for enqueue request rate higher than expected: PFA issues this report when any one or more tracked, persistent jobs cause an exception due to the enqueue request rate being higher than expected. Only the tracked jobs that caused an exception are in the list of jobs on the report.

Tracked jobs exception report for enqueue request rate lower than expected:

```
    Enqueue Request Rate Prediction Report
Last successful model time     : 01/27/2009 11:08:01
Next model time           : 01/27/2009 23:08:01
Model interval           : 720
Last successful collection time  : 01/27/2009 17:41:38
Next collection time        : 01/27/2009 17:56:38
Collection interval         : 15

Persistent address spaces with high rates:
                                            Predicted Enqueue
                      Enqueue                 Request Rate
  Job                 Request
  Name     ASID          Rate         1 Hour        24 Hour           7 Day
  TRACKED1 001D         58.00          23.88          22.82           15.82
  TRACKED2 0028         11.00           0.34          11.11           12.11
  TRACKED3 0029         11.00          12.43           2.36            8.36
```

Figure 17. Prediction report for enqueue request rate higher than expected - total jobs

PFA issues this report when any one or more tracked, persistent jobs cause an exception due to the enqueue request rate being lower than expected. Only the tracked jobs that caused an exception are in the list of jobs on the report.

```
      Enqueue Request Rate Prediction Report
Last successful model time     : 10/10/2010 11:08:01
Next model time           : 10/10/2010 23:08:01
Model interval           : 720
Last successful collection time  : 10/10/2010 17:41:38
Next collection time         : 10/10/2010 17:56:38
Collection interval          : 15

Persistent address spaces with low rates:
                                               Predicted Enqueue
                          Enqueue                 Request Rate
  Job                     Request
  Name       ASID           Rate         1 Hour       24 Hour         7 Day
  IBMUSER2   002F       1.17       23.88      22.82      15.82
  IBMUSER1   002E       2.01        8.34      11.11      12.11

Runtime Diagnostics Output:
Runtime Diagnostics detected a problem in job: JOBS4
  EVENT 06: HIGH - HIGHCPU - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID CPU RATE: 96% ASID: 0027 JOBNAME: JOBS4
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MAY BE LOOPING.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
  EVENT 07: HIGH - LOOP - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID: 0027 JOBNAME: JOBS4 TCB: 004E6850
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE APPEARS TO BE IN A LOOP.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
Runtime Diagnostics detected a problem in job: JOBS5
  EVENT 03: HIGH - HIGHCPU - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID CPU RATE: 96% ASID: 0027 JOBNAME: JOBS5
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MAY BE LOOPING.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
  EVENT 04: HIGH - LOOP - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID: 0027 JOBNAME: JOBS5 TCB: 004E6850
STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE APPEARS TO BE IN A LOOP.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
```

Figure 18. Prediction report for enqueue request rate lower than expected - total jobs

Total system exception report for enqueue request rate higher than expected: The no problem report and the total system exception report (when the rate is higher than expected) show the totals at the top and the list of the tracked jobs.

```
 Enqueue request rate Prediction Report

Last successful model time      :  01/27/2009 17:08:01
Next model time                 :  01/27/2009 23:08:01
Model interval                  :  360
Last successful collection time :  01/27/2009 17:41:38
Next collection time            :  01/27/2009 17:56:38
Collection interval             :  15

Enqueue request rate
 at last collection interval        :  83.52
Prediction based on 1 hour of data  :  98.27
Prediction based on 24 hours of data:  85.98
Prediction based on 7 days of data  : 100.22
Top persistent users:

                                             Predicted Enqueue
                         Enqueue                  Request Rate
  Job                Request
  Name     ASID         Rate        1 Hour       24 Hour          7 Day
  TRACKED1 001D        58.00         23.88         22.82          15.82
  TRACKED2 0028        11.00          0.34         11.11          12.11
  TRACKED3 0029        11.00         12.43          2.36           8.36
```

Figure 19. Total system exception report: enqueue request rate higher than expected

Total system exception report for enqueue request rate lower than expected: PFA issues the enqueue request rate exception report when there is a shortage or unusually low rate of enqueue requests. Runtime Diagnostics examines the system and PFA lists all output it receives from Runtime Diagnostics.

```
 Enqueue Request Rate Prediction Report

Last successful model time      :  01/27/2009 11:08:01
Next model time                 :  01/27/2009 23:08:01
Model interval                  :  720
Last successful collection time :  01/27/2009 17:41:38
Next collection time            :  01/27/2009 17:56:38
Collection interval             :  15

Persistent address spaces with low rates:

                                             Predicted ENQ
                         ENQ                  Request Rate
  Job                Request
  Name      ASID          Rate        1 Hour       24 Hour          7 Day
  JOBS4    001F    1.17       23.88     22.82      15.82
  JOBS5    002D    2.01       8.34     11.11      12.11

Runtime Diagnostics Output:
----------------------------------------------------------------------
EVENT 01: HIGH - ENQ          - SYSTEM: SY1      2010/10/04 - 10:19:53
ENQ WAITER  - ASID:002F - JOBNAME:IBMUSER2 - SYSTEM:SY1
ENQ BLOCKER - ASID:002E - JOBNAME:IBMUSER1 - SYSTEM:SY1
QNAME: TESTENQ
RNAME: TESTOFAVERYVERYVERYVERYLOOOOOOOOOOOOOOOOOOOOOONGRNAME1234567...
  ERROR: ADDRESS SPACES MIGHT BE IN ENQ CONTENTION.
 ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE BLOCKING JOBS AND
 ACTION: ASIDS.
----------------------------------------------------------------------
```

Figure 20. Total system exception report: low enqueue request rate

Note: In accordance with the IBM Health Checker for z/OS messaging guidelines, the largest generated output length for decimal variable values up to 2147483647 (X'7FFFFFF') is 10 bytes. When any PFA report value is greater

than 2147483647, it displays using multiplier notation with a maximum of six characters. For example, if the report value is 2222233333444445555, PFA displays it as `1973P` (2222233333444445555 ÷ 1125899906842) using the following multiplier notation:

Table 9. Multiplier notation used in values for PFA reports

Name	Sym	Size
Kilo	K	1,024
Mega	M	1,048,576
Giga	G	1,073,741,824
Tera	T	1,099,511,627,776
Peta	P	1,125,899,906,842

The following fields apply to all reports:

- **Last successful model time:** The date and time of the last successful model for this check. The predictions on this report were generated at that time.
- **Next model time:** The date and time of the next model. The next model will recalculate the predictions.
- **Model interval:** The value in the configured MODELINT parameter for this check. If PFA determines new prediction calculations are necessary, modeling can occur earlier.
- **Last successful collection time:** The date and time of the last successful data collection for this check.
- **Next collection time:** The date and time of the next collection.
- **Collection interval:** The value in the configured COLLECTINT parameter for this check.
- | **Enqueue request rate in last collection interval:** The actual enqueue request rate in the last collection interval where the rate is defined to be the count returned by the **GRS ISGQUERY** API normalized by the milliseconds used.
- **Predicted rates based on...:** The enqueue request rates based on one hour, 24 hours, and seven days. If no prediction is available for a given time range, the line is not printed. For example, if the check has been running for 2 days, there is not enough data for seven days of data therefore PFA does not print the "Prediction based on 7 days of data" line. If there is not enough data for a time range, `INELGIBLE` is printed for that time range and no comparisons are made.
- **Runtime Diagnostics Output:** Runtime Diagnostics event records to assist you in diagnosing and fixing the problem. See the topic on "Runtime Diagnostics symptoms" on page 39 in Chapter 4, "Runtime Diagnostics," on page 35.
- **Job Name:** The name of the job that has enqueue arrivals in the last collection interval.
- **ASID:** The ASID for the job that has enqueue arrivals in the last collection interval.
- **Enqueue request rate:** The current enqueue request rate for the system.
- **Predicted enqueue request rate:** The predicted enqueue request rates based on one hour, 24 hours, and seven days of data. If PFA did not previously run on this system or the same jobs previously tracked are not all active, there is not be enough data for two prediction time ranges until that amount of time has passed. Also, gaps in the data caused by stopping PFA or by an IPL might cause the time ranges to not have enough data available. After the

check collects enough data for two time ranges, predictions are made again for those time ranges. If there is not enough data for two time ranges, INELIGIBLE is printed and comparisons are not made.

- **Runtime Diagnostics Output:** The reports generated by Runtime Diagnostic for this check. These reports contain additional details to help you narrow down the source of the problem and sometimes corrective actions you can take. For complete details about using Runtime Diagnostics, see Chapter 4, “Runtime Diagnostics,” on page 35.

Directories

Note: The content and names for these files and directories are subject to change and cannot be used as programming interfaces; these files are documented only to provide help in diagnosing problems with PFA.

pfa_directory

This directory contains all the PFA checks and is pointed to by the home directory of the started task. The following files only contain data if messages are generated by the JVM:

- java.stderr (generated by JVM)
- java.stdout (generated by JVM)

pfa_directory/PFA_ENQUEUE_REQUEST_RATE/data

The directory for enqueue request rate that holds data and modeling results. PFA automatically deletes the contents of the `PFA_ENQUEUE_REQUEST_RATE/data` directory that could lead to skewed predictions in the future.

Guideline: If the use of the z/OS image is radically different after an IPL (for instance, the change from a test system to a production system) of if you modify anything that affects enqueue details, delete the files in the *PFA_ENQUEUE_REQUEST_RATE/data* directory to ensure the check can collect the most accurate modeling information.

Results files

- systemName.1hr.prediction - This file is generated by the modeling code for the predictions made for one hour of historical data. It contains predictions for each of the tracked address spaces and the total system category. It also contains additional information required for PFA processing.
- systemName.24hr.prediction - This file is generated by the modeling code for the predictions made for 24 hours of historical data. It contains predictions for each of the tracked address spaces and the total system category. It also contains additional information required for PFA processing.
- systemName.7day.prediction - This file is generated by the modeling code for the predictions made for seven days of historical data. It contains predictions for each of the tracked address spaces and the total system category. It also contains additional information required for PFA processing.
- systemName.1hr.prediction.html - This file contains an .html report version of the data found in the systemName.1hr.prediction file.
- systemName.24hr.prediction.html - This file contains an .html report version of the data found in the systemName.24hr.prediction file.
- systemName.7day.prediction.html - This file contains an .html report version of the data found in the systemName.7day.prediction file.

- systemName.prediction.stddev - The file generated by the modeling code to list the standard deviation of the predictions across the time ranges for each job.

Data store files:

- systemName.OUT - The data collection file.

Intermediate files:

- systemName.data - The file is used as input to the modeling to track if enough data is available to model.
- systemName.1hr.data - The file used as input to modeling code. It contains one hour of historical data.
- systemName.24hr.data - The file used as input to modeling code. It contains 24 hours of historical data.
- systemName.7day.data - The file used as input to modeling code. It contains seven days of historical data.
- systemName.1hr.holes - The file is used to track gaps in data, caused by stopping PFA or by an IPL, for a one hour period.
- systemName.24hr.holes - The file is used to track gaps in the data, caused by stopping PFA or by an IPL, for a 24 hour time period.
- systemName.7day.holes - The file is used to track gaps in the data, caused by stopping PFA or by an IPL, for the seven day time period.

This directory holds the following log files. Additional information is written to these log files when DEBUG(1).

- systemName.1hr.cart.log - The log file generated by modeling code with details about code execution while one hour of historical data was being modeled.
- systemName.24hr.cart.log - The log file generated by modeling code with details about code execution while 24 hours of historical data was being modeled.
- systemName.7day.cart.log - The log file generated by modeling code with details about code execution while seven days of historical data was being modeled.
- systemName.builder.log - The log file generated by intermediate code that builds the files that are input to modeling with details about code execution.
- systemName.launcher.log - The log file generated by launcher code.
- systemName.1hr.tree - This file is generated by the modeling code. It contains information about the model tree which was built based on the last one hour of collected data.
- systemName.24hr.tree - This file is generated by the modeling code. It contains information about the model tree which was built based on the last 24 hours of collected data.
- systemName.7day.tree - This file is generated by the modeling code. It contains information about the model tree which was built based on the last seven days of collected data.
- systemNameCONFIG.LOG - The log file containing the configuration history for the last 30 days for this check.
- systemNameCOLLECT.LOG - The log file used during data collection.
- systemNameMODEL.LOG - The log file used during portions of the modeling phase.

- systemNameRUN.LOG - The log file used when the check runs.

pfa_directory/PFA_ENQUEUE_REQUEST_RATE/EXC_timestamp
This directory contains all the relevant data for investigating exceptions issued by this check at the timestamp provided in the directory name. PFA keeps directories only for the last 30 exceptions. Therefore at each exception, if more than 30 exception directories exist, the oldest directory is deleted so that only 30 exceptions remain after the latest exception is added.

- systemNameREPORT.LOG - The log file containing the same contents as the IBM Health Checker for z/OS report for this exception as well as other diagnostic information issued during report generation.

pfa_directory/PFA_ENQUEUE_REQUEST_RATE/config
This directory contains the configuration files for the check.

- EXCLUDED_JOBS - The file containing the list of excluded jobs for this check.

Note: When using Runtime Diagnostics, it is possible to see data for jobs previously defined to the excluded jobs list in the “other persistent jobs” and “total system” categories because PFA must return any potential problem activity on the system identified by Runtime Diagnostics.

PFA_JES_SPOOL_USAGE

Description:

The JES spool usage check detects abnormalities in persistent jobs (a persistent job is a job that starts in the first hour after IPL). The check tracks the top 15 persistent jobs, by name, that have the greatest change in the number of track groups used since the last model. If the jobs stops and restarts, they are still considered persistent by name.

PFA uses the metric of the amount of JES spool usage to determine if an address space is behaving abnormally based on the previous behavior for the address space. If a change in JES spool usage is too high, it can indicate a damaged address space. This check detects abnormalities in the amount JES spool usage as follows:

- The check collects data for all the persistent jobs that start within an hour after IPL.
- Modeling occurs for the top 15 persistent jobs that had the greatest amount of change in the model interval.
- If abnormal behavior is detected based on the expected values for a persistent job that has been modeled, PFA issues a health check exception message.
- When an exception occurs, the check reports the persistent jobs whose JES spool usage was abnormal (those jobs that caused the exception).
- When no problem exists, the check reports the persistent jobs that were modeled (for informational purposes).
- The check detects when a persistent job has restarted and still considers it persistent even if the restart occurred more than one hour after IPL.
- If there are duplicate jobs that are persistent, neither job is tracked.
- This check supports supervised learning. For details, see “Using and configuring supervised learning” on page 86.

Reason for check:
The objective of the JES spool usage check is to detect address spaces that are damaged by comparing the amount of change in the size of the JES spool used by the address space to the expected value.

Best practice:
The best practice is to inspect the job log and other files written to the JES spool. If a job's spool usage is higher than expected, this can indicate a damaged job. To fix the problem, recycle the job.

z/OS releases the check applies to:
z/OS V1R13 and later.

Type of check:
Remote

Restrictions:
This check applies only to JES2.

Parameters accepted:
Yes, as follows:

Table 10. PFA_JES_SPOOL_USAGE check parameters

Parameter name	Default value	Minimum Value	Maximum Value	Description
collectint	5 Minutes	1	360	This parameter determines the time (in minutes) to run the data collector that determines the number of track groups used. The default is 5 minutes (5).
modelint	720 Minutes	4	1440	This parameter determines how often (in minutes) you want the system to analyze the data and construct a new model. Note that, even when you set a value larger than 360, PFA performs the first model at 360 minutes (6 hours). By default, PFA analyzes the data and constructs a new model every 720 minutes (12 hours).
collectinactive	1 (on)	0 (off)	1 (on)	Defines whether data will be collected and modeled even if the check is not eligible to run, not ACTIVE(ENABLED), in IBM Health Checker for z/OS.
stddev	3	2	100	The number by which to multiply the predicted JES spool used to determine if the actual used has increased beyond the allowable limit, which might indicate an address space is damaged.
exceptionmin	10	0 (off)	1000	This parameter defines the minimum JES spool usage required to cause a too high exception.
debug	0 (off)	0 (off)	1 (on)	This parameter (an integer of 0 or 1) is used at the direction of IBM service to generate additional diagnostic information for the IBM Support Center. This debug parameter is used in place of the IBM Health Checker for z/OS policy. The default is off (0).

To determine the status of the JES spool usage check, issue `f pfa,display,check(PFA_JES_SPOOL_USAGE),detail`. For the command example and more details, see . The following example shows the output written to message AIR018I in SDSF:

```
AIR018I 02.22.54 PFA CHECK DETAIL

CHECK NAME:  PFA_JES_SPOOL_USAGE
    ACTIVE                          : YES
    TOTAL COLLECTION COUNT          : 5
    SUCCESSFUL COLLECTION COUNT     : 5
    LAST COLLECTION TIME            : 02/05/2009 10.18.22
    LAST SUCCESSFUL COLLECTION TIME : 02/05/2009 10.18.22
    NEXT COLLECTION TIME            : 02/05/2009 10.23.22
    TOTAL MODEL COUNT               : 1
    SUCCESSFUL MODEL COUNT          : 1
    LAST MODEL TIME                 : 02/05/2009 10.18.24
    LAST SUCCESSFUL MODEL TIME      : 02/05/2009 10.18.24
    NEXT MODEL TIME                 : 02/05/2009 22.18.24
    CHECK SPECIFIC PARAMETERS:
       COLLECTINT                   : 5
       MODELINT                     : 720
       COLLECTINACTIVE              : 1=YES
       DEBUG                        : 0=NO
       STDDEV                       : 3
       EXCEPTIONMIN                 : 10
```

User override of IBM values:

The following shows keywords you can use to override check values on either a POLICY statement in the HZSPRMxx parmlib member or on a MODIFY command. This statement can be copied and modified to override the check defaults:

```
UPDATE CHECK(IBMPFA,PFA_JES_SPOOL_USAGE)
            ACTIVE
            SEVERITY(MEDIUM)
            INTERVAL(ONETIME)
      PARMS=('COLLECTINT(5)','MODELINT(720)','STDDEV(3)','DEBUG(0)',
            'COLLECTINACTIVE(1)','EXCEPTIONMIN(10)', DATE(20100505)
      REASON('To detect a damaged address space by comparing the amount of
             change in the size of the JES spool to the expected value.')
```

Note: The JES spool usage check is designed to run automatically after every data collection. Do not change the INTERVAL parameter.

Verbose support:

The check provides additional detail in verbose mode. You can put a check into verbose mode using the `UPDATE,filters,VERBOSE=ON` parameters on either the MODIFY command or in a POLICY statement in an HZSPRMxx parmlib member.

Debug support:

The DEBUG parameter in IBM Health Checker for z/OS is ignored by this check. Rather, the debug parameter is a PFA check specific parameter. For details, see “Understanding how to modify PFA checks” on page 79.

Reference:

For more information about PFA, see the topic on “Overview of Predictive Failure Analysis” on page 65.

Messages:

The output is a JES spool usage prediction report that corresponds to the message issued. PFA generates one of the following reports:

- AIRH198E - JES spool usage exception report

For additional message information, see the topics on:

- AIRH messages in *z/OS MVS System Messages, Vol 1 (ABA-AOM)*.
- AIR messages*z/OS MVS System Messages, Vol 1 (ABA-AOM)*.

SECLABEL recommended for MLS users:
SYSLOW

Output:

The output is a variation of the JES spool usage prediction report. The values found are as follows:
Persistent Jobs Exception Report (for AIRH198E): PFA issues this report when any one or more top persistent jobs cause an exception. Only the top jobs that caused an exception are in the list of jobs on the report.
No problem report: When no exception is issued for the JES spool usage check

```
 JES Spool Usage Prediction Report

Last successful model time    : 01/28/2010 16:10:15
Next model time         : 01/28/2010 16:14:57
Model interval          : 25
Last successful collection time : 01/28/2010 16:09:57
Next collection time       : 01/28/2010 16:14:57
Collection interval       : 5

Address spaces causing exception:

           Current Change in    Expected Change in          Current
Job           Number of Track       Number of Track  Number of Track
Name  ASID       Groups Used           Groups Used      Groups Used
JOB1  0019               252                    10             892
JOB55  000E               129                     3              400
```

Figure 21. JES spool usage persistent jobs exception report

(AIRH200I) is issued, the following report is generated:

```
 JES Spool Usage Prediction Report

Last successful model time    : 01/27/2010 15:58:13
Next model time         : 01/27/2010 16:03:05
Model interval          : 25
Last successful collection time : 01/27/2010 15:58:05
Next collection time       : 01/27/2010 16:03:05
Collection interval       : 5

Address spaces with the highest increased usage:

      Current Change in  Expected Change in       Current
Job           Number of Track    Number of Track   Number of Track
Name   ASID    Groups Used       Groups Used     Groups Used
JOB1   0019         52          51          892
JOB2   0014         30          35          735
JOB55  000E         29          23          400
JOB16  0009         23          16          452
```

Figure 22. JES spool usage no problem report

Note: In accordance with the IBM Health Checker for z/OS messaging guidelines, the largest generated output length for decimal variable values up to 2147483647 (X'7FFFFFF') is 10 bytes. When any PFA report value is greater than 2147483647, it displays using multiplier notation with a maximum of six characters. For example, if the report value is 2222233333444445555, PFA displays it as 1973P (2222233333444445555 ÷ 1125899906842) using the following multiplier notation:

Table 11. Multiplier notation used in values for PFA reports

Name	Sym	Size
Kilo	K	1,024
Mega	M	1,048,576
Giga	G	1,073,741,824
Tera	T	1,099,511,627,776
Peta	P	1,125,899,906,842

The following fields apply to all four reports:

- **Last successful model time:** The date and time of the last successful model for this check. The predictions on this report were generated at that time.
- **Next model time:** The date and time of the next model. The next model will recalculate the predictions.
- **Model interval:** The value in the configured MODELINT parameter for this check. If PFA determines new prediction calculations are necessary, modeling can occur earlier.
- **Last successful collection time:** The date and time of the last successful data collection for this check.
- **Next collection time:** The date and time of the next collection.
- **Collection interval:** The value in the configured COLLECTINT parameter for this check.
- **Address spaces with the highest increased usage:** The address spaces with the highest recent increase in JES spool usage since the last collection.
- **Address spaces causing the exception:** The address spaces with the highest recent increase in JES spool usage whose increase is greater than expected and caused the exception.
- **Job Name:** The name of the job that has increased usage in the last collection interval.
- **ASID:** The ASID for the job that has increased usage in the last collection interval.
- **Current change in number of track groups used:** The current change in track groups (units of SPOOL space) that a job is using.
- **Expected change in number of track groups used:** The expected change in the number of track groups used by the persistent job.
- **Current number of track groups used:** The current number of track groups being used by the persistent address space.

Directories

Note: The content and names for these files and directories are subject to change and cannot be used as programming interfaces; these files are documented only to provide help in diagnosing problems with PFA.

pfa_directory
This directory contains all the PFA checks and is pointed to by the home directory of the started task. The following files only contain data if messages are generated by the JVM:

- java.stderr (generated by JVM)
- java.stdout (generated by JVM)

pfa_directory/PFA_JES_SPOOL_USAGE/data
The directory for JES spool usage that holds data and modeling results.

PFA automatically deletes the contents of the `PFA_JES_SPOOL_USAGE/data` directory that could lead to skewed predictions in the future.

Results files

- systemName.prediction - This file is generated by the modeling code. It lists the jobs that have the highest recent growth in use of spool usage. For each job in the list, it records the spool usage and additional information required for PFA processing.
- systemName.prediction.html - This file contains an .html report version of the data found in the systemName.prediction file.

Data store files:

- systemName.All.timestamp - The data collection file.
- systemName.Sum.timestamp - The Sum file summarize the data in the All files. After five days, the ALL data is averaged and compressed to one file each day, and then time-stamped with the start of that day. The data then moves to a Sum file and ALL files are deleted.

Intermediate files:

- systemName.data - The file is used as input to the modeling to track if enough data is available to model.
- systemName.SumX.timestamp - This file is used during summarization of the All files.

This directory holds the following log files. Additional information is written to these log files when DEBUG(1).

- systemName.builder.log - The log file generated by intermediate code that builds the files that are input to modeling with details about code execution.
- systemName.launcher.log - The log file generated by launcher code.
- systemNameCONFIG.LOG - The log file containing the configuration history for the last 30 days for this check.
- systemNameCOLLECT.LOG - The log file used during data collection.
- systemNameMODEL.LOG - The log file used during portions of the modeling phase.
- systemNameRUN.LOG - The log file used when the check runs.
- systemName.cart.log - The log file generated by modeling code that contains the execution details of modeling code.
- systemName.tree - This file is generated by the modeling code. It contains information about the model tree that is built based on collected JES spool usage data.

pfa_directory/PFA_JES_SPOOL_USAGE/EXC_timestamp

This directory contains all the relevant data for investigating exceptions issued by this check at the timestamp provided in the directory name. PFA keeps directories only for the last 30 exceptions. Therefore at each exception, if more than 30 exception directories exist, the oldest directory is deleted so that only 30 exceptions remain after the latest exception is added.

- systemNameREPORT.LOG - The log file containing the same contents as the IBM Health Checker for z/OS report for this exception as well as other diagnostic information issued during report generation.

pfa_directory/PFA_JES_SPOOL_USAGE/config
This directory contains the configuration files for the check.
- EXCLUDED_JOBS - The file containing the list of excluded jobs for this check.

PFA_LOGREC_ARRIVAL_RATE

Description:
The check is looking at the arrival frequency of selected software logrec entries. By monitoring the arrival rate of these logrec entries over time, PFA can detect when the rate of logrec entries exceeds what is considered normal for a system. An unusually high rate of logrec entries can be indicative of recurring failures on the system. PFA can identify when these rates exceed the normal frequency and make an accurate prediction of when you need to take corrective action.

Analyzing the arrival rate by category prevents an expected, normal, but large number of logrecs in the key 8-15 category from masking an unexpected, critical, but small number of logrecs in the key 0 category.

To avoid skewing the logrec arrival rate, PFA ignores the first hour of logrec arrivals after IPL and the last hour of logrec arrivals prior to shutdown.

Tip: z/OS logrec provides two options for the logrec recording medium. Your installation either uses System Logger to produce a logrec log stream or writes logrec to a data set. When a logrec is produced, PFA is notified through an ENF listener. If your installation is set up to write to a data set and that data set fills up, PFA will stop getting notification when a logrec is produced. Therefore, for the best reliability, it is recommended that you use the log stream method with PFA_LOGREC_ARRIVAL_RATE check.

Reason for check:
By detecting recurring failures on the system early, you are able to take corrective action for the failure before a system outage results.

Best practice:
The best practice is to:
1. To determine if the number of software logrec entries is excessive, look at the Logrec Arrival Rate Prediction Report. See "Output:" on page 116 for an example.
2. If the number of software logrec entries are excessive, look at the system logrec entries and try to identify any trend that might exist. That is, for example, many LOGRECs are associated with a single job (address space) or component. For IBM code, the first three letters of the module name identify the component. For component identification, see the module identification chart in *z/OS MVS Diagnosis: Reference*.
   a. If a particular job (address space) is causing the problem, look at the job logs in SDSF.
      - If the job is issuing messages, follow the directions in the message text.
   b. If the job supports commands to evaluate its status, issue those commands.
      - If the job is responsive, if possible, schedule a recycle for the time that has the lowest business impact.
      - If the job is nonresponsive, if possible, capture diagnostic information and recycle the job.
   c. Look in SYSLOG for messages issued by the job or about the job.

- If the job is issuing message, follow the directions in the message text.

3. If a particular component can be identified as causing the problem, follow the standard approach to diagnosing a problem with the component. (For example, have recent changes been made that impact the component?)
   - Look for messages issued by that component in the SYSLOG. If the component is issuing message, follow the directions in the message text.
   - Some components have monitors or operator commands to further evaluate the health of the component.
4. When in a parallel sysplex, you have the option of moving work from the LPAR experiencing the problem (excessive number of logrec entries) to a different LPAR.

z/OS releases the check applies to:
z/OS V1R10 and later.

Type of check:
Remote

Parameters accepted:
Yes, as follows:

Table 12. PFA_LOGREC_ARRIVAL_RATE check parameters

Parameter name	Default value	Minimum Value	Maximum Value	Description
collectint	60 Minutes	1	360	This parameter determines the time (in minutes) to run the data collector that determines the amount of logrec entries. The default is 60 minutes (60).
modelint	720 Minutes	4	1440	This parameter determines how often (in minutes) you want the system to analyze the data and construct a new model. Note that, even when you set a value larger than 360, PFA performs the first model at 360 minutes (6 hours). By default, PFA analyzes the data and constructs a new model every 720 minutes (12 hours).
stddev	2	1	100	This parameter is used to specify how much variance is allowed between the actual logrec arrival rate and the expected logrec arrival rate. It also determines how much variance is allowed across the time range predictions. If you set the STDDEV parameter to a small value, an exception will be issued if the actual logrec arrivals are closer to the expected logrec arrivals and the predictions across the time ranges are consistent. If you set the STDDEV parameter to a larger value, an exception will be issued if the actual logrec arrivals are significantly greater than the expected logrec arrivals even if the predictions across the different time ranges are inconsistent.
collectinactive	1	0 (off)	1 (on)	Defines whether data will be collected and modeled even if the check is not eligible to run (is not ACTIVE(ENABLED)) in IBM Health Checker for z/OS.
exceptionmin	25	0	1000	This parameter defines the minimum logrec arrival rate and the minimum predicted logrec arrival rate required to cause a too high exception.
debug	0	0 (off)	1 (on)	This parameter (an integer of 0 or 1) is used at the direction of IBM service to generate additional diagnostic information for the IBM Support Center. This debug parameter is used in place of the IBM Health Checker for z/OS policy. The default is off (0).

To determine the status of the logrec arrival usage check, issue `f pfa,display,check(pfa_logrec_arrival_rate),detail`. For the command example and more details, see . The following is an example of the output written to message AIR018I in SDSF user log (ULOG):

```
AIR018I 02:22:54 PFA CHECK DETAIL

CHECK NAME:  PFA_LOGREC_ARRIVAL_RATE
    ACTIVE                          : YES
    TOTAL COLLECTION COUNT          : 5
    SUCCESSFUL COLLECTION COUNT     : 5
    LAST COLLECTION TIME            : 04/05/2008 10:18:22
    LAST SUCCESSFUL COLLECTION TIME : 04/05/2008 10:18:22
    NEXT COLLECTION TIME            : 04/05/2008 11:18:22
    TOTAL MODEL COUNT               : 1
    SUCCESSFUL MODEL COUNT          : 1
    LAST MODEL TIME                 : 04/05/2008 10:18:24
    LAST SUCCESSFUL MODEL TIME      : 04/05/2008 10:18:24
    NEXT MODEL TIME                 : 04/05/2008 22:18:24
    CHECK SPECIFIC PARAMETERS:
       COLLECTINT                   : 60
       MODELINT                     : 720
       COLLECTINACTIVE              : 1=ON
       DEBUG                        : 0=OFF
       STDDEV                       : 2
       EXCEPTIONMIN                 : 25
```

User override of IBM values:

The following shows keywords you can use to override check values on either a POLICY statement in the HZSPRMxx parmlib member or on a MODIFY command. This statement can be copied and modified to override the check defaults:

```
UPDATE CHECK(IBMPFA,PFA_LOGREC_ARRIVAL_RATE)
            ACTIVE
            SEVERITY(MEDIUM)
            INTERVAL(00:15)
      PARMS=('COLLECTINT(60)','MODELINT(720)','STDDEV(2)',
            'DEBUG(0)','COLLECTINACTIVE(1)','EXCEPTIONMIN(25)')
            DATE(20080330)
      REASON('LOGREC entry arrival rate is approaching the
             user defined standard deviation.')
```

If you change the COLLECTINT parameter or the INTERVAL parameter, the minutes set in the larger of the two parameters must elapse before the reports will be accurate.

Verbose support:

The check provides additional detail in verbose mode. You can put a check into verbose mode using the `UPDATE,filters,VERBOSE=ON` parameters on either the MODIFY command or in a POLICY statement in an HZSPRMxx parmlib member.

Debug support:

The DEBUG parameter in IBM Health Checker for z/OS is ignored by this check. Rather, the debug parameter is a PFA check specific parameter. For details, see “Understanding how to modify PFA checks” on page 79.

Reference:

For more information about PFA, see the topic on “Overview of Predictive Failure Analysis” on page 65.

Messages:

This check issues the following exception messages:

- AIRH110E

For additional message information, see the topics on:
- AIRH messages in *z/OS MVS System Messages, Vol 1 (ABA-AOM)*.
- AIR messages*z/OS MVS System Messages, Vol 1 (ABA-AOM)*.

SECLABEL recommended for MLS users:
SYSLOW

Output:
The LOGREC Arrival Rate Prediction Report:

```
       LOGREC Arrival Rate Prediction Report

Last successful model time      :  11/06/2008 17:32:44
Next model time                 :  11/06/2008 23:33:44
Model interval                  :  360
Last successful collection time :  11/06/2008 18:33:49
Next collection time            :  11/06/2008 19:34:49
Collection interval             :  60

                                   Key 0      Key 1-7     Key 8-15
                               __________  __________  __________
Arrivals in last
   collection interval:              1           0           2
Predicted rates based on...
     1 hour of data:                 1           0           1
   24 hours of data:                 0           0           1
     7 days of data:                 0           0           1
    30 days of data:                 0           0           1

Jobs having LOGREC arrivals in last collection interval:
   Job Name        ASID        Arrivals
   ________        ____        ________
   LOGREC08        0029               2
   LOGREC00        0027               1
```

Figure 23. LOGREC arrival rate prediction report

Note: In accordance with the IBM Health Checker for z/OS messaging guidelines, the largest generated output length for decimal variable values up to 2147483647 (X'7FFFFFF') is 10 bytes. When any PFA report value is greater than 2147483647, it displays using multiplier notation with a maximum of six characters. For example, if the report value is 2222233333444445555, PFA displays it as 1973P (2222233333444445555 ÷ 1125899906842) using the following multiplier notation:

Table 13. Multiplier notation used in values for PFA reports

Name	Sym	Size
Kilo	K	1,024
Mega	M	1,048,576
Giga	G	1,073,741,824
Tera	T	1,099,511,627,776
Peta	P	1,125,899,906,842

- **Last successful model time:** The date and time of the last successful model for this check. The predictions on this report were generated at that time.
- **Next model time:** The date and time of the next model. The next model will recalculate the predictions.

- **Model interval:** The value in the configured MODELINT parameter for this check. If PFA determines new prediction calculations are necessary, modeling can occur earlier.
- **Last successful collection time:** The date and time of the last successful data collection for this check.
- **Next collection time:** The date and time of the next collection.
- **Collection interval:** The value in the configured COLLECTINT parameter for this check.
- **Key column headings:** The program key of the issuer of the logrec entry.
- **Arrivals in last collection interval:** The value is one of the following:

  nnnn
  : The actual count of the logrec entries received in the last collection interval period.

  ****
  : The number of arrivals in the last collection interval is not available because the system generated a very large number of logrec entries during this period. For example:

  ```
                                Key 0      Key 1-7     Key 8-15
  Arrivals in last            __________  __________  __________
  collection interval:             ****        ****        ****
  ```

  If the arrival count is unavailable (****), the following message also appears on the report immediately following the predicted rates:

  ```
  **** Arrivals in last collection interval unavailable
   due to high level of system activity.
  ```

  In addition, if the check was unable to report all the logrec entries found in the last collection interval, it can also display the following message before the job listings:

  ```
  Job list may not include all arrivals in last collection
  interval due to high level of system activity.
  ```

  This message typically appears when the arrival count is unavailable (with ****) however, it can also appear when a logrec entry is skipped because of a high level of system activity
- **Predicted rates based on...:** The logrec entry counts based on different ranges of historical data. If the required amount of data is not available, the line in this report is not generated. For example, if PFA has been running for a week, it has not yet collected 30 days of historical data and therefore the "30 days of data" line is not generated in the report.
- **Jobs having LOGREC arrivals in last collection interval:** The jobs that contributed to the logrec arrivals in the last collection interval.
- **Job name:** The name of the job that had logrec arrivals in the last collection interval.
- **ASID:** The ASID of the job that had logrec arrivals in the last collection interval.
- **Arrivals:** The actual count of the logrec arrivals for the job.

Directories:

When you install the PFA_LOGREC_ARRIVAL_RATE check, the shell script creates the following directories:

pfa_directory
: This directory contains all the PFA checks and is pointed to by the home directory of the started task. The following files only contain data if messages are generated by the JVM:

- java.stderr (generated by JVM)
- java.stdout (generated by JVM).

pfa_directory/PFA_LOGREC_ARRIVAL_RATE/data/
The directory for logrec arrival rate that holds data and modeling results.

Guideline: If the use of the z/OS image is radically different after an IPL (for instance, the change from a test system to a production system), delete the files from *PFA_LOGREC_ARRIVAL_RATE/data* directory to enable the check to collect the most accurate modeling information. After the *PFA_LOGREC_ARRIVAL_RATE* check finds a problem and you correct it, delete the files in *PFA_LOGREC_ARRIVAL_RATE/data* to prevent that previous error from hiding any new errors.

Results files:

- systemName.1hr.prediction - This file is generated by the modeling code for the predictions made for one hour of historical data. It contains the list of program keys with their corresponding predictions and additional information required for PFA processing.
- systemName.24hr.prediction - This file is generated by the modeling code for the predictions made for 24 hours of historical data. It contains the list of program keys with their corresponding predictions and additional information required for PFA processing.
- systemName.7day.prediction - This file is generated by the modeling code for the predictions made for seven days of historical data. It contains the list of program keys with their corresponding predictions and additional information required for PFA processing.
- systemName.30day.prediction - This file is generated by the modeling code for the predictions made for 30 days of historical data. It contains the list of program keys with their corresponding predictions and additional information required for PFA processing.
- systemName.1hr.prediction.html - This file contains an .html report version of the data found in the systemName.1hr.prediction file.
- systemName.24hr.prediction.html - This file contains an .html report version of the data found in the systemName.24hr.prediction file.
- systemName.7day.prediction.html - This file contains an .html report version of the data found in the systemName.7day.prediction file.
- systemName.30day.prediction.html - This file contains an .html report version of the data found in the systemName.30day.prediction file.
- systemName.prediction.stddev - The file generated by the modeling code to list the standard deviation of the predictions across the time ranges for each program key.

The values for the logrec arrival prediction file in .html (prediction.html) files are as follows:

- **Program Key:** The program key when the error described by the logrec entry occurred.
- **Instance Count:** The number of records with this program key that were factored into the prediction model.
- **Current Logrec Arrivals:** The number of logrec arrivals for this program key in the last collection interval included in this model.

- **Prediction Look Forward Seconds:** The number of seconds the prediction should project into the future.
- **Predicted Logrec Arrivals:** The predicted logrec arrivals for this program key.

Intermediate files:

- systemNameLAR.OUT - The data collection file.
- systemName.1hr.lardata - The file used as input to modeling code. It contains one hour of historical data.
- systemName.24hr.lardata - The file used as input to modeling code. It contains 24 hours of historical data.
- systemName.7day.lardata - The file used as input to modeling code. It contains seven days of historical data.
- systemName.30day.lardata - The file is used as input to modeling code. It contains 30 days of historical data.
- systemName.lardata -- The file is used as input to the modeling to track if enough data is available to model.

This directory holds the following log files. Additional information is written to these log files when DEBUG(1).

- systemName.1hr.cart.log - The log file generated by modeling code with details about code execution while one hour of historical data was being modeled.
- systemName.24hr.cart.log - The log file generated by modeling code with details about code execution while 24 hours of historical data was being modeled.
- systemName.7day.cart.log - The log file generated by modeling code with details about code execution while seven days of historical data was being modeled.
- systemName.30day.cart.log - The log file generated by modeling code with details about code execution while 30 days of historical data was being modeled.
- systemName.buildLar.log - The log file generated by intermediate code that builds the files that are input to modeling with details about code execution.
- systemName.launcher.log - The log file generated by launcher code.
- systemName.1hr.tree - This file is generated by the modeling code. It contains information about the model tree which was built based on the last one hour of collected data.
- systemName.24hr.tree - This file is generated by the modeling code. It contains information about the model tree which was built based on the last 24 hours of collected data.
- systemName.7day.tree - This file is generated by the modeling code. It contains information about the model tree which was built based on the last seven days of collected data.
- systemName.30day.tree - This file is generated by the modeling code. It contains information about the model tree which was built based on the last 30 days of collected data.
- systemName.1hr.holes - The file is used to track gaps in data for a one hour period. Gaps are caused by stopping PFA or by an IPL.

- systemName.24hr.holes - The file is used to track gaps in the data for a 24 hour time period. Gaps are caused by stopping PFA or by an IPL.
- systemName.7day.holes - The file is used to track gaps in the data for the seven day time period. Gaps are caused by stopping PFA or by an IPL.
- systemName.30day.holes - The file is used to track gaps in the data for the 30 day time period. Gaps are caused by stopping PFA or by an IPL.
- systemNameCONFIG.LOG - The log file containing the configuration history for the last 30 days for this check.
- systemNameCOLLECT.LOG - The log file used during data collection.
- systemNameMODEL.LOG - The log file used during portions of the modeling phase.
- systemNameRUN.LOG - The log file used when the check runs.

pfa_directory/PFA_LOGREC_ARRIVAL_RATE/EXC_timestamp

This directory contains all the relevant data for investigating exceptions issued by this check at the timestamp provided in the directory name. PFA keeps directories only for the last 30 exceptions. Therefore at each exception, if more than 30 exception directories exist, the oldest directory is deleted so that only 30 exceptions remain after the latest exception is added.

- systemNameREPORT.LOG - The log file containing the same contents as the IBM Health Checker for z/OS report for this exception as well as other diagnostic information issued during report generation.

pfa_directory/PFA_LOGREC_ARRIVAL_RATE/config

This directory contains configuration information for the check.

- EXCLUDED_JOBS - The file containing the list of excluded jobs for this check.

PFA_MESSAGE_ARRIVAL_RATE

Description:

This check is determining when an LPAR is damaged by checking the arrival rate of abnormal messages per number of processor millisecond used. If the arrival rate is too high, it might indicate a damaged address space or partition. If the arrival rate is too low, it might indicate a hung address space or partition.

To avoid skewing the message arrival rate, PFA ignores the first hour of message data after IPL and the last hour of message data before shutdown. In addition, PFA attempts to track the same persistent jobs that it tracked before IPL or PFA restart if the same persistent jobs are still active. (The same persistent jobs must still be active for PFA to track and there ten jobs must have previously tracked.)

This check is not designed to detect performance problems that are caused by insufficient resources, faulty WLM policy, or spikes in work. However, it might help to determine if a performance problem detected by a performance monitor or WLM is caused by a damaged system.

The message arrival rate check issues an exception using four types of comparisons, which are described in more detail in the next section:

- top persistent jobs
- other persistent
- non-persistent jobs
- total system

After PFA issues an exception, the next comparison type is not performed. The CONSOLE address space and any jobs that match the job and system combinations defined in the config/EXCLUDED_JOBS file that have been read for PFA processing are not included in the processing for any of the four types of comparisons. By default, an EXCLUDED_JOBS file containing the all address spaces that match `JES*` on all systems is created during installation. Therefore, if you have not made any modifications to the EXCLUDED_JOBS file, these jobs will be excluded. See “Using and configuring supervised learning” on page 86 for more information.

Top persistent jobs

PFA tracks the top persistent jobs individually. Jobs are considered persistent if they start within an hour after IPL. PFA determines which jobs to track individually based on the following criteria:

- If PFA previously ran on this system and the same 10 jobs that were previously tracked are active, PFA tracks the same jobs.
- If PFA did not previously run on this system or the same jobs previously tracked are not all active, PFA collects data for a period of time to use in determining which jobs have the highest arrival rates. After this time passes, PFA individually tracks the jobs that have the highest arrival rates for that period.
  - During the first hour after IPL and during the time PFA is determining the jobs to track individually, normal data collection and modeling are suspended.
  - Changing the COLLECTINT or the MODELINT parameters during these times is allowed, but the changes are not used until after these times have passed.
  - Next collection and model times change automatically during these times to reflect the most accurate times known at each phase of the initial processing.

This top persistent jobs comparison is performed to determine if the message rate is higher than expected or lower than expected.

Other persistent jobs

The persistent jobs that PFA does not track individually are the other persistent jobs. PFA generates the predictions using the totals for this group. When determining if the message arrival rate is higher than expected, PFA performs the comparisons individually using a mathematical formula. When determining if the message rate is lower than expected, PFA performs the comparisons using the totals for the group.

Non-persistent jobs

The jobs that start over an hour after IPL are the non-persistent jobs. PFA performs the predictions and the comparisons using the totals for this group. This type of comparison is only used to determine if the message rate is higher than expected.

Total system
This group includes all jobs. PFA performs the predictions and the comparisons for the entire system to determine if the message rate is higher than expected or lower than expected.

Reason for check:
The objective of this check is to determine if an LPAR is damaged and if an address space or partition is hung by checking the arrival rate of abnormal messages per number of CPU millisecond used.

Best practice:
If PFA detects an unexpectedly high amount of messages, the best practice is to analyze the messages being sent by the address spaces identified on the report by examining the system log to determine what is causing this burst of message activity. Establish which messages were issued around the time of the activity and review the message details. Follow the directions provided by the message to continue to diagnose and fix the problem.

If PFA detects an unexpectedly low number of messages, examine the report in SDSF for details about why the exception was issued. Use the Runtime Diagnostics output in the report to assist you in diagnosing and fixing the problem. For more information about Runtime Diagnostics see Chapter 4, “Runtime Diagnostics,” on page 35.

z/OS releases the check applies to:
z/OS V1R11 and later.

Type of check:
Remote

Parameters accepted:
Yes, as follows:

Table 14. PFA_MESSAGE_ARRIVAL_RATE check parameters

Parameter name	Default value	Minimum Value	Maximum Value	Description
collectint	15 Minutes	15	360	This parameter determines how often (in minutes) to run the data collector that retrieves the current message arrival rate.
modelint	720 Minutes	60	1440	This parameter determines how often (in minutes) you want the system to analyze the data and construct a new message arrival rate model or prediction. By default, PFA analyzes the data and constructs a new model every “default value” minutes. The model interval must be at least four times larger than the collection interval. Note that, even when you set a value larger than 360, PFA performs the first model at 360 minutes (6 hours). By default, PFA analyzes the data and constructs a new model every 720 minutes (12 hours).

Table 14. PFA_MESSAGE_ARRIVAL_RATE check parameters (continued)

Parameter name	Default value	Minimum Value	Maximum Value	Description
stddev	10	2	100	This parameter is used to specify how much variance is allowed between the actual message arrival rate per amount of CPU and the expected message arrival rate. It is used when determining if the actual message arrival rate has increased beyond the allowable upper limit. It also determines how much variance is allowed across the time range predictions. If you set the STDDEV parameter to a smaller value, an exception is issued if the actual message arrival rate is closer to the expected message arrival rate and the predictions across the time ranges are consistent. If you set the STDDEV parameter to a larger value, an exception is issued if the actual message arrival rate is significantly greater than the expected message arrival rate even if the predictions across the different time ranges are inconsistent.
collectinactive	1 (on)	0 (off)	1 (on)	Defines whether data will be collected and modeled even if the check is not eligible to run, not ACTIVE(ENABLED), in IBM Health Checker for z/OS.
trackedmin	3	0	1000	This parameter defines the minimum message arrival rate required for a persistent job in order for it to be considered a top persistent job that should be tracked individually.
exceptionmin	1	0	1000	This parameter is used when determining if an exception should be issued for an unexpectedly high message arrival rate. For tracked jobs and other persistent jobs, this parameter defines the minimum message arrival rate and the minimum predicted message arrival rate required to cause a too high exception. For non-persistent jobs and the total system comparisons, this parameter defines the minimum message arrival rate required to cause a too high exception.
checklow	1 (on)	0 (off)	1	Defines whether Runtime Diagnostics is run to validate that the absence of messages is caused by a problem. If this value is off, exceptions are not issued for conditions in which the message arrival rate is unexpectedly low.
stddevlow	4	2	100	This parameter is used to specify how much variance is allowed between the actual message arrival rate per amount of CPU and the expected message arrival rate when determining if the actual rate is unexpectedly low. • If you set the STDDEVLOW parameter to a smaller value, an exception is issued when the actual message arrival rate is closer to the expected message arrival rate. • If you set the STDDEVLOW parameter to a larger value, an exception is issued when the actual message arrival rate is significantly lower than the expected message arrival rate.

Table 14. PFA_MESSAGE_ARRIVAL_RATE check parameters (continued)

Parameter name	Default value	Minimum Value	Maximum Value	Description
limitlow	3	1	100	This parameter defines the maximum message arrival rate allowed when issuing an exception for an unexpectedly low number of messages.
debug	0 (off)	0 (off)	1 (on)	This parameter (an integer of 0 or 1) is used at the direction of IBM service to generate additional diagnostic information for the IBM Support Center. This debug parameter is used in place of the IBM Health Checker for z/OS policy. The default is off (0).

To determine the status of the message arrival rate check, issue `f pfa,display,check(pfa_message_arrival_rate),detail`. For the command example and more details, see . The following example shows the output written to message AIR018I in SDSF:

```
AIR018I 02:22:54 PFA CHECK DETAIL

CHECK NAME:  PFA_MESSAGE_ARRIVAL_RATE
    ACTIVE                         : YES
    TOTAL COLLECTION COUNT         : 5
    SUCCESSFUL COLLECTION COUNT    : 5
    LAST COLLECTION TIME           : 02/05/2009 10:18:22
    LAST SUCCESSFUL COLLECTION TIME: 02/05/2009 10:18:22
    NEXT COLLECTION TIME           : 02/05/2009 10:33:22
    TOTAL MODEL COUNT              : 1
    SUCCESSFUL MODEL COUNT         : 1
    LAST MODEL TIME                : 02/05/2009 10:18:24
    LAST SUCCESSFUL MODEL TIME     : 02/05/2009 10:18:24
    NEXT MODEL TIME                : 02/05/2009 22:18:24
    CHECK SPECIFIC PARAMETERS:
       COLLECTINT                  : 15
       MODELINT                    : 720
       COLLECTINACTIVE             : 1=ON
       DEBUG                       : 0=OFF
       STDDEV                      : 10
       TRACKEDMIN                  : 3
       EXCEPTIONMIN                : 1
       CHECKLOW                    : 1=ON
       STDDEVLOW                   : 4
       LIMITLOW                    : 3
       EXCLUDED JOBS:
      NAME     SYSTEM   DATE ADDED       REASON ADDED
      JES      *        2010/03/31 00:00 Exclude JES* jobs on ALL.
```

User override of IBM values:

The following shows keywords you can use to override check values on either a POLICY statement in the HZSPRMxx parmlib member or on a MODIFY command. This statement can be copied and modified to override the check defaults:

```
UPDATE CHECK(IBMPFA,PFA_MESSAGE_ARRIVAL_RATE)
            ACTIVE
            SEVERITY(MEDIUM)
            INTERVAL(ONETIME)
      PARMS=('COLLECTINT(15)','MODELINT(720)','STDDEV(10)','DEBUG(0)',
            'COLLECTINACTIVE(1)','EXCEPTIONMIN(1)','TRACKEDMIN(3)'
            'CHECKLOW(1)','STDDEVLOW(4)','LIMITLOW(3)')
            DATE(20080330)
      REASON('The message arrival rate is abnormal which
      can indicate a system that is damaged.')
```

The message arrival rate check is designed to run automatically after every data collection. Do not change the INTERVAL parameter.

Verbose support:
The check provides additional detail in verbose mode. You can put a check into verbose mode using the UPDATE,filters,VERBOSE=ON parameters on either the MODIFY command or in a POLICY statement in an HZSPRMxx parmlib member.

Debug support:
The DEBUG parameter in IBM Health Checker for z/OS is ignored by this check. Rather, the debug parameter is a PFA check specific parameter. For details, see "Understanding how to modify PFA checks" on page 79.

Reference:
For more information about PFA, see the topic on "Overview of Predictive Failure Analysis" on page 65.

Messages:
The output is a message arrival rate prediction report that corresponds to the message issued. One of the following reports is generated:

- AIRH152E or AIRH153E – total system exception report
- AIRH165E or AIRH206E – tracked jobs exception report
- AIRH166E or AIRH207E – other persistent jobs exception report
- AIRH169E – other non-persistent jobs exception report

For additional message information, see the topics on:

- AIRH messages in *z/OS MVS System Messages, Vol 1 (ABA-AOM)*.
- AIR messages*z/OS MVS System Messages, Vol 1 (ABA-AOM)*.

SECLABEL recommended for MLS users:
SYSLOW

Output:

The output is a variation of the message arrival rate prediction report. The values found in the message arrival prediction report are as follows:
Tracked top persistent jobs exception report: PFA issues the message arrival rate tracked jobs exception report when any one or more tracked, persistent jobs cause an exception. The exception can be caused by a higher than expected message arrival rate or a lower than expected message arrival rate. Only the tracked jobs that caused the exception are included in the list of jobs on the report. If the report was generated due to a lower than expected message arrival rate, the report includes Runtime Diagnostics output, which can help you diagnose the behavior. The following example is the message arrival rate tracked jobs exception report for jobs that had a higher than expected message arrival rate (for AIRH165E):

```
 Message Arrival Rate Prediction Report

Last successful model time      :  01/27/2009 17:08:01
Next model time                 :  01/27/2009 23:08:01
Model interval                  :  360
Last successful collection time :  01/27/2009 17:41:38
Next collection time            :  01/27/2009 17:56:38
Collection interval             :  15

Persistent address spaces with high rates:

                                           Predicted Message
                      Message                Arrival Rate
  Job                 Arrival
  Name     ASID          Rate        1 Hour       24 Hour         7 Day
  ________ ____  ____________  ____________  ____________  ____________
  TRACKED1 001D         75.63         23.88         22.82         15.82
  TRACKED2 0028         43.52          0.34         11.11         12.11
  TRACKED3 0029         11.00         12.43          2.36          8.36
```

Figure 24. Message arrival rate prediction report: tracked jobs higher than expected

This example is the message arrival rate tracked jobs exception report for jobs that had a lower than expected message arrival rate (for AIH206E):

```
 Message Arrival Rate Prediction Report

Last successful model time      : 01/27/2009 17:08:01
Next model time                 : 01/27/2009 23:08:01
Model interval                  : 360
Last successful collection time : 01/27/2009 17:41:38
Next collection time            : 01/27/2009 17:56:38
Collection interval             : 15

Persistent address spaces with low rates:
                                           Predicted Message
                   Message                   Arrival Rate
Job                Arrival
Name     ASID         Rate       1 Hour      24 Hour        7 Day

________ ____ ____________ ____________ ____________ ____________
JOBS4    001F         1.17        23.88        22.82        15.82
JOBS5    002D         2.01         8.34        11.11        12.11

Runtime Diagnostics Output:
Runtime Diagnostics detected a problem in job: JOBS4
  EVENT 06: HIGH - HIGHCPU - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID CPU RATE: 96% ASID: 0027 JOBNAME: JOBS4
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MAY BE LOOPING.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
  EVENT 07: HIGH - LOOP - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID: 0027 JOBNAME: JOBS4 TCB: 004E6850
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE APPEARS TO BE IN A LOOP.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
Runtime Diagnostics detected a problem in job: JOBS5
  EVENT 03: HIGH - HIGHCPU - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID CPU RATE: 96% ASID: 0027 JOBNAME: JOBS5
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MAY BE LOOPING.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
  EVENT 04: HIGH - LOOP - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID: 0027 JOBNAME: JOBS5 TCB: 004E6850
STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE APPEARS TO BE IN A LOOP.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
```

Figure 25. Message arrival rate prediction report: tracked jobs lower than expected

Other persistent jobs exception report: PFA issues the message arrival rate other persistent jobs exception report when a comparison of a persistent job (that is not being individually tracked) causes an exception when compared to the totals of the other persistent jobs. The exception can be caused by a higher than expected message arrival rate or a lower than expected message arrival rate. The predictions listed on this report are the predicted rates for the total other persistent jobs group. The list of jobs is only those persistent jobs (not tracked individually) that have a problem and is only generated for a higher than expected message arrival rate. No predictions are given for these jobs because PFA does not model individual predictions for jobs that are not tracked individually. If there is more than one job with the same name, four asterisks **** are printed for the ASID in the report. If the report was generated due to a lower than expected message arrival rate, the report will include Runtime Diagnostics output which can help diagnose the behavior. The following example is the message arrival rate exception report for other persistent jobs that had an unexpectedly high message arrival rate (for AIRH166E):

```
Message Arrival Rate Prediction Report

Last successful model time      :  01/27/2009 17:08:01
Next model time                 :  01/27/2009 23:08:01
Model interval                  :  360
Last successful collection time :  01/27/2009 17:41:38
Next collection time            :  01/27/2009 17:56:38
Collection interval             :  15

Other persistent jobs group:
Prediction based on 1 hour of data   : 20.27
Prediction based on 24 hours of data : 27.98
Prediction based on 7 days of data   : 31.22

Persistent address spaces with high rates:

                      Message
  Job                 Arrival
  Name     ASID          Rate
  ________ ____  ____________
  PERS1    001E         83.22
  PERS2    0038         75.52
  PERS3    0039         47.47
```

Figure 26. Message arrival rate prediction report: other persistent jobs with high arrival rate

Note: In the “other persistent jobs” and “total system” categories, when using Runtime Diagnostics, it is possible to see data for jobs previously defined to the excluded jobs list because PFA must return any potential problem activity on the system identified by Runtime Diagnostics.

The following example is the message arrival rate exception report for other persistent jobs that had an unexpectedly low message arrival rate (for AIRH207E):

```
 Message Arrival Rate Prediction Report

Last successful model time      : 01/27/2009 17:08:01
Next model time                 : 01/27/2009 23:08:01
Model interval                  : 360
Last successful collection time : 01/27/2009 17:41:38
Next collection time            : 01/27/2009 17:56:38
Collection interval             : 15

Other persistent jobs group:
Prediction based on 1 hour of data   : 20.27
Prediction based on 24 hours of data : 27.98
Prediction based on 7 days of data   : 31.22

Runtime Diagnostics Output:

  EVENT 01: HIGH - HIGHCPU - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID CPU RATE: 96% ASID: 0027 JOBNAME: PERS4
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MAY BE LOOPING.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
  EVENT 02: HIGH - LOOP - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID: 0027 JOBNAME: PERS4 TCB: 004E6850
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE APPEARS TO BE IN A LOOP.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
```

Figure 27. Message arrival rate prediction report: other persistent jobs with low arrival rate

Non-persistent jobs exception report: PFA issues the message arrival rate non-persistent jobs exception report when the non-persistent jobs as a group can cause an exception. This exception is only issued for a higher than expected message arrival rate. The message arrival rate and predictions listed on this report are calculated for the total non-persistent jobs group. The list of jobs contains only three non-persistent jobs that have high arrival counts. No predictions are given for these jobs because PFA does not model individual predictions for jobs that are not tracked individually. The following example is the message arrival rate non-persistent jobs exception report (for AIRH169E):

```
Message Arrival Rate Prediction Report

Last successful model time      :  01/27/2009 17:08:01
Next model time                 :  01/27/2009 23:08:01
Model interval                  :  360
Last successful collection time :  01/27/2009 17:41:38
Next collection time            :  01/27/2009 17:56:38
Collection interval             :  15

Non-persistent jobs group:
 Message arrival rate
   in last collection interval        : 65.49
 Prediction based on 1 hour of data   : 20.27
 Prediction based on 24 hours of data : 27.98
 Prediction based on 7 days of data   : 31.22

Address spaces with high arrivals:

                      Message
  Job                 Arrival
  Name     ASID        Counts

  ________ ____  ____________
  NONPERS1 001F            83
  NONPERS2 0048            52
  NONPERS3 0049            47
```

Figure 28. Message arrival rate prediction report: non-persistent jobs higher than expected

No problem and total system exception report: PFA issues the message arrival rate system report when no exception is issued or when the total message arrival rate exception is issued. When there is no problem or when an exception occurs due to a higher than expected message arrival rate, the list of jobs contains all of the jobs being tracked individually. The Runtime Diagnostics section is written in the report when the exception is issued due to a lower than expected message arrival rate. When the report is issued due to a lower than expected message arrival rate, the list of tracked jobs will not be printed on the report. The following example is the message arrival rate no problem total system report issued due to a higher than expected message arrival rate (AIRH152E):

```
            Message Arrival Rate Prediction Report

Last successful model time      :  01/27/2009 17:08:01
Next model time                 :  01/27/2009 23:08:01
Model interval                  :  360
Last successful collection time :  01/27/2009 17:41:38
Next collection time            :  01/27/2009 17:56:38
Collection interval             :  15

Message arrival rate
at last collection interval          :       83.52
Prediction based on 1 hour of data   :       98.27
Prediction based on 24 hours of data :       85.98
Prediction based on 7 days of data   :      100.22

Top persistent users:

                                               Predicted Message
                      Message                    Arrival Rate
  Job                 Arrival
  Name     ASID          Rate         1 Hour       24 Hour         7 Day
  ________ ____  ____________  ____________  ____________  ____________
  JOB1     001D         58.00         23.88         22.82         15.82
  JOB2     0028         11.00          0.34         11.11         12.11
  JOB3     0029         11.00         12.43          2.36          8.36
  ...
```

Figure 29. Message arrival rate prediction report: total system higher than expected

This example is the message arrival rate total system exception report issued due to a lower than expected message arrival rate (for AIRH153E) with Runtime Diagnostic output:

```
Message Arrival Rate Prediction Report

Last successful model time      : 01/27/2009 17:08:01
Next model time                 : 01/27/2009 23:08:01
Model interval                  : 360
Last successful collection time : 01/27/2009 17:41:38
Next collection time            : 01/27/2009 17:56:38
Collection interval             : 15

Message arrival rate
  in last collection interval        : 2.02
Prediction based on 1 hour of data   : 98.27
Prediction based on 24 hours of data : 85.98
Prediction based on 7 days of data   : 100.22

Runtime Diagnostics Output:

  EVENT 01: HIGH - CF - SYSTEM: SY1 2009/06/12 - 13:23:25
  IXL013I IXLCONN REQUEST FOR STRUCTURE SYSZWLM_WORKUNIT FAILED.
  JOBNAME: WLM ASID: 000A CONNECTOR NAME: #SY1
  IXLCONN RETURN CODE: 0000000C, REASON CODE: 02010C05
  STRUCTURE NOT DEFINED IN THE CFRM ACTIVE POLICY
  CONADIAG0: 00000002
----------------------------------------------------------------------
  EVENT 02: HIGH - XCF - SYSTEM: SY1 2009/06/12 - 13:23:55
  IXC467I STOPPING PATHIN STRUCTURE IXCT_SIGNAL
  RSN: START REQUEST FAILED
----------------------------------------------------------------------
  EVENT 03: HIGH - XCF - SYSTEM: SY1 2009/06/12 - 13:23:58
  IXC467I STOPPING PATHIN STRUCTURE IXCT_SIGNAL
  RSN: START REQUEST FAILED
----------------------------------------------------------------------
  EVENT 04: HIGH - CF - SYSTEM: SY1 2009/06/12 - 13:24:02
  IXL013I IXLCONN REQUEST FOR STRUCTURE IXCT_SIGNAL FAILED.
  JOBNAME: XCFAS ASID: 0006 CONNECTOR NAME: SIGPATH_01000008
  IXLCONN RETURN CODE: 0000000C, REASON CODE: 02010C05
  STRUCTURE NOT DEFINED IN THE CFRM ACTIVE POLICY
  CONADIAG0: 00000002
----------------------------------------------------------------------
  EVENT 05: HIGH - XCF - SYSTEM: SY1 2009/06/12 - 13:24:02
  IXC467I STOPPING PATHIN STRUCTURE IXCT_SIGNAL
  RSN: START REQUEST FAILED
----------------------------------------------------------------------
  EVENT 06: HIGH - HIGHCPU - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID CPU RATE: 96% ASID: 0027 JOBNAME: DAVIDZ
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MAY BE LOOPING.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
  EVENT 07: HIGH - LOOP - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID: 0027 JOBNAME: DAVIDZ TCB: 004E6850
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE APPEARS TO BE IN A LOOP.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
```

Figure 30. Message arrival rate prediction report: total system lower than expected

Note: In accordance with the IBM Health Checker for z/OS messaging guidelines, the largest generated output length for decimal variable values up to 2147483647 (X'7FFFFFF') is 10 bytes. When any PFA report value is greater

than 2147483647, it displays using multiplier notation with a maximum of six characters. For example, if the report value is 2222233333444445555, PFA displays it as 1973P (2222233333444445555 ÷ 1125899906842) using the following multiplier notation:

Table 15. Multiplier notation used in values for PFA reports

Name	Sym	Size
Kilo	K	1,024
Mega	M	1,048,576
Giga	G	1,073,741,824
Tera	T	1,099,511,627,776
Peta	P	1,125,899,906,842

The following fields apply to all reports:

- **Last successful model time:** The date and time of the last successful model for this check. The predictions on this report were generated at that time.
- **Next model time:** The date and time of the next model. The next model will recalculate the predictions.
- **Model interval:** The value in the configured MODELINT parameter for this check. If PFA determines new prediction calculations are necessary, modeling can occur earlier.
- **Last successful collection time:** The date and time of the last successful data collection for this check.
- **Next collection time:** The date and time of the next collection.
- **Collection interval:** The value in the configured COLLECTINT parameter for this check.
- **Message arrival rate in last collection interval:** The actual message arrival rate in the last collection interval where the rate is defined to be the number of messages divided by the CPU milliseconds.
- **Predicted rates based on...:** The message arrival rates based on one hour, 24 hours, and seven days. If no prediction is available for a given time range, the line is not printed. For example, if the check has been running for two days, seven days of data is not available and the “Prediction based on 7 days of data” line is not printed.
- **Runtime Diagnostics Output:** Runtime Diagnostics event records to assist you in diagnosing and fixing the problem. See the topic on “Runtime Diagnostics symptoms” on page 39 in Chapter 4, “Runtime Diagnostics,” on page 35.
- **Job Name:** The name of the job that has message arrivals in the last collection interval.
- **ASID:** The ASID for the job that has message arrivals in the last collection interval.
- **Message Arrival Rate:** The current message arrival rate for the persistent job.
- **Message Arrival Counts:** The message arrival count for the non-persistent job.

  Note: The “Message Arrival Count” field is unique to the non-persistent jobs exception report.
- **Predicted Message Arrival Rate:** The predicted message arrival rate based on one hour, 24 hours, and seven days of data. If PFA did not previously

run on this system or the same jobs previously tracked are not all active, there will not be enough data for a time range until that amount of time has passed. Also, gaps in the data caused by stopping PFA or by an IPL might cause the time range to not have enough data available. After the check collects enough data for any time range, predictions are made again for that time range. If there is not enough data for a time range, INELIGIBLE is printed and comparisons are not made for that time range.

Directories

Note: The content and names for these files and directories are subject to change and cannot be used as programming interfaces; these files are documented only to provide help in diagnosing problems with PFA.

pfa_directory

This directory contains all the PFA checks and is pointed to by the home directory of the started task. The following files only contain data if messages are generated by the JVM:

- java.stderr (generated by JVM)
- java.stdout (generated by JVM)

pfa_directory/PFA_MESSAGE_ARRIVAL_RATE/data

The directory for message arrival rate that holds data and modeling results.

Guideline: If the use of the z/OS image is radically different after an IPL (for instance, the change from a test system to a production system), delete the files in the *PFA_MESSAGE_ARRIVAL_RATE/data* directory to enable the check to collect the most accurate modeling information.

Results files

- systemName.1hr.prediction - This file is generated by the modeling code for the predictions made for one hour of historical data. It contains predictions for each of the tracked address spaces, the other persistent category, the non-persistent category, and the total system category. It also contains additional information required for PFA processing.
- systemName.24hr.prediction - This file is generated by the modeling code for the predictions made for 24 hours of historical data. It contains predictions for each of the tracked address spaces, the other persistent category, the non-persistent category, and the total system category. It also contains additional information required for PFA processing.
- systemName.7day.prediction - This file is generated by the modeling code for the predictions made for seven days of historical data. It contains predictions for each of the tracked address spaces, the other persistent category, the non-persistent category, and the total system category. It also contains additional information required for PFA processing.
- systemName.1hr.prediction.html - This file lists the persistent address spaces in an .html report format for the predictions made for one hour of historical data.
- systemName.24hr.prediction.html - This file lists the persistent address spaces in an .html report format for the predictions made for 24 hours of historical data.

- systemName.7day.prediction.html - This file lists the persistent address spaces in an .html report format for the predictions made for seven days of historical data.
- systemName.prediction.stddev - The file generated by the modeling code to list the standard deviation of the predictions across the time ranges for each address space.

Data store files:

- systemNameMAR.OUT - The data collection file.

Intermediate files:

- systemName.mardata - The file is used as input to the modeling to track if enough data is available to model.
- systemName.1hr.mardata - The file used as input to modeling code. It contains one hour of historical data.
- systemName.24hr.mardata - The file used as input to modeling code. It contains 24 hours of historical data.
- systemName.7day.mardata - The file used as input to modeling code. It contains seven days of historical data.
- systemName.1hr.holes - The file is used to track gaps in the data for a one hour time period. Gaps are caused by stopping PFA or by an IPL.
- systemName.24hr.holes - The file is used to track gaps in the data for a 24 hour time period. Gaps are caused by stopping PFA or by an IPL.
- systemName.7day.holes - The file is used to track gaps in the data for a seven day time period. Gaps are caused by stopping PFA or by an IPL.

This directory holds the following log files. Additional information is written to these log files when DEBUG(1).

- systemName.1hr.cart.log - The log file generated by modeling code with details about code execution while one hour of historical data was being modeled.
- systemName.24hr.cart.log - The log file generated by modeling code with details about code execution while 24 hours of historical data was being modeled.
- systemName.7day.cart.log - The log file generated by modeling code with details about code execution while seven days of historical data was being modeled.
- systemName.1hr.tree - This file is generated by the modeling code. It contains information about the model tree which was built based on the last one hour of collected data.
- systemName.24hr.tree - This file is generated by the modeling code. It contains information about the model tree which was built based on the last 24 hours of collected data.
- systemName.7day.tree - This file is generated by the modeling code. It contains information about the model tree which was built based on the last seven days of collected data.
- systemName.buildMar.log - The log file generated by intermediate code that builds the files that are input to modeling with details about code execution.
- systemName.launcher.log - The log file generated by launcher code.

- systemNameCONFIG.LOG - The log file containing the configuration history for the last 30 days for this check.
- systemNameCOLLECT.LOG - The log file used during data collection.
- systemNameMODEL.LOG - The log file used during portions of the modeling phase.
- systemNameRUN.LOG - The log file used when the check runs.

pfa_directory/PFA_MESSAGE_ARRIVAL_RATE/EXC_timestamp
This directory contains all the relevant data for investigating exceptions issued by this check at the timestamp provided in the directory name. PFA keeps directories only for the last 30 exceptions. Therefore at each exception, if more than 30 exception directories exist, the oldest directory is deleted so that only 30 exceptions remain after the latest exception is added.

- systemNameREPORT.LOG - The log file containing the same contents as the IBM Health Checker for z/OS report for this exception as well as other diagnostic information issued during report generation.

pfa_directory/PFA_MESSAGE_ARRIVAL_RATE/config
This directory contains the configuration files for the check.

- EXCLUDED_JOBS - The file containing the list of excluded jobs for this check.

  Note: When using Runtime Diagnostics, it is possible to see data for jobs previously defined to the excluded jobs list in the “other persistent jobs” and “total system” categories because PFA must return any potential problem activity on the system identified by Runtime Diagnostics.

PFA_SMF_ARRIVAL_RATE

Description:

When SMF is active on the system, this check determines if there is an abnormal SMF arrival rate per CPU millisecond. PFA examines only the SMF record types you set the SMFPRMxx parmlib member to generate. When the number of SMF SMF records written per CPU millisecond is unusually high or low, PFA can provide an early indication of a problem and potentially prevent damage to an LPAR.

To avoid skewing the SMF arrival rate, PFA ignores the first hour of SMF data after IPL and the last hour of SMF data prior to shutdown. In addition, PFA attempts to track the same persistent jobs that it tracked prior to IPL or PFA restart if the same persistent jobs are still active. (The same persistent jobs must still be active for PFA to track the same jobs and there must have been ten jobs tracked previously.)

This check is not designed to detect performance problems caused by insufficient resources, faulty WLM policy, or spikes in work. However, it might help to determine if a performance problem detected by a performance monitor or WLM is caused by a damaged system.

Guideline: If you modify the SMF record types in the SMFPRMxx parmlib member, delete the files in the *PFA_SMF_ARRIVAL_RATE/data* directory to ensure PFA is collecting relevant information.

The SMF arrival rate check issues an exception using the following four types of comparisons. After the check issues an exception, it does not perform the next comparison type. All jobs are included in this group except those that match a job specified in the /config/EXCLUDED_JOBS file for this check.

Note: If SMF is restarted, the data previously collected will be automatically discarded and the check will enter the phase where it collects data for a period of time to use in determining the jobs to track. This processing is done to reduce false positives and to ensure that data that was potentially collected using different SMF record types is not used by PFA after SMF is restarted.

1. **Top persistent jobs:** The SMF arrival rate check tracks the top persistent jobs individually. Jobs are considered persistent when they start within an hour after IPL. PFA determines which jobs to track individually based on the following:
   - If PFA previously ran on this system and the same 10 jobs that were previously tracked are active, PFA tracks the same jobs.
   - If PFA never ran on the system or the same jobs previously tracked are not all active, PFA collects data for a period of time to use in determining the jobs with the highest arrival rates. After this time, the jobs with the highest arrival rates for that period are individually tracked.

   PFA performs this type of comparison to determine if the SMF arrival rate is higher than expected or lower than expected.
2. **Other persistent jobs:** The persistent jobs that are not individually track are considered "other persistent jobs". The SMF arrival rate check models predictions using the totals for this group. When PFA determines the SMF arrival rate is higher than expected, the comparisons are performed individually using a mathematical formula. When PFA determines the SMF rate is lower than expected, the comparisons are performed using the totals for the group.
3. **Non-persistent jobs:** The jobs that start over an hour after IPL are the non-persistent jobs. The predictions and comparisons are done using the totals for this group. PFA only performs this type of comparison to determine if the SMF arrival rate is higher than expected.
4. **Total system:** The predictions and the comparisons are done using the totals for the entire system. PFA performs this type of comparison to determine if the SMF arrival rate is higher than expected or lower than expected.

Reason for check:

| The objective of this check is to determine if there is potential of damage to an
| LPAR by checking the arrival rate of SMF records per number of CPU
| milliseconds on a system.

Best practice:

If an unexpectedly high number of SMF records was detected, the best practice is to review the SMF records being sent by the address spaces identified on the report and examine the system log to determine what is causing the increase in SMF activity.

If an unexpectedly low number of SMF records was detected, the best practice is to examine the report in SDSF for details about why the exception was issued. Use the Runtime Diagnostics output in the report to assist you in diagnosing and fixing the problem. For more information about Runtime Diagnostics, see Chapter 4, "Runtime Diagnostics," on page 35.

z/OS releases the check applies to:
z/OS V1R12 and later.

Type of check:
Remote

Parameters accepted:
Yes, as follows:

Table 16. PFA_SMF_ARRIVAL_RATE check parameters

Parameter name	Default value	Minimum Value	Maximum Value	Description
collectint	15 Minutes	15	360	This parameter determines how often (in minutes) to run the data collector that retrieves the current SMF arrival rate.
modelint	720 Minutes	60	1440	This parameter determines how often (in minutes) you want the system to analyze the data and construct a new SMF arrival rate model or prediction. By default, PFA analyzes the data and constructs a new model every “default value” minutes. The model interval must be at least four times larger than the collection interval. Note that, even when you set a value larger than 360, PFA performs the first model at 360 minutes (6 hours). By default, PFA analyzes the data and constructs a new model every 720 minutes (12 hours).
stddev	3	2	100	This parameter is used to specify how much variance is allowed between the actual SMF arrival rate per amount of CPU and the expected SMF arrival rate. It is used when determining if the actual SMF arrival rate has increased beyond the allowable upper limit. It also determines how much variance is allowed across the time range predictions. If you set the STDDEV parameter to a small value, an exception will be issued if the actual SMF arrival rate is closer to the expected SMF arrival rate and the predictions across the time ranges are consistent. If you set the STDDEV parameter to a larger value, an exception will be issued if the actual SMF arrival rate is significantly greater than the expected SMF arrival rate even if the predictions across the different time ranges are inconsistent.
collectinactive	1 (on)	0 (off)	1 (on)	Defines whether data will be collected and modeled even if the check is not eligible to run, not ACTIVE(ENABLED), in IBM Health Checker for z/OS.
trackedmin	3	0	1000	This parameter defines the minimum SMF arrival rate required for a persistent job in order for it to be considered a top persistent job that should be tracked individually.
exceptionmin	1	0	1000	This parameter is used when determining if an exception should be issued for an unexpectedly high SMF arrival rate. For tracked jobs and other persistent jobs, this parameter defines the minimum SMF arrival rate and the minimum predicted SMF arrival rate required to cause a too high exception. For non-persistent jobs and the total system comparisons, this parameter defines the minimum SMF arrival rate required to cause a too high exception.
checklow	1 (on)	0 (off)	1	Defines whether Runtime Diagnostics is run to validate that the absence of SMF records is caused by a problem. If this value is off then exceptions will not be issued for conditions in which the SMF arrival rate is unexpectedly low.

Table 16. PFA_SMF_ARRIVAL_RATE check parameters (continued)

Parameter name	Default value	Minimum Value	Maximum Value	Description
stddevlow	4	2	100	This parameter is used to specify how much variance is allowed between the actual SMF arrival rate per amount of CPU and the expected SMF arrival rate when determining if the actual rate is unexpectedly low. • If you set the STDDEVLOW parameter to a smaller value, an exception is issued when the actual SMF arrival rate is closer to the expected SMF arrival rate. • If you set the STDDEVLOW parameter to a larger value, an exception is issued when the actual SMF arrival rate is significantly lower than the expected SMF arrival rate.
limitlow	3	1	100	This parameter defines the maximum SMF arrival rate allowed when issuing an exception for an unexpectedly low number of SMF records.
debug	0 (off)	0 (off)	1 (on)	This parameter (an integer of 0 or 1) is used at the direction of IBM service to generate additional diagnostic information for the IBM Support Center. This debug parameter is used in place of the IBM Health Checker for z/OS policy. The default is off (0).

To determine the status of the SMF arrival rate check, issue `f pfa,display,check(pfa_SMF_arrival_rate),detail`. For the command example and more details, see . The following example shows the output written to message AIR018I in SDSF:

```
AIR018I 02:22:54 PFA CHECK DETAIL

CHECK NAME:  PFA_SMF_ARRIVAL_RATE
    ACTIVE                         : YES
    TOTAL COLLECTION COUNT         : 5
    SUCCESSFUL COLLECTION COUNT    : 5
    LAST COLLECTION TIME           : 02/05/2009 10:18:22
    LAST SUCCESSFUL COLLECTION TIME : 02/05/2009 10:18:22
    NEXT COLLECTION TIME           : 02/05/2009 10:33:22
    TOTAL MODEL COUNT              : 1
    SUCCESSFUL MODEL COUNT         : 1
    LAST MODEL TIME                : 02/05/2009 10:18:24
    LAST SUCCESSFUL MODEL TIME     : 02/05/2009 10:18:24
    NEXT MODEL TIME                : 02/05/2009 22:18:24
    CHECK SPECIFIC PARAMETERS:
       COLLECTINT                  : 15
       MODELINT                    : 720
       COLLECTINACTIVE             : 1=YES
       DEBUG                       : 0=NO
       STDDEV                      : 3
       TRACKEDMIN                  : 3
       EXCEPTIONMIN                : 1
       CHECKLOW                    : 1=YES
       STDDEVLOW                   : 4
       LIMITLOW                    : 3
```

User override of IBM values:

The following shows keywords you can use to override check values on either a POLICY statement in the HZSPRMxx parmlib member or on a MODIFY command. This statement can be copied and modified to override the check defaults:

```
UPDATE CHECK(IBMPFA,PFA_SMF_ARRIVAL_RATE)
            ACTIVE
            SEVERITY(MEDIUM)
```

```
      INTERVAL(ONETIME)
PARMS=('COLLECTINT(15)','MODELINT(720)','STDDEV(3)','DEBUG(0)',
      'COLLECTINACTIVE(1)','EXCEPTIONMIN(1)','TRACKEDMIN(3)'
      'CHECKLOW(1)','STDDEVLOW(4)','LIMITLOW(3)')
      DATE(20080330)
REASON('The SMF arrival rate is abnormal which can indicate a
       system that is damaged.')
```

The SMF arrival rate check is designed to run automatically after every data collection. Do not change the INTERVAL parameter.

Verbose support:
The check provides additional detail in verbose mode. You can put a check into verbose mode using the UPDATE,filters,VERBOSE=ON parameters on either the MODIFY command or in a POLICY statement in an HZSPRMxx parmlib member.

Debug support:
The DEBUG parameter in IBM Health Checker for z/OS is ignored by this check. Rather, the debug parameter is a PFA check specific parameter. For details, see “Understanding how to modify PFA checks” on page 79.

Reference:
For more information about PFA, see the topic on “Overview of Predictive Failure Analysis” on page 65.

Messages:
The output is a SMF arrival rate prediction report that corresponds to the message issued. PFA generates one of the following reports:

- AIRH187E and AIRH208E - tracked jobs exception report
- AIRH188E and AIRH209E - other persistent jobs exception report
- AIRH191E - other non-persistent jobs exception report
- AIRH174E and AIRH175E - total system exception report

For additional message information, see the topics on:

- AIRH messages in *z/OS MVS System Messages, Vol 1 (ABA-AOM)*.
- AIR messages*z/OS MVS System Messages, Vol 1 (ABA-AOM)*.

SECLABEL recommended for MLS users:
SYSLOW

Output:

The output is a variation of the SMF arrival rate prediction report. The values found in the SMF arrival prediction file are as follows:
Tracked jobs exception report: PFA issues the SMF arrival rate exception report for tracked jobs when any one or more tracked, persistent jobs cause an exception. These exceptions can be caused by a higher than expected SMF arrival rate or a lower than expected SMF arrival rate. Only the tracked jobs that caused the exception are included in the list of jobs on the report. If the report was generated due to a lower than expected SMF arrival rate, the report includes Runtime Diagnostics output to can help you diagnose the behavior. The following example is the SMF arrival rate tracked jobs exception report for jobs that had a higher than expected SMF arrival rate (AIRH187E):

```
 SMF Arrival Rate Prediction Report

Last successful model time      :  01/27/2009 11:08:01
Next model time                 :  01/27/2009 23:08:01
Model interval                  :  720
Last successful collection time :  01/27/2009 17:41:38
Next collection time            :  01/27/2009 17:56:38
Collection interval             :  15

Persistent address spaces with high rates:

                                               Predicted SMF
                            SMF                 Arrival Rate
  Job                   Arrival
  Name     ASID            Rate        1 Hour       24 Hour        7 Day
______________________________________________________________________________
  TRACKED1 001D           75.63         23.88         22.82        15.82
  TRACKED2 0028           43.52          0.34         11.11        12.11
  TRACKED3 0029           53.25         12.43          2.36         8.36
```

Figure 31. SMF arrival rate: tracked jobs higher than expected

The following example is the SMF arrival rate tracked jobs exception report for jobs that had a lower than expected SMF arrival rate (for AIH208E):

```
SMF Arrival Rate Prediction Report

Last successful model time      : 01/27/2009 11:08:01
Next model time                 : 01/27/2009 23:08:01
Model interval                  : 360
Last successful collection time : 01/27/2009 17:41:38
Next collection time            : 01/27/2009 17:56:38
Collection interval             : 15

Persistent address spaces with low rates:
                                       Predicted SMF
                        SMF              Arrival Rate
Job                     Arrival
Name       ASID         Rate         1 Hour       24 Hour       7 Day
TRACKED4   005D         0.20         23.88        22.82         15.82
TRACKED5   0034         0.01         12.43        11.11          8.36

Runtime Diagnostics Output:

Runtime Diagnostics detected a problem in job: TRACKED4
  EVENT 06: HIGH - HIGHCPU - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID CPU RATE: 96% ASID: 0027 JOBNAME: TRACKED4
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MAY BE LOOPING.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
  EVENT 07: HIGH - LOOP - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID: 0027 JOBNAME: TRACKED4 TCB: 004E6850
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE APPEARS TO BE IN A LOOP.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
Runtime Diagnostics detected a problem in job: TRACKED5
  EVENT 08: HIGH - HIGHCPU - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID CPU RATE: 96% ASID: 0027 JOBNAME: TRACKED5
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MAY BE LOOPING.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
  EVENT 09: HIGH - LOOP - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID: 0027 JOBNAME: TRACKED5 TCB: 004E6850
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE APPEARS TO BE IN A LOOP.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
```

Figure 32. SMF arrival rate: tracked jobs lower than expected

Other persistent jobs exception report: PFA issues the SMF arrival rate other persistent jobs exception report when a comparison of a persistent job (that is not being individually tracked) causes an exception when compared to the totals of the other persistent jobs. The exception can be caused by a higher than expected SMF arrival rate or a lower than expected SMF arrival rate. The predictions listed on this report are the predicted rates for the total other persistent jobs group. The list of jobs is only those persistent jobs (not tracked individually) that have a problem and is only generated for a higher than expected SMF arrival rate. No predictions are given for these jobs because PFA does not model individual predictions for jobs that are not tracked individually. If there is more than one job with the same name, four asterisks **** are printed for the ASID in the report. If the report was generated due to a lower than expected SMF arrival rate, the report will include Runtime Diagnostics output which can help diagnose the behavior. The following example is the SMF arrival rate other persistent jobs exception report for jobs that had an unexpectedly high SMF arrival rate (AIRH188E):

```
SMF Arrival Rate Prediction Report

Last successful model time      :  01/27/2009 11:08:01
Next model time                 :  01/27/2009 23:08:01
Model interval                  :  360
Last successful collection time :  01/27/2009 17:41:38
Next collection time            :  01/27/2009 17:56:38
Collection interval             :  15

Other persistent jobs group:
Prediction based on 1 hour of data   : 20.27
Prediction based on 24 hours of data : 27.98
Prediction based on 7 days of data   : 31.22

Persistent address spaces with high rates:

                            SMF
  Job                   Arrival
  Name     ASID            Rate
  ________ ____   ____________
  PERS1    001E           83.22
  PERS2    0038           75.52
  PERS3    0039           47.47
```

Figure 33. SMF arrival rate: other persistent jobs higher than expected

This example is the SMF arrival rate other persistent jobs exception report for jobs that had a lower than expected SMF arrival rate (for AIRH209E):

```
Last successful model time      : 01/27/2009 11:08:01
Next model time                 : 01/27/2009 23:08:01
Model interval                  : 360
Last successful collection time : 01/27/2009 17:41:38
Next collection time            : 01/27/2009 17:56:38
Collection interval             : 15

Other persistent jobs group:
 Prediction based on 1 hour of data   : 20.27
 Prediction based on 24 hours of data : 27.98
 Prediction based on 7 days of data   : 31.22

Runtime Diagnostics Output:

  EVENT 01: HIGH - HIGHCPU - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID CPU RATE: 96% ASID: 0027 JOBNAME: PERS4
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MAY BE LOOPING.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
  EVENT 02: HIGH - LOOP - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID: 0027 JOBNAME: PERS4 TCB: 004E6850
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE APPEARS TO BE IN A LOOP.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
  EVENT 03: HIGH - HIGHCPU - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID CPU RATE: 96% ASID: 0027 JOBNAME: PERS5
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MAY BE LOOPING.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
  EVENT 04: HIGH - LOOP - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID: 0027 JOBNAME: PERS5 TCB: 004E6850
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE APPEARS TO BE IN A LOOP.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
```

Figure 34. SMF arrival rate: other persistent jobs lower than expected

Non-persistent jobs exception report: PFA issues the SMF arrival rate non-persistent jobs exception report when the non-persistent jobs (as a group) cause an exception. This exception is only issued for a higher than expected SMF arrival rate. The SMF arrival rate and predictions listed on this report are calculated for the total non-persistent jobs group. The list of jobs contains only three non-persistent jobs that have high arrival counts. No predictions are given for these jobs because PFA does not model individual predictions for jobs that are not tracked individually. The following example is the SMF arrival rate non-persistent jobs exception report (AIRH191E):

```
 SMF Arrival Rate Prediction Report

Last successful model time      :  01/27/2009 11:08:01
Next model time                 :  01/27/2009 23:08:01
Model interval                  :  360
Last successful collection time :  01/27/2009 17:41:38
Next collection time            :  01/27/2009 17:56:38
Collection interval             :  15

Non-persistent jobs group:
 SMF arrival rate
   in last collection interval        : 65.49
 Prediction based on 1 hour of data   : 20.27
 Prediction based on 24 hours of data : 27.98
 Prediction based on 7 days of data   : 31.22
Address spaces with high arrivals:

                          SMF
  Job                 Arrival
  Name     ASID        Counts

  ________ ____   ___________
  NONPERS1 001F            83
  NONPERS2 0048            52
  NONPERS3 0049            47
```

Figure 35. SMF arrival rate: non-persistent jobs with high counts

No problem and total system exception report: When no exception is issued or when a total SMF arrival rate exception is issued, the following report is generated. When there is no problem or when an exception occurs because of a higher than expected SMF arrival rate, the list of jobs contains the jobs being tracked individually and the list of jobs can vary from one to ten. The Runtime Diagnostics section is written in the report when the exception is issued because there is a lower than expected SMF arrival rate. The following example is the SMF arrival rate no problem report (AIRH176I) and total system exception report issued due to a high than expected SMF arrival rate (AIRH174E) showing three jobs.

```
 SMF Arrival Rate Prediction Report

Last successful model time      :  01/27/2009 11:08:01
Next model time                 :  01/27/2009 23:08:01
Model interval                  :  360
Last successful collection time :  01/27/2009 17:41:38
Next collection time            :  01/27/2009 17:56:38
Collection interval             :  15

SMF arrival rate
   at last collection interval       :       83.52
Prediction based on 1 hour of data   :       98.27
Prediction based on 24 hours of data :       85.98
Prediction based on 7 days of data   :      100.22

Top persistent users:

                                            Predicted SMF
                        SMF                  Arrival Rate
  Job                Arrival
  Name     ASID         Rate         1 Hour        24 Hour         7 Day

  ________ ____  ____________  ____________  ____________  ____________
  TRACKED1 001D         58.00         23.88         22.82         15.82
  TRACKED2 0028         11.00          0.34         11.11         12.11
  TRACKED3 0029         11.00         12.43          2.36          8.36
.
.
.
```

Figure 36. SMF arrival rate: no problem and total system higher than expected

The following example is the SMF arrival rate total system exception report issued because of a lower than expected SMF arrival rate (AIRH175E):

```
 SMF Arrival Rate Prediction Report

Last successful model time      : 01/27/2009 11:08:01
Next model time                 : 01/27/2009 23:08:01
Model interval                  : 360
Last successful collection time : 01/27/2009 17:41:38
Next collection time            : 01/27/2009 17:56:38
Collection interval             : 15

SMF arrival rate
   in last collection interval       : 2.05
Prediction based on 1 hour of data   : 98.27
Prediction based on 24 hours of data : 85.98
Prediction based on 7 days of data   : 100.22

Runtime Diagnostics Output:

  EVENT 01: HIGH - CF - SYSTEM: SY1 2009/06/12 - 13:23:25
  IXL013I IXLCONN REQUEST FOR STRUCTURE SYSZWLM_WORKUNIT FAILED.
  JOBNAME: WLM ASID: 000A CONNECTOR NAME: #SY1
  IXLCONN RETURN CODE: 0000000C, REASON CODE: 02010C05
  STRUCTURE NOT DEFINED IN THE CFRM ACTIVE POLICY
  CONADIAG0: 00000002
----------------------------------------------------------------------
  EVENT 02: HIGH - XCF - SYSTEM: SY1 2009/06/12 - 13:23:55
  IXC467I STOPPING PATHIN STRUCTURE IXCT_SIGNAL
  RSN: START REQUEST FAILED
----------------------------------------------------------------------
  EVENT 03: HIGH - XCF - SYSTEM: SY1 2009/06/12 - 13:23:58
  IXC467I STOPPING PATHIN STRUCTURE IXCT_SIGNAL
  RSN: START REQUEST FAILED
----------------------------------------------------------------------
  EVENT 04: HIGH - CF - SYSTEM: SY1 2009/06/12 - 13:24:02
  IXL013I IXLCONN REQUEST FOR STRUCTURE IXCT_SIGNAL FAILED.
  JOBNAME: XCFAS ASID: 0006 CONNECTOR NAME: SIGPATH_01000008
  IXLCONN RETURN CODE: 0000000C, REASON CODE: 02010C05
  STRUCTURE NOT DEFINED IN THE CFRM ACTIVE POLICY
  CONADIAG0: 00000002
----------------------------------------------------------------------
  EVENT 05: HIGH - XCF - SYSTEM: SY1 2009/06/12 - 13:24:02
  IXC467I STOPPING PATHIN STRUCTURE IXCT_SIGNAL
  RSN: START REQUEST FAILED
----------------------------------------------------------------------
  EVENT 06: HIGH - HIGHCPU - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID CPU RATE: 96% ASID: 0027 JOBNAME: DAVIDZ
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MAY BE LOOPING.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
  EVENT 07: HIGH - LOOP - SYSTEM: SY1 2009/06/12 - 13:28:46
  ASID: 0027 JOBNAME: DAVIDZ TCB: 004E6850
  STEPNAME: DAVIDZ PROCSTEP: DAVIDZ JOBID: STC00042 USERID: ++++++++
  JOBSTART: 2009/06/12 - 13:28:35
Error:
  ADDRESS SPACE APPEARS TO BE IN A LOOP.
Action:
  USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
----------------------------------------------------------------------
```

Figure 37. SMF arrival rate: no problem and total system lower than expected

Note: In accordance with the IBM Health Checker for z/OS messaging guidelines, the largest generated output length for decimal variable values up to 2147483647 (X'7FFFFFF') is 10 bytes. When any PFA report value is greater

than 2147483647, it displays using multiplier notation with a maximum of six characters. For example, if the report value is 2222233333444445555, PFA displays it as 1973P (2222233333444445555 ÷ 1125899906842) using the following multiplier notation:

Table 17. Multiplier notation used in values for PFA reports

Name	Sym	Size
Kilo	K	1,024
Mega	M	1,048,576
Giga	G	1,073,741,824
Tera	T	1,099,511,627,776
Peta	P	1,125,899,906,842

The following fields apply to all four reports:

- **Last successful model time:** The date and time of the last successful model for this check. The predictions on this report were generated at that time.
- **Next model time:** The date and time of the next model. The next model will recalculate the predictions.
- **Model interval:** The value in the configured MODELINT parameter for this check. If PFA determines new prediction calculations are necessary, modeling can occur earlier.
- **Last successful collection time:** The date and time of the last successful data collection for this check.
- **Next collection time:** The date and time of the next collection.
- **Collection interval:** The value in the configured COLLECTINT parameter for this check.
- **SMF arrival rate in last collection interval:** The actual SMF arrival rate in the last collection interval where the rate is defined to be the number of messages divided by the CPU milliseconds.
- **Predicted rates based on...:** The SMF arrival rates based on one hour, 24 hours, and seven days. If no prediction is available for a given time range, the line is not printed. For example, if the check has been running for 2 days, 7 days of data is not available therefore PFA does not print the “Prediction based on 7 days of data” line.
- **Runtime Diagnostics Output:** Runtime Diagnostics event records to assist you in diagnosing and fixing the problem. See the topic on “Runtime Diagnostics symptoms” on page 39 in Chapter 4, “Runtime Diagnostics,” on page 35.
- **Job Name:** The name of the job that has SMF arrivals in the last collection interval.
- **ASID:** The ASID for the job that has SMF arrivals in the last collection interval.
- **SMF Arrival Rate:** The current SMF arrival rate for the job.
- **SMF Arrival Count:** The SMF arrival rate, from the last interval report, for the non-persistent job.

  Note: The “SMF Arrival Count” field is unique to the non-persistent jobs exception report.
- **Predicted SMF Arrival Rate:** The predicted SMF arrival rate based on one hour, 24 hours, and seven days of data. If PFA did not previously run on this system or the same jobs previously tracked are not all active, there will

not be enough data for a time range until that amount of time has passed. Also, gaps in the data caused by stopping PFA or by an IPL might cause the time range to not have enough data available. After the check collects enough data for any time range, predictions are made again for that time range. If there is not enough data for a time range, INELIGIBLE is printed and comparisons are not made for that time range.

Directories

Note: The content and names for these files and directories are subject to change and cannot be used as programming interfaces; these files are documented only to provide help in diagnosing problems with PFA.

pfa_directory

This directory contains all the PFA checks and is pointed to by the home directory of the started task. The following files only contain data if messages are generated by the JVM:

- java.stderr (generated by JVM)
- java.stdout (generated by JVM)

pfa_directory/PFA_SMF_ARRIVAL_RATE/data

The directory for SMF arrival rate that holds data and modeling results. PFA automatically deletes the contents of the `PFA_SMF_ARRIVAL_RATE/data` directory that could lead to skewed predictions in the future.

Guideline: If the use of the z/OS image is radically different after an IPL (for instance, the change from a test system to a production system) of if you modify the SMF record types in SMFPRMxx, delete the files in the *PFA_SMF_ARRIVAL_RATE/data* directory to ensure the check can collect the most accurate modeling information.

Results files

- systemName.1hr.prediction - This file is generated by the modeling code for the predictions made for one hour of historical data. It contains predictions for each of the tracked address spaces, the other persistent category, the non-persistent category, and the total system category. It also contains additional information required for PFA processing.
- systemName.24hr.prediction - This file is generated by the modeling code for the predictions made for 24 hours of historical data. It contains predictions for each of the tracked address spaces, the other persistent category, the non-persistent category, and the total system category. It also contains additional information required for PFA processing.
- systemName.7day.prediction - This file is generated by the modeling code for the predictions made for seven days of historical data. It contains predictions for each of the tracked address spaces, the other persistent category, the non-persistent category, and the total system category. It also contains additional information required for PFA processing.
- systemName.1hr.prediction.html - This file contains an .html report version of the data found in the systemName.1hr.prediction file.
- systemName.24hr.prediction.html - This file contains an .html report version of the data found in the systemName.24hr.prediction file.

- systemName.7day.prediction.html - This file contains an .html report version of the data found in the systemName.7day.prediction file.
- systemName.prediction.stddev - The file generated by the modeling code to list the standard deviation of the predictions across the time ranges for each job.

Data store files:

- systemNameSAR.OUT - The data collection file.

Intermediate files:

- systemName.sardata - The file is used as input to the modeling to track if enough data is available to model.
- systemName.1hr.sardata - The file used as input to modeling code. It contains one hour of historical data.
- systemName.24hr.sardata - The file used as input to modeling code. It contains 24 hours of historical data.
- systemName.7day.sardata - The file used as input to modeling code. It contains seven days of historical data.
- systemName.1hr.holes - The file is used to track gaps in data, caused by stopping PFA or by an IPL, for a one hour period.
- systemName.24hr.holes - The file is used to track gaps in the data, caused by stopping PFA or by an IPL, for a 24 hour time period.
- systemName.7day.holes - The file is used to track gaps in the data, caused by stopping PFA or by an IPL, for the seven day time period.

This directory holds the following log files. Additional information is written to these log files when DEBUG(1).

- systemName.1hr.cart.log - The log file generated by modeling code with details about code execution while one hour of historical data was being modeled.
- systemName.24hr.cart.log - The log file generated by modeling code with details about code execution while 24 hours of historical data was being modeled.
- systemName.7day.cart.log - The log file generated by modeling code with details about code execution while seven days of historical data was being modeled.
- systemName.buildSar.log - The log file generated by intermediate code that builds the files that are input to modeling with details about code execution.
- systemName.launcher.log - The log file generated by launcher code.
- systemName.1hr.tree - This file is generated by the modeling code. It contains information about the model tree which was built based on the last one hour of collected data.
- systemName.24hr.tree - This file is generated by the modeling code. It contains information about the model tree which was built based on the last 24 hours of collected data.
- systemName.7day.tree - This file is generated by the modeling code. It contains information about the model tree which was built based on the last seven days of collected data.
- systemNameCONFIG.LOG - The log file containing the configuration history for the last 30 days for this check.
- systemNameCOLLECT.LOG - The log file used during data collection.

- systemNameMODEL.LOG - The log file used during portions of the modeling phase.
- systemNameRUN.LOG - The log file used when the check runs.

pfa_directory/PFA_SMF_ARRIVAL_RATE/EXC_timestamp
This directory contains all the relevant data for investigating exceptions issued by this check at the timestamp provided in the directory name. PFA keeps directories only for the last 30 exceptions. Therefore at each exception, if more than 30 exception directories exist, the oldest directory is deleted so that only 30 exceptions remain after the latest exception is added.

- systemNameREPORT.LOG - The log file containing the same contents as the IBM Health Checker for z/OS report for this exception as well as other diagnostic information issued during report generation (such as Runtime Diagnostic event records).

pfa_directory/PFA_SMF_ARRIVAL_RATE/config
This directory contains the configuration files for the check.

- EXCLUDED_JOBS - The file containing the list of excluded jobs for this check.

Note: When using Runtime Diagnostics, it is possible to see data for jobs previously defined to the excluded jobs list in the “other persistent jobs” and “total system” categories because PFA must return any potential problem activity on the system identified by Runtime Diagnostics.

Part 4. Diagnosing by problem type

After you identify the problem type, use the following diagnosis procedures to identify the source and extract symptoms to build a search argument.

Chapter 10. Diagnosing an abend

- “Overview of an abend” includes abend symptoms and examples
- “Steps for diagnosing an abend” on page 158 contains a flowchart and these steps to guide your diagnosis of an abend:
  1. “Obtaining the abend and reason code” on page 159
  2. “Identifying the module and component” on page 163
  3. “Searching the problem reporting databases” on page 166
  4. “Gathering additional problem data for abends” on page 167
     – “Steps for gathering trace data for abends” on page 168
     – “Steps for collecting additional messages and logrec for abends” on page 169
     – “Steps for obtaining a dump for the error” on page 171

Overview of an abend

The purpose of this chapter is to guide the diagnosis of an abnormal end (abend). Abends have an associated system completion code to describe the error and most have a reason code to further explain the error. These codes can be found by searching:

- *z/OS MVS System Codes*
- The documentation for the particular application that failed. For example:
  – For Language Environment completion codes, see *z/OS Language Environment Runtime Messages*.
  – For RMF completion codes, see *z/OS RMF Messages and Codes*.

An abend is classified as follows:

- **Software-detected:**
  – A system code in the form of three hexadecimal digits, possibly with a four byte reason code. For example, ABEND075. A system abend code is issued with the ABEND or CALLRTM macros used to terminate a task or address space when a system service or function detects an error.
  – A user code in the form of a four decimal digits, possibly with a four byte reason code. For example, ABENDU4094. A user code is issued using the ABEND macro to terminate a task or the entire job step. When the highest-level task in a job step ends abnormally, all related tasks or subtasks also terminate. When a subtask terminates, only work running on behalf of the subtask is affected, unless STEP=YES is specified.
- **Hardware-detected:**

  Hardware might present a program interrupt or machine check on the execution of an instruction. The operating system detects these hardware problems and presents them as an abend.

  Example: An instruction in an application running in storage key 7 references storage assigned to key 0. The difference in storage key causes a protection exception. This exception results in hardware presenting a program interruption code of 0004 to the operating system, which is externalized as ABEND0C4.

Related information:

- z/Architecture® Principles of Operation, SA22-7832

Symptoms of an abend: You can identify an abend by one or more of the following indicators:

- A symptom dump message on the console, in the system log, or job log can indicate a system or user abend.

  For example, message IEA995I is issued to the console:

  System message indicating an abend

  Notice the indication of a system completion code.

```
IEA995I SYMPTOM DUMP OUTPUT 731
SYSTEM COMPLETION CODE=EC6  REASON CODE=0000FD18
 TIME=13.58.26  SEQ=00724  CPU=0000  ASID=0147
 PSW AT TIME OF ERROR  070C4400   A90B111A  ILC 2  INTC 78
   NO ACTIVE MODULE FOUND - PRIMARY NOT EQUAL TO HOME
   NAME=UNKNOWN
   DATA AT PSW  290B1114 - 1F001F11  05EFEBEC  D2640096
   AR/GR 0: 00000000/00000000   1: 00000000/00000000
         2: 00000000/2ED9E4D0   3: 00000007/820A33B8
         4: 00000000/294F31D8   5: 00000000/2ED9E200
         6: 00000000/28AADD08   7: 00000002/00000100
         8: 00000007/7F29ECF8   9: 00000002/7F29EC18
         A: 00000000/00FD8B28   B: 00000000/000000E3
         C: 00000000/006FF390   D: 00000000/00F9C5B8
         E: 00000000/A90B111A   F: 00000000/00FFFB4C
```

- A system message indicating an SVC dump was requested for an error:

  For example, here are some messages that might be issued when a SVC dump is taken for an error:

```
IEA794I SVC DUMP HAS CAPTURED: 357
DUMPID=002 REQUESTED BY JOB (OMVS)
DUMP TITLE=COMPON=BPX,COMPID=SCPX1,ISSUER=BPXMIPCE,MODULE=BPXLK
           LCP+1DC2,ABEND=S0422,REASON=083A01A5

IEA611I COMPLETE DUMP ON DUMP.MVS06.D060320.T162245.S00034
DUMPID=034 REQUESTED BY JOB (ZFS )
FOR ASID (1001)
INCIDENT TOKEN: ORACLE MVS06 03/21/2006 00:22:45
ID = IOEDFS

IEA911E COMPLETE DUMP ON SYS1.DUMP08
DUMPID=001 REQUESTED BY JOB (RESOLVER)
FOR ASIDS(003B,0001)
INCIDENT TOKEN: CWYPLEX1 CPUX 04/21/2006 14:38:56
ERROR ID = SEQ00046 CPU00 ASID003B TIME09.38.56.1
```

- An application detects an error. One example is the following ISMF panel:

```
ISMF ABEND PANEL
COMMAND ===> _
******************************************************************************
******************************************************************************
**                                                                          **
**                                                                          **
**                 AN ABEND OCCURRED WHILE EXECUTING ISMF                   **
**                                                                          **
**                        SYSTEM ABEND CODE:  0C4                           **
**                                                                          **
**                          ISMF CANNOT CONTINUE                            **
**                                                                          **
**           PRESS THE ENTER KEY OR USE END TO TERMINATE ISMF               **
**           USE HELP TO DISPLAY A LIST OF COMMON ABEND CODES               **
**                                                                          **
******************************************************************************
******************************************************************************
```

  Another example is the ISPF panel:

```
--------------------ERROR RECOVERY-------------------------------
COMMAND ===>
    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
    * *                                                         * *
* *      ISPF PROCESSOR ENDED ABNORMALLY                 * *
* *                                                      * *
* *                                                      * *
* *                                                      * *
* *                                                      * *
* *           Task ABEND code 0C1                        * *
* *                                                      * *
* *                                                      * *
* *                                                      * *
* *                                                      * *
* *   Press ENTER to display primary option menu.        * *
* *   Enter HELP command for list of common ABEND CODES.  * *
* *                                                      * *
* *                                                      * *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

- A component, function, subsystem or application message indicating an abend occurred through a message. For example, TSO/E message INMR030I that identifies an abend condition:

  TSO/E message
  ```
  INMR030I RECEIVE command terminated.  ABEND abend_code.
  ```
- An error is recorded in SYS1.LOGREC record.

  For example:

```
ERRORID: SEQ=11696 CPU=0040 ASID=00A1 TIME=12:48:20.3

SEARCH ARGUMENT ABSTRACT

PIDS/5752SCXMS RIDS/IEANUC01#L RIDS/IEAVXALA AB/S013E REGS/0D000 REGS/C009C
RIDS/IEAVXALR#R

SYMPTOM DESCRIPTION
------- -----------
PIDS/5752SCXMS PROGRAM ID: 5752SCXMS
RIDS/IEANUC01#L LOAD MODULE NAME: IEANUC01
RIDS/IEAVXALA CSECT NAME: IEAVXALA
AB/S013E SYSTEM ABEND CODE: 013E
REGS/0D000 REGISTER/PSW DIFFERENCE FOR R0D: 000
REGS/C009C REGISTER/PSW DIFFERENCE FOR R0C:-009C
RIDS/IEAVXALR#R RECOVERY ROUTINE CSECT NAME: IEAVXALR

OTHER SERVICEABILITY INFORMATION

RECOVERY ROUTINE LABEL: IEAVXALR
DATE ASSEMBLED: 96270
MODULE LEVEL: HBB6603
SUBFUNCTION: ACCESS LIST ADD

TIME OF ERROR INFORMATION

PSW: 070C0000 80FF5D00 INSTRUCTION LENGTH: 02 INTERRUPT CODE: 0078
FAILING INSTRUCTION TEXT: 5DB88140 174458C0 022856C0

REGISTERS 0-7
GR: 01381495 00000000 00000002 7FFFBF00 00000000 00000000 7FFFBF10 00000000
AR: 00000000 00000000 00000000 00000002 00000000 00000000 00000002 00000000
REGISTERS 8-15
GR: 00FF3D88 01382494 81380496 00F9B700 80FF5D9C 80FF5D00 81380D58 01BFD620
AR: 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

HOME ASID: 00A1 PRIMARY ASID: 0002 SECONDARY ASID: 00A1
PKM: 80C0 AX: 0001 EAX: FFFF
```

Often, a system completion code or a wait state code indicate an abend. However, there are some exceptions, so use the following table to help guide your diagnosis of an abend:

If you receive wait state code	Code represents	Diagnose with	Notes®
X'071'	System failure or the operator initiated a restart.	Chapter 13, "Diagnosing a loop," on page 203	Find complete explanations of wait state codes in *z/OS MVS System Codes*.
Abend X'122'	Operator canceled the job, requesting a dump.	Chapter 12, "Diagnosing a job or subsystem hang," on page 193	This abend might also indicate a loop, see Chapter 13, "Diagnosing a loop," on page 203. Find complete explanations of wait state codes in *z/OS MVS System Codes*.
Abend X'222'	Operator canceled the job, without requesting a dump.	Chapter 12, "Diagnosing a job or subsystem hang," on page 193	This abend might also indicate a loop, see Chapter 13, "Diagnosing a loop," on page 203. Find complete explanations of wait state codes in *z/OS MVS System Codes*.
X'322'	Job exceeded the time limit specified by the TIME option.	Chapter 13, "Diagnosing a loop," on page 203	Find complete explanations of wait state codes in *z/OS MVS System Codes*.
All others		"Steps for diagnosing an abend"	To find the abend code and reason code, see "Obtaining the abend and reason code" on page 159.

Steps for diagnosing an abend

About this task

Before you begin: You need to know how to use Interactive Problem Control System (IPCS) and have access to the following:

- SVC dump, SYSMDUMP, or stand-alone dump
- EREP of software records (TYPE=S) from SYS1.LOGREC or IPCS VERBX LOGDATA report from a dump.
- OPERLOG, SYSLOG, Job log, other message log for the time frame of the error

You should also be able to locate and use the following:

- LookAt for messages and codes (z/OS V1R13 and earlier), see www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/.
- IBM product documentation; see Chapter 22, "Diagnosis information for z/OS base elements and features," on page 285.

Use the following flowchart to guide diagnosis of an abend.

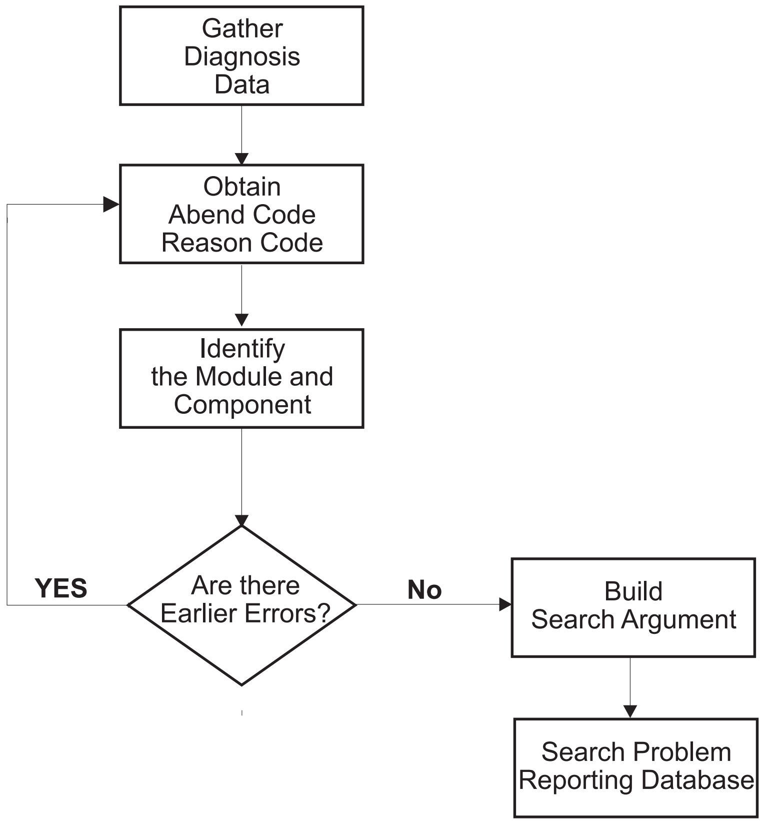

Gather Diagnosis Data
Obtain Abend Code Reason Code
Identify the Module and Component
Are there Earlier Errors?
YES
No
Build Search Argument
Search Problem Reporting Database

Figure 38. Flowchart for abend analysis

Use the following steps to guide diagnosis of an abend:

Procedure

1. "Obtaining the abend and reason code"
2. "Identifying the module and component" on page 163
3. "Searching the problem reporting databases" on page 166
4. "Gathering additional problem data for abends" on page 167
   - "Steps for collecting additional messages and logrec for abends" on page 169
   - "Steps for obtaining a dump for the error" on page 171
   - "Steps for gathering trace data for abends" on page 168

Obtaining the abend and reason code

The abend code indicates the nature of the problem. If you have the abend code, move on to "Identifying the module and component" on page 163.

Steps for obtaining the abend code

Before you begin: You need to know how to access the following:

- SVC dump, SYSMDUMP, and transaction dump
- Software record in SYS1.LOGREC from the time frame of the error.

If an SVC dump was taken for the error as indicated by IEA794I, IEA611I or IEE911E, use the following IPCS commands to format information and extract the abend and reason code (when available).

- IPCS STATUS FAILDATA

  For the abend code, look for AB/S0hhh in the symptoms under the heading SEARCH ARGUMENT ABSTRACT, if present. For the reason code, look for PRCS/hhhhhhhh in the symptoms, or look in the register indicated in the abend code explanation.

  Not every dump has error information associated with it. Some dumps, like those requested through the SLIP or DUMP command will not have STATUS FAILDATA output. There are occasions when STATUS FAILDATA will not provide the information you need. Check SYSTRACE and SUMMARY FORMAT (look for RTM2WA SUMMARY) for the abend. If you cannot find it, the dump was probably not requested for an abend. Check the dump title (L Title) and determine why it was not requested. Also, check SYS1.LOGREC, and the SYSLOG and job log for the time frame of the dump.

  In the following example, fields **AB/S005C** and **PRCS/00000214** indicate what to extract for the abend and reason code. The free-format search argument is: ABEND05C and RSN00000214.

```
* * * DIAGNOSTIC DATA REPORT * * *


SEARCH ARGUMENT ABSTRACT

RIDS/IEFW21SD#L RIDS/#UNKNOWN AB/S005C PRCS/00000214 REGS/0E01E REGS/0C6D8

Symptom        Description
-------        -----------
RIDS/IEFW21SD#L    Load module name: IEFW21SD
RIDS/#UNKNOWN     Csect name: #UNKNOWN
AB/S005C       System abend code: 005C
PRCS/00000214     Abend reason code: 00000214
REGS/0E01E       Register/PSW difference for R0E: 01E
REGS/0C6D8       Register/PSW difference for R0C: 6D8

SERVICEABILITY INFORMATION NOT PROVIDED BY THE RECOVERY ROUTINE

Program id
Recovery routine csect name
Recovery Routine Label
Date Assembled
Module Level
Subfunction

Time of Error Information

PSW: 071C2000 83AA2110 Instruction length: 02 Interrupt code: 000D
Failing instruction text: BFFFB148 0A0D98EC B08807FE

Breaking event address: 00000000_00000000
AR/GR 0-1 00000000/00000001_00000000 00000000/00000000_0405C000
AR/GR 2-3 00000000/00000000_7F10C128 00000000/00000000_00000024
AR/GR 4-5 00000000/00000000_006EA338 00000000/00000000_006A3648
AR/GR 6-7 00000000/00000000_006A2D64 00000000/00000000_7F363028
AR/GR 8-9 00000000/00000000_00000000 00000000/00000000_00000000
```

  In the following **STATUS FAILDATA** output, the symptom string indicates a system abend code of X'0C4' with a reason code of X'00000010'

```
SEARCH ARGUMENT ABSTRACT

   PIDS/5752SCPX1 RIDS/BPXINPVT#L RIDS/BPXVFPCT AB/S00C4 PRCS/00000010
   REGS/C4E10 RIDS/BPXMIPCE#R

   Symptom            Description
   -------            -----------
   PIDS/5752SCPX1      Program id: 5752SCPX1
```

```
    RIDS/BPXINPVT#L     Load module name: BPXINPVT
    RIDS/BPXVFPCT       Csect name: BPXVFPCT
    AB/S00C4            System abend code: 00C4
    PRCS/00000010       Abend reason code: 00000010
    REGS/C4E10          Register/PSW difference for R0C:-4E10
    RIDS/BPXMIPCE#R     Recovery routine csect name: BPXMIPCE

  OTHER SERVICEABILITY INFORMATION

    Recovery Routine Label:  BPXMIPCE
    Date Assembled:          12/19/04
    Module Level:             HBB7720
    Subfunction:             OpenMVS

  Time of Error Information

    PSW: 47043000 80000000 00000000 27EA624C
    Instruction length: 02   Interrupt code: 0010
    Failing instruction text: 58E0D56C 18F10E0E B2190200
    Translation exception address: 00000000_66831000
```

- EREP or **VERBX LOGDATA**

  Use EREP to format software records (TYPE=S) recorded to SYS1.LOGREC for the time frame of the failure or error or format SYS1.LOGREC records from the in storage buffer using the **VERBX LOGDATA** command. For information about EREP, see *EREP Reference*.

  To find software records that might be associated with the failure, follow these suggestions:

  – Look for the general time frame
  – Search on the job name or ASID involved
  – Search for the failing component id.

  For the abend code, look for AB/S0hhh in the symptoms under the heading SEARCH ARGUMENT ABSTRACT, if present.

  For the reason code, look for PRCS/hhhhhhhh in the symptoms, or look in the register indicated in the abend code explanation.

  Here is an example of a software logrec entry formatted by EREP or the IPCS **VERBX LOGDATA** command against an SVC dump. The free-format search argument generated for this entry is: ABEND05C RSN00000214.

```
TYPE: SOFTWARE RECORD REPORT: SOFTWARE EDIT REPORT DAY.YEAR
(SVC 13) REPORT DATE: 115.06
FORMATTED BY: IEAVTFDE HBB7703 ERROR DATE: 096.06
MODEL: 2084 HH:MM:SS.TH
SERIAL: 0D8B9F TIME: 12:14:42.66

JOBNAME: HSMVM0A SYSTEM NAME: VM0A
ERRORID: SEQ=03567 CPU=0000 ASID=003C TIME=12:14:42.6

SEARCH ARGUMENT ABSTRACT

PIDS/5752SC1B4 RIDS/IEFW21SD#L RIDS/IEFDB400 AB/S005C PRCS/00000214 REGS/0E01E
REGS/0C6D8 RIDS/IEFDB402#R

SYMPTOM DESCRIPTION
------- -----------
PIDS/5752SC1B4 PROGRAM ID: 5752SC1B4
RIDS/IEFW21SD#L LOAD MODULE NAME: IEFW21SD
RIDS/IEFDB400 CSECT NAME: IEFDB400
AB/S005C SYSTEM ABEND CODE: 005C
PRCS/00000214 ABEND REASON CODE: 00000214
REGS/0E01E REGISTER/PSW DIFFERENCE FOR R0E: 01E
REGS/0C6D8 REGISTER/PSW DIFFERENCE FOR R0C: 6D8
```

```
RIDS/IEFDB402#R RECOVERY ROUTINE CSECT NAME: IEFDB402

OTHER SERVICEABILITY INFORMATION

RECOVERY ROUTINE LABEL: IEFAB4ED
DATE ASSEMBLED: 05223
MODULE LEVEL: UA20441
SUBFUNCTION: DYNAMIC ALLOCATION

TIME OF ERROR INFORMATION

PSW: 071C2000 83AA2110 INSTRUCTION LENGTH: 02 INTERRUPT CODE: 000D
FAILING INSTRUCTION TEXT: BFFFB148 0A0D98EC B08807FE

REGISTERS 0-7
GR: 00000000 0405C000 7F10C128 00000024 006EA338 006A3648 006A2D64 7F363028
```

- OPERLOG, SYSLOG, or job log or **VERBEXIT MTRACE**

  Look for message IEA995I or other messages with an abend code in the message text. The message might also give a reason code. For example:

  ```
  IEA995I SYMPTOM DUMP OUTPUT 694
  USER COMPLETION CODE=4039 REASON CODE=00000000
  TIME=05.07.41 SEQ=33565 CPU=0000 ASID=0247
  PSW AT TIME OF ERROR 078D1000 99937C66 ILC 2 INTC 0D
  ACTIVE LOAD MODULE ADDRESS=1987A4C0 OFFSET=000BD7A6
  NAME=CEEPLPKA
  DATA AT PSW 19937C60 - 00181610 0A0D58D0 D00498EC
  AR/GR 0: 80C4BB3E/84000000 1: 00000000/84000FC7
  2: 00000000/000A1E08 3: 00000000/0002000D
  4: 00000000/1992CA78 5: 00000000/000A1954
  6: 00000000/00000000 7: 00000000/198758E0
  8: 00000000/000A1E08 9: 00000000/000A34C6
  A: 00000000/000A1954 B: 00000000/19937B90
  C: 00000000/0009EA58 D: 00000000/000A4608
  E: 00000000/9992BBAE F: 00000000/00000000
  END OF SYMPTOM DUMP
  ```

  In message texts, an abend code can be called a *system completion code* or a *SYS CODE*. A message can show an abend code in the variable text without identifying it as an abend code; use the message explanation to understand the variable text.

- *VERBEXIT SYMPTOM output from a dump:* Format the dump completely, as described in step 1 on page 171 of "Gathering additional problem data for abends" on page 167. Look for AB/S0hhh and PRCS/hhhhhhhh in the symptoms.

  In the following VERBEXIT SYMPTOM output, the primary symptom string indicates a system abend code of X'03C' and a return code of X'2D000810'.

  ```
    Primary Symptom String:

      RIDS/NUCLEUS#L RIDS/IARYTASS PIDS/5752SC1CR AB/S003C RIDS/IARRR#R
      VALU/HC0099680 REGS/0E0B4 REGS/088FA PRCS/2D000810 VALU/CNAGEMENT

      Symptom            Symptom data       Explanation
      ---------------    -------------      -----------
      RIDS/NUCLEUS#L     NUCLEUS#L          Routine identifier
      RIDS/IARYTASS      IARYTASS           Routine identifier
      PIDS/5752SC1CR     5752SC1CR          Component identifier
      AB/S003C           003C               ABEND code - system
      RIDS/IARRR#R       IARRR#R            Routine identifier
      VALU/HC0099680     C0099680           Error related hexadecimal value
      REGS/0E0B4         0E0B4              Program register
      REGS/088FA         088FA              Program register
      PRCS/2D000810      2D000810           Return code
  ```

```
    VALU/CNAGEMENT     NAGEMENT          Error related character value
```

```
The dump does not contain a secondary symptom string.
```

- Dump title

  Look at the dump title; some titles contain the abend and reason codes. Use the DISPLAY DUMP,TITLE or DISPLAY DUMP,ERRDATA to display the dump title and any error information associated with captured dumps, or dumps written to pre-allocated or dynamically allocated data sets. In response to the DISPLAY command, message IEE853I or IEE854I are issued containing the requested information. Look in the IEE853I or IEE854I message replies for the abend code, reason codes and the registers. For example:

  ```
  IEE853I 12.54.26 SYS1.DUMP TITLES 939
  SYS1.DUMP DATA SETS AVAILABLE=000 AND FULL=002
  CAPTURED DUMPS=0000, SPACE USED=00000000M, SPACE FREE=00000200M
  DUMP00 TITLE=ABEND=S0C4,RC=0010,COMPON=SDSF-ESTAE,COMPID=5647-A01
  ,ISSUER=ISFSTAE,SDSF ABEND ROUTINE
  DUMP TAKEN TIME=17.49.48 DATE=nn/nn/nnnn
  DUMP01 TITLE=ABEND=S0C4,RC=0010,COMPON=SDSF-ESTAE,COMPID=5647-A01
  ,ISSUER=ISFSTAE,SDSF ABEND ROUTINE
  DUMP TAKEN TIME=nn.nn.nn DATE=nn/nn/nnnn
  ```

  For a dump that has been copied from the SYS1.DUMPxx data set or for an SVC dump you are viewing in IPCS, use the IPCS **LIST TITLE** subcommand to obtain the dump title.

You know you are done when you locate the abend and reason codes. You can then look up a description of the abend code using the product documentation or LookAt and follow the recommendations.

Related information:

- For LookAt and IBM product documentation, see www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/ and Chapter 22, "Diagnosis information for z/OS base elements and features," on page 285.
- For information about the IPCS STATUS FAILDATA subcommand, see *z/OS MVS IPCS Commands*.
- For information about the logrec data set, see *z/OS MVS Diagnosis: Tools and Service Aids*.
- For information about abend codes, see the product documentation. For example:
  - *z/OS MVS System Codes*
  - *z/OS UNIX System Services Messages and Codes*
  - *z/OS Communications Server: IP Messages Volume 1 (EZA)*
  - *z/OS Communications Server: IP Messages Volume 2 (EZB, EZD)*
  - *z/OS Communications Server: IP Messages Volume 3 (EZY)*
  - *z/OS Communications Server: IP Messages Volume 4 (EZZ, SNM)*.

Identifying the module and component

In some cases, the abend code indicates the source of the problem and how to repair it. However, there are cases when you cannot identify if the problem was caused by a z/OS, a vendor, or an installation problem. In this case, you must analyze the abend code to see which module or component was involved to conduct a more granular search for a known problem.

Steps for identifying the module and component

About this task

Before you begin: You need to know how to use IPCS and also have access to the following:

- SVC dump, SYSMDUMP, and SADUMP
- Job log, system log, OPERLOG or application error log.
- Master trace

You should also be able to locate and use:

- LookAt and IBM product documentation, see www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/ and Chapter 22, "Diagnosis information for z/OS base elements and features," on page 285.

Perform the following steps to identify the module that caused the abend and the offset of the failing instruction into the module.

Procedure

1. Do one of the following, depending on the problem data available.
   - For an **SVC dump**, obtain the component name, component identifier, and module name from the dump title.
   - **Obtain the PIDS/ccccccccc and RIDS/cccccccc** symptoms from the search argument. PIDS is the program identifier, which is the four character product identifier and the five character component identifier. RIDS identifies the module.

     If the search argument in an SVC dump does not identify the program and module or if the problem involves multiple abends, analyze the dump for the failing task or service request. See *z/OS MVS Diagnosis: Tools and Service Aids* for information about analyzing an SVC dump.
   - Obtain the address in the right half of the program status word (PSW) in **STATUS FAILDATA dump output**. The zArchitecture PSW is 128 bits (16 bytes). The combination of bits 31 and 32 show the current addressing mode. These are the low order bit of the left half and the high order bit of the right half of the PSW. The meanings are as follows:
     - 00 - 24-bit mode
     - 01 - 31-bit mode
     - 10 - Invalid
     - 11 - 64-bit mode

     In some places the PSW is shown in a shorter 64-bit (8 bytes) form. This indicates that the addressing mode is 24-bit or 31-bit and the PSW is shown in an ESA/390 form. In that case bit 31, the low order bit in the first half, shows the addressing mode:
     - 0 - 24-bit mode
     - 1 - 31-bit mode

     Subtract the instruction length from the PSW address to obtain the address of the failing instruction. Do not subtract the instruction length in the following cases; the failing instruction is at the PSW address.
     - Program check interruptions for which the processing of the instruction identified by the old PSW is nullified. See *z/Architecture Principles of Operation* for the interruption action. Some examples are:

- Page translation exception: `interrupt code = 0011`
- Segment translation exception: `interrupt code = 0010`
- Access register translation exception

  The following interrupt codes result in the operation being nullified:
  - LFX translation exception = 0026
  - LSX translation exception = 0027
  - ASX-translation exception = 0021
  - ASTE-validity exception = 002B
  - ASTE-sequence exception = 002C
  - LSTE sequence exception = 002E
  - ASTE instance exception = 002F
- Region-first-translation exception
- Region-second-translation exception
- Region-third-translation exception

– Obtain the PSW and registers for the error from the **STATUS CPU REGISTERS** subcommand.

  In the following STATUS CPU REGISTERS output, the address can be found in the second half of the PSW. Note that this presentation uses both the first (bit 32) and last (bit 63) bits in the PSW shown combine to indicate addressing mode. Bit 64 will be on when the PSW represents code running in 64 bit mode.

```
CPU STATUS:
 PSW=070C1000  83D00B72 (RUNNING IN PRIMARY, KEY 0, AMODE 31, DAT ON)
     DISABLED FOR PER
   ASID(X'0015') 03D00B72. DATSVY02+03CA IN EXTENDED PRIVATE
  ASCB21 at F9CD80, JOB(DAESVY01), for the home ASID
  ASXB21 at 6FE038 for the home ASID. No block is dispatched
  HOME ASID: 0015 PRIMARY ASID: 0015 SECONDARY ASID: 0015
  GPR VALUES
      0-3  00000000  03D017B0  00000000  03D01A12
      4-7  03D00EC1  03D00CE8  006D4FF8  FD000000
      8-11 03D025BF  83D007A8  03D015C0  03D017A7
     12-15 03D01830  03D015C0  03D019EB  03D00DA9
IEA11015I The requested ALETs are zero.
  CONTROL REGISTER VALUES
      0-3  5EB1EE40  00C0407F  002B5040  00800015
      4-7  00000015  01756540  FE000000  00C0407F
      8-11 00000000  00000000  00000000  00000000
     12-15 01F7C27F  00C0407F  DF881755  7F704008
THE PRECEDING STATUS CPU INCLUDED THE REGS OPTION
```

  Example for for 31-bit:

```
PSW: 040C0000 816B65A6 Instruction Length: 04 Interrupt Code: 0011
Failing instruction text: 58F0C030 50F0B222 5BF0B240
Translation Exception Address: 00000000_7F37B003
```

  Example for 64-bit:

```
Time of Error Information:
PSW: 04046001 80000000 00000000 0178F356
Instruction length: 04 Interrupt code: 0004
Failing instruction text: 000A5023 00005032 00044172
```

  STATUS CPU REGISTERS supplies the name of the failing module and its offset without having to do a separate WHERE subcommand.

2. Do one of the following:

  - If analyzing the dump interactively, use the instruction address in an IPCS **WHERE** subcommand to obtain the name of the load module and the offset of the address into the load module. If the module name is not proceeded with IEANUC01, then IPCS has given the load module name. If you enter the **STATUS CPU REGISTERS** subcommand, a **WHERE** is performed automatically.

    Use the AMBLIST service aid to list the CSECTs in the load module. Use the offset into the load module to identify the CSECT containing the failing instruction. Then subtract the starting address of the CSECT from the instruction address to obtain the offset into the CSECT.

    For instructions on using the AMBLIST service aid, see AMBLIST in *z/OS MVS Diagnosis: Tools and Service Aids*.
  - If the **WHERE** command does not present a module name, follow this procedure:
    - Using the IPCS Browse panel, enter the PSW instruction address and ASID obtained from the time of error information. Browse backwards looking for the CSECT name eye-catcher. IBM module names are generally followed by an assembly date and a product identifier or PTF level, such as HBB7720; most eye-catchers are at the beginning of a module, but some are at the end.

3. The module prefix identifies the component, subsystem, or product, if provided by IBM. See the module identification chart in *z/OS MVS Diagnosis: Reference*.

   For example, using the information in the following output from you can determine what component was involved in an error from the module prefix. The ADY module prefix indicates a DAE-type error.

```
Module      Component                         Product   Component
Prefix        Name                              ID          ID
 ADF      TSO and TSO/E session manager       5665     28505
 ADY      Dump analysis and elimination (DAE)  5752     SC143
```

4. Continue diagnosis as follows:
   - For a z/OS component, continue with "Steps for searching the problem reporting databases" on page 167.
   - For an IBM subsystem or product, continue diagnosis with the subsystem or product. See Chapter 22, "Diagnosis information for z/OS base elements and features," on page 285 for a listing of components and products.
   - For an installation-provided program, including an installation exit routine, continue diagnosis with that program, using the dump for the abend.

Results

Related information:

- See "Searching problem reporting databases" on page 8 for more information.
- See *z/OS MVS Diagnosis: Tools and Service Aids*for dump analysis of a problem in an installation-provided program.
- See *z/OS MVS IPCS Commands* for information about IPCS subcommands.

Searching the problem reporting databases

Search arguments are used to search problem reporting databases. If the problem you are diagnosing was already reported and the symptoms are in the database, the search produces a match. Searching is an iterative process; you might need to gather additional data and continue your search.

Steps for searching the problem reporting databases

1. Search the problem reporting database to determine if the problem was previously reported. See "Searching for a known problem" on page 10.

   Use the free-format search argument abstract as a symptom string to determine if the problem already exists. For example `ABEND05C RSN00000241`.

   For example, the following search argument abstract would generate the free-format search argument: `ABEND03C RSN2D000810:`

   ```
   SEARCH ARGUMENT ABSTRACT

     PIDS/5752SC1CR RIDS/NUCLEUS#L RIDS/IARYTASS AB/S003C PRCS/2D000810 REGS/0E0B
     REGS/088FA RIDS/IARRR#R
   ```
2. If the search provides no match, remove some symptoms to broaden the search. If the search provides too many symptoms, trying adding some symptoms to limit the scope. Check to see which matches pertain to the operating system environment.
3. If the search finds that the problem was previously reported, request the problem fix. If not, continue with "Gathering additional problem data for abends." Use the problem data gathered there to create more symptoms; use these symptoms in later searches.
4. If you still cannot find the cause of the abend or if the problem is new, report the problem to IBM.

   Provide the information in Chapter 24, "Problem diagnostic worksheet," on page 291, such as:
   - Any changes made to the system recently, preceding when the problem began occurring (for example, PTFs or new products installed or new hardware).
   - Problem type: abend
   - Search argument
   - Dump, formatted by IPCS, online or printed
   - Failing input request: macro, command, or statement
   - SDWAVRA keys, lengths, and contents
   - Offset of the failing instruction into the module or CSECT
   - Accompanying messages: identifiers and texts
   - Logrec report, if used
   - All printed output and output data sets related to the problem
   - Data on any related problems
   - Module name and level
   - Name and level of the operating system(s) with a list of program temporary fixes (PTF) applied at the time of the problem and all installation modifications, exits, and products with other than Class A service
   - Other problem data developed while using the diagnosis book for the component, subsystem, or program

Gathering additional problem data for abends

About this task

Gathering additional data will increase your chances of finding a match in the problem reporting databases. Use the procedures outlined in this section to create more symptoms; use these symptoms in later searches.

Steps for gathering additional data for abends

It is important to gather the following information:

- The impact of the problem to system or sysplex
- The names of jobs, functions or programs that were running at the time of the error
- The existence of any new software maintenance or hardware changes
- The associated messages from job log, SYSLOG, or OPERLOG
- The related component traces that are active.

This procedure is divided into the following sections:

- "Steps for gathering trace data for abends"
- "Steps for collecting additional messages and logrec for abends" on page 169
- "Steps for obtaining a dump for the error" on page 171

Before you begin any of these tasks: Complete the steps in Chapter 10, "Diagnosing an abend," on page 155.

Steps for gathering trace data for abends

Use the following steps to gather additional data from system trace table:

1. Analyze the system trace table, which is formatted by the **SYSTRACE CURRENT ERROR** subcommand. A system trace provides a record of system events. Use it to create a picture of the processing occurring at the time of the error.
2. Starting at the end of the trace, back up to the entry for the abend being diagnosed:
   - *SVC D or *SVCE D in the IDENT CD/D columns
   - The abend code in the right 3 bytes in the UNIQUE-3 column
   - The reason code in the UNIQUE-1 column

   Example: In the following SYSTRACE output, the ***SVC D** indicates that an abend code has been loaded for processor **02**. When examining system trace output, look for **RCVY** entries that represent entry into a recovery routine following an error or interruption.

   ```
   PR ASID TCB-ADDR  IDENT CD/D PSW----- ADDRESS-...
                     .
                     .
                     .
   02 000D 006F8E88   SSRV  12D          8120BFD8  006F8E88 000B0000 00000000
                                                   00000000
   01 002E 006BEE88   SVC    30 070C3000 827FAF36  00000000 00000001 072CFBF4
   01 002E 006BEE88   PC    ...   0      81157326  00100
   01 002E 006BEE88   PT    ...   0      811B782C  002E
   02 000D 006F8E88   DSP       070C0000 8101A9B0  00000000 0001035C 000295A8
   02 000D 006F8E88  *SVC     D 070C0000 8101A9B2  80D12090 0001035C 000295A8
   02 000D 006F8E88   PC    ...   0      811D7626  00506
   01 002E 006BEE88   SVCR   30 070C3000 827FAF36  00000000 00000028 0080002E
   01 002E 006BEE88   SVC    38 070C3000 827FAF64  00000000 00000028 072CFBF4
   01 002E 006BEE88   SVC     A 070C1000 80F8146A  00000000 FD000236 80F81468
   ```

   For system trace, when viewing program checks, look for the PGM entry just before a RCVY entry.

   ```
   06 00C4 009FF540 PGM 004 078D3400 A930BD12 00040004 00000000 00000000
   06 00C4 009FF540 *RCVY PROG 940C4000 00000004 00000000
   ```

3. In the entry, note the processor in the PR column, the address space identifier in the ASID column, and the task control block (TCB) address in the

TCB-ADDR column. The ASID should be the same as the ASID identified in STATUS FAILDATA or STATUS CPU output.

4. Continue backing up, looking for the following entries:
   - The entry for the system service being processed when the abend occurred, if the abend occurred during system processing. This entry will have SVC, SSRV, or SVCE in the IDENT column and the same ASID as the abend entry.
   - Problem entries, which have an asterisk (*) before the identifier in the IDENT column.
   - Other entries for the same processor, PR column.
   - Other entries for the same address space, ASID column.
   - Repeated requests by a program for one or more system services. This pattern indicates a loop. An enabled loop has multiple types of entries before the pattern repeats. Continue diagnosis with the program requesting the system services.

You should now be able to determine the source of the abend or have more information to search the problem reporting database.

Related information

- For information about IPCS subcommands: SELECT, SUMMARY, VERBEXIT LOGDATA, and VERBEXIT MTRACE, see *z/OS MVS IPCS Commands*.
- For the format and contents of the CDE, RB, RTM2WA, SDWA, VRAMAP (VRA keys), and TCB, see the version of *z/OS MVS Data Areas* that corresponds to the release you are running in your environment. See the z/OS library .
- For the formats of system trace entries, see *z/OS MVS Diagnosis: Tools and Service Aids*.
- For information about the SLIP command, see *z/OS MVS System Commands*.
- See *EREP User's Guide* for formatting logrec records.
- See *z/OS MVS System Messages, Vol 6 (GOS-IEA)* for message IEA995I.
- See *z/OS MVS System Messages, Vol 9 (IGF-IWM)* for the IOS messages.
- For the PGM parameter, see *z/OS MVS JCL Reference*.
- For interactive TSO/E commands, see *z/OS TSO/E Command Reference*.

Steps for collecting additional messages and logrec for abends

Use the following steps to gather additional messages and logrec:

1. Collect and analyze messages and logrec records about the problem. Use the ERRORID from the dump message and time stamps to select messages and software, symptom, and hardware logrec records related to the problem. Look in the following:
   - The job log
   - A TSO/E user's ISPF transaction log or session manager log
   - The hardcopy log, also known as the system log (SYSLOG)
   - VERBEXIT MTRACE dump output, which shows the buffer for system messages
   - VERBEXIT LOGDATA dump output, which formats the logrec buffer
   - The logrec data set, formatted by EREP
2. Look for the following:
   - Symptom dump message IEA995I for a previous, related abend
   - Messages identifying a failing program with a nonzero return code
   - I/O error messages

3. Identify the program being processed when the abend occurred by obtaining the job name from the following:
   - SUMMARY output
   - VERBEXIT LOGDATA output
   - Messages in the job log
   - Messages in VERBEXIT MTRACE output
   - SELECT output

     For example, in the following SELECT output, the job name NVAST in address space 0073 contains an error.

     ```
     ASID JOBNAME  ASCBADDR  SELECTION CRITERIA
     ---- -------- --------  ------------------
     0073 NVAST    00F6B600  CURRENT ERROR
     ```
4. If a batch job was being processed, obtain the program name from the PGM parameter on the JCL EXEC statement.

   In the following output, the PGM parameter of JCL statement indicates that the program name is UNIONE.

   ```
   //BANK1 EXEC PGM=UNIONE,PARM='@PLANID=1,10S,SHR',
   //            REGION=1024K,COND=(8,LE)
   //BANKLOG  DD  DSN=NULLFILE,DISP=SHR
   ```
5. If interactive work was being processed, use the command being processed to identify the program.
6. Analyze the problem data for multiple problems. Collect data for related problem ERRORIDs that occur in a similar time frame. You can find this data in the logrec data set. The time stamps are a few seconds before or after the time stamp for the abend being diagnosed. The data involves the following:
   - The same job step
   - The same task (TCB) or service request (SRB)
   - The same home address space (ASID)
   - The address spaces involved in cross-memory mode processing
   - The same processor (CPU), if the problem occurred while the system was disabled for input/output (I/O) and external interrupts (EXT), as indicated in STATUS CPU dump output.

   Look for the following:
   - In the output from IPCS **SUMMARY TCBERROR**, look at the task completion codes in job step program TCB CMP fields; a nonzero completion code indicates an abend. You are looking at the correct abend if it has an associated RTM2WA. If a related task abended seconds before the abend being diagnosed, check the task's CDE, RTM2WA, and SDWA control blocks for the module name and other data about the abend. The output contains one RTM2WA for each abend being processed.

     In the following SUMMARY TCBERROR output, the nonzero CMP field of the TCB indicates an error.

     ```
     TCB: 009F3E88
        CMP...... 940C9000  PKF...... 80       LMP...... FF       DSP...... FF
        TSFLG.... 00        STAB..... 009FD200  NDSP..... 00002000
        JSCB..... 009FF40C  BITS..... 00000000  DAR...... 00
        RTWA..... 00000000  FBYT1.... 00
        Task non-dispatchability flags from TCBFLGS5:
         Secondary non-dispatchability indicator
        Task non-dispatchability flags from TCBNDSP2:
         SVC Dump is executing for another task
     ```
   - In VERBEXIT LOGDATA output or the logrec reports and in messages in all locations, look for previous abends and symptom records for earlier

problems that did not cause abends. The previous abend or the earlier problem might have led to the abend being diagnosed.

- Look for the name of the program that called the abending module.
  - The address of the calling program can be in the second half of the PSW stored in the caller's RB, which will precede the running RB, except for branch entries.
  - Determine the linkage conventions of individual save areas. The calling program's address might be in a save area.
  - In register 14 of the top RB.
  - If a command or macro was being processed, obtain the name of the module issuing the command or macro. The name is in the NAME field of the CDE for a request block (RB) for the abending module's TCB.

  Check for problems in the calling program. The calling program might have caused the abend being diagnosed.

Investigate the following:

- Many abends relating to the same area of the system.
- Many TCBs with the same abend code.

You should now be able to identify the program causing the abend. If not, go to the next step.

Steps for obtaining a dump for the error

Use the following steps to obtain a dump and collect additional data using IPCS:

1. If a dump was not written for the abend, recreate the problem and obtain a dump by doing one of the following:
   - Set a SLIP command to obtain an SVC dump. For example:

     ```
     SLIP SET,C=0C9,JOBNAME=RMF,A=SVCD,END
     ```

     This **SLIP** trap will request a dump when an ABEND0C9 occurs in the RMF address space. For more information, see the topic on the SLIP command in *z/OS MVS System Commands*.
   - Insert a SYSABEND, SYSUDUMP or SYSMDUMP DD statement in the JCL for the abending job step to obtain an ABEND dump. For more information, see *z/OS MVS JCL Reference*.
2. Use IPCS to look at the dump. Use IPCS subcommands in the order indicated by the following list. If using IPCS interactively for an SVC dump, respond yes to the IPCS message that asks if summary data can be used by dump access.
   a. STATUS FAILDATA
   b. STATUS SYSTEM

      Example: In the following STATUS SYSTEM output, AMDSADMP indicates that this dump was scheduled. Also note the date and time the dump was taken.

      ```
      SYSTEM STATUS:
        Nucleus member name: IEANUC01
      I/O configuration data:
           IODF data set name: IODF.IODF12
           IODF configuration ID: TC4SYST
           EDT ID: 00
        Sysplex name: ENGTEST2
        TIME OF DAY CLOCK: BE5E67AF 7ED6370E  02/15/2006 14:33:32.124515 GMT
        TIME OF DAY CLOCK: BE5E24A1 5B96370E  02/15/2006 09:33:32.124515 local
        Program Producing Dump: SADUMP
        Program Requesting Dump: AMDSADMP
      ```

c. STATUS CPU REGISTERS DATA CONTENTION
d. STATUS WORKSHEET
e. SUMMARY FORMAT
f. VERBEXIT LOGDATA
g. VERBEXIT SUMDUMP
h. SYSTRACE
i. VERBEXIT MTRACE

   Example: In the following VERBEXIT MTRACE output, message IEF450I indicates a system abend of X'522' with a reason code of X'00'.

```
0001 007A5F54  N 0000000 ANO3     93039 10:26:08.31          00000281
               IEA989I SLIP TRAP ID=X13E MATCHED
0001 007A5F54  N 0000000 ANO3     93039 10:26:08.34          00000281
               IEA989I SLIP TRAP ID=X13E MATCHED
0001 007A5F54  N 0000000 ANO3     93039 10:26:08.43          00000281
               IEA989I SLIP TRAP ID=X13E MATCHED
0001 007A5F54  N 0000000 ANO3     93039 10:26:08.49          00000281
               IEA989I SLIP TRAP ID=X13E MATCHED
0001 007A5F54  N 4000000 ANO3     93039 10:26:09.21 TSU05807 00000091
               IEF450I LASSEC2 AAIRACF AAIRACF - ABEND=S522 U0000
               REASON=00000000
0001 007A7430  N 4000000 ANO3     93039 10:26:09.45 TSU06038 00000091
               IEF450I LAMMLF AAIUSER AAIUSER - ABEND=S522 U0000
               REASON=00000000
0001 007A7430  M 4000000 ANO3     93039 10:26:09.59 TSU05807 00000090
               IEF377I LASSEC2 AAIRACF AAIRACF
0001 007A5F54  E                                         064 00000090
               LASSEC2.SPFLOG1.LIST NOT CATLGD 2
0001 007EC02C  N 4000000 ANO3     93039 10:26:09.66 TSU05807 00000090
               /HASP395 LASSEC2 ENDED
0001 007A79C0  N 0200000 ANO3     93039 10:26:10.06 TSU05807 00000081
               /HASP250 LASSEC2 IS PURGED
```

j. Subcommand selected from the list in Table 18.
k. VERBEXIT SYMPTOM

3. Before the **VERBEXIT SYMPTOM** subcommand, add other IPCS subcommands, depending on the problem indicated in the abend explanation or accompanying messages. Pick the subcommands from the following list:

Table 18. Summary of IPCS dump subcommands by problem

Problem involves	IPCS dump command
Allocation/unallocation of jobs	VERBEXIT ALCMWAIT
Asynchronous operations manager (AOM)	VERBEXIT AOMDATA 'TRCDUMP'
Auxiliary storage	ASMCHECK VERBEXIT ASMDATA
Availability management	VERBEXIT AVMDATA
Callable service requests	CBFORMAT addr STRUCTURE(CSRCPOOL) CBSTAT addr STRUCTURE(CSRCPOOL)
Communications	COMCHECK
Cross-system coupling facility	COUPLE SUMMARY ALL
(XCF)	COUPLE DETAIL ALL COUPLE EXCEPTION ALL
Data-in-virtual	DIVDATA SUMMARY ALL

Table 18. Summary of IPCS dump subcommands by problem (continued)

Problem involves	IPCS dump command
Data lookaside facility of VLF	DLFDATA
Global resource serialization	VERBEXIT GRSTRACE
Input/output	IOSCHECK
JES2	VERBEXIT JES2
JES3	VERBEXIT JES3
Language Environment	VERBEXIT LEDATA
MVS message service (MMS)	VERBEXIT MMSDATA
z/OS UNIX System Services (OMVS)	OMVSDATA SUMMARY OMVSDATA DETAIL OMVSDATA EXCEPTION
Real storage manager (RSM)	RSMDATA SUMMARY RSMDATA EXCEPTION
System resources manager (SRM)	VERBEXIT SRMDATA
Storage Management Subsystem (SMS)	VERBEXIT SMSDATA
Time sharing option (TSO)	VERBEXIT TSODATA
Virtual storage manager (VSM)	VERBEXIT VSMDATA
Virtual lookaside facility (VLF)	VLFDATA

4. Use the RSMDATA SUMMARY output to get a summary of real storage usage in the system. Use the RSMDATA EXCEPTION report to determine where errors might have occurred. The following is an example of RSMDATA SUMMARY output:

```
                R S M   S U M M A R Y   R E P O R T


                          Tot real       Prf real       Below Prf     ...
                          ------------- ------------- ----- -----    ...
In configuration . . . .       131,072         98,234  4,096 4,026    ...
Available for allocation       126,084         93,247  4,093 4,023    ...
Allocated  . . . . . . .        54,127         53,253    184   118    ...
Percent usage  . . . . .            42             57      4     2    ...
Common fixed frames  . .         3,291          3,145     19    19    ...
Percent of available .               2              3      0     0    ...
Total fixed frames . . .         8,283              -     28     -    ...
Percent of available .               6              -      0     -    ...


V=R Region:
First frame number X'00006'
Last frame number X'0004B'
Size (in frames)              70

Total disabled reference (DREF) pages in real:          2,309

Number of shared data pages:
Valid and fixed in real  . .             3
Valid and pageable in real .         1,356
On auxiliary storage . . . .             0

Number of 64-bit common memory pages:
Backed in real . . . . . . .                     513
Fixed in real  . . . . . . .                     144
DREF in real . . . . . . . .                     256
On auxiliary storage . . . .                       0
```

Figure 39. RSMDATA SUMMARY report

5. Examine the VRADATA output in the STATUS FAILDATA, VERBX LOGDATA or EREP report for an error for additional clues about the error. For some components, the data consists of a key, a length, and the contents.

   Example: In the following Variable Recording Area from STATUS FAILDATA output, the VRA key is X'1A' and the length is X'94'.

   ```
   VARIABLE RECORDING AREA (SDWAVRA)

     +000    Key: 1A     Length: 94
     +002    02000000    08004000    00040001    1D00E610     |...... .......W.|
     +012    E0001300    00000000    00000000    00000000     |\...............|
     +022    00000000    00000000    00000000    00000000     |................|
     +032    00000000    00000000    00000000    0000002B     |................|
     +042    00100000    00000000    00000000    00000000     |................|
     +052    00000000    00000000    00000000    FFFC0000     |................|
     +062    012DBD64    00000000    01A9CD24    00000C80     |.........z......|
     +072    00000000    019D6690    012DC698    00000000     |..........Fq....|
     +082    01B42000    00000000    00000000    00000000     |................|
     +092    00000000                                         |....            |
   ```

   STATUS FAILDATA will not format an SDWA for a dump requested by SLIP. If SDWA data is not in the dump, obtain problem data from STATUS CPU REGISTERS or view the SDUMP 4K SQA buffer. (See Reading the SDUMPX 4K SQA buffer in *z/OS MVS Diagnosis: Tools and Service Aids*.)

You should now have extracted enough problem data to do a search for a known problem, identify the source of the problem, or report the new problem to IBM or the appropriate vendor.

Related information

- For information about the SYSMDUMP DD, see
  - *z/OS MVS Diagnosis: Tools and Service Aids*
  - *z/OS MVS IPCS Commands*
  - *z/OS UNIX System Services Planning*
  - *z/OS MVS JCL Reference*.
  - *z/OS MVS System Commands*

Chapter 11. Diagnosing a system hang or wait state

- "Overview of a hang or wait" includes descriptions of system hangs and wait states and the symptoms you might encouter
- "Steps for diagnosing a system hang" on page 178 contains a flowchart and these steps to guide you through diagnosis of an hang or wait state:
  1. "Collecting the problem description" on page 181
  2. "Diagnosing a hang or wait during IPL" on page 182
  3. "Diagnosing an enabled wait state" on page 184
  4. "Diagnosing a coded disabled wait state" on page 186
  5. "Diagnosing a system partitioned from a sysplex because of status update missing" on page 188
  6. "Searching the problem reporting databases" on page 188
  7. "Gathering additional data for hangs and waits" on page 190

Overview of a hang or wait

A system hang or wait can occur gradually as a resource contention problem or abruptly when a disabled wait state is loaded for a critical software-detected error. Externally, the following list of symptoms might be noticed:

- A disabled coded wait state is loaded
- A hang during IPL or system initialization
- The consoles can be locked
- There can be contention for system resources
- The system code can be looping.

Note: If the problem is contention or system code looping, use Runtime Diagnostics to diagnose and possibly solve the problem before continuing these steps. See "Runtime Diagnostics symptoms" on page 39 in Chapter 4, "Runtime Diagnostics," on page 35.

When there is a system failure or outage, a stand-alone dump must be taken for problem diagnosis. OPERLOG, SYSLOG, and EREP reports from the time frame of the system outage are also important.

This section will only discuss system hangs and waits. When a job or subsystem is hung, see Chapter 12, "Diagnosing a job or subsystem hang," on page 193.

Symptoms of a wait or hang: The system enters a wait or the entire system hangs. The terms hang and wait are used synonymously in these procedures. Some symptoms of a hang:

- No response occurs on the user's or system operator's console.
- No communication with the system through the console occur.
- No response from subsystems (TSO/E, CICS, IMS™, DB2, and others) occur.
- The system does not issue or receive messages on the console.
- A series of messages that indicate waits followed by bursts of activity.
- A message indicating a wait appears on the system console.
- The program status word (PSW) contains X'070E0000 00000000'.

- The job entry subsystem does not respond to any commands. For example, in a JES2 system, enter a $DI1 command and JES2 does not respond.

There are two types of wait states: enabled and disabled.

Enabled wait
The system stops processing without issuing a wait state code when the dispatcher did not find any work to be dispatched.

A special type of enabled wait is called a **no work wait** or a **dummy wait**. An indication of a dummy wait or no work wait is a PSW of X'070E0000 00000000' and GPRs containing all zeroes. Diagnosis is required for this type of wait only when the system does not resume processing.

The most common causes of an enabled wait are that the system is waiting for:

- Work – the system has no active jobs to process or all active jobs are swapped out.
- Action – an operator reply or other action.
- Missing interrupts – the system is waiting for a critical device, which is busy, not ready, reserved by another system, or has a mount pending. If the system residence (SYSRES) or paging (PAGE) volumes have missing interrupts, the operator may not get a message.
- System resource – work is waiting for a resource, which can be a lock, queue, input/output (I/O) device, page, or device allocation.

Disabled wait with a wait state code
The system issues a wait state code and stops. The operator can see the wait state code on the system console. This wait is called a **coded wait state** or a **disabled wait**. There are two types of disabled wait state codes:

restartable wait state
You can restart the system.

A restartable wait is one of the following:

- An attempt by the operating system to communicate with the operator. When the system cannot send a message to a console, the system can use a restartable wait state to contact the operator and obtain a response.
- A way to preempt processing. For a SLIP trap with an action of wait, the system will issue a message, then enter a restartable wait.
- A symptom of another problem.

non-restartable
You cannot restart the system. After capturing a stand-alone dump, you must reIPL the system.

Steps for diagnosing a system hang

About this task

You must know how to use IPCS and have access to the following types of information:

- Stand-alone dump
- EREP report of SYS1.LOGREC

- OPERLOG or SYSLOG
- The level of z/OS operating system. Use the IPCS CBFORMAT CVT command to find the level of the z/OS.

  The following is an example of the CVT output:

```
********************************************************* TOP OF DATA*************************************************************
CVT: 00FD4938

-0028  PRODN.... SP7.0.6   PRODI.... HBB7709   VERID....                               MDL...... 2064      RELNO.... 038
+0000  TCBP..... 00000218  0EF00.... 00FF24EC  LINK..... 00FD48B4  AUSCB.... 00FD4F20  BUF...... 00000000  XAPG..... 00FDE310
+0018  0VL00.... 00FF63DE  PCNVT.... 00FE0CD4  PRLTV.... 00FE0B04  LLCB..... 018E50F0  LLTRM.... 8146E288  XTLER.... 00FE6D10
+0030  SYSAD.... 00EEA898  BTERM.... 00FEF820  DATE..... 0106114F  MSLT..... 00FD4F48  ZDTAB.... 00DAD000  XITP..... 00FF9740
+0048  0EF01.... 00FF250C  VSS...... 0000      VPSM..... 0000      EXIT..... 0A03      BRET..... 07FE      SVDCB....
```

- The state of the system. Use IPCS **STATUS CPU** and note the PSW and mode of each CPU. **For example:**

```
CPU(X'00') STATUS:
PSW=07060000 00000000 00000000 00000000
    No work wait
  CPU is in no work wait
```

Normally, a wait state code appears in the program status word (PSW) when the operating system enters a wait state. Use this code and the associated reason code to diagnose and fix the problem. Explanations for wait state codes are found in *z/OS MVS System Codes*.

The following steps will guide you through diagnosing a hang or wait:

1. "Collecting the problem description" on page 181
2. "Diagnosing a hang or wait during IPL" on page 182
3. "Diagnosing an enabled wait state" on page 184
4. "Diagnosing a coded disabled wait state" on page 186
5. "Diagnosing a system partitioned from a sysplex because of status update missing" on page 188
6. "Searching the problem reporting databases" on page 188
7. "Gathering additional data for hangs and waits" on page 190

Use the following flowchart to guide diagnosis of a system hang:

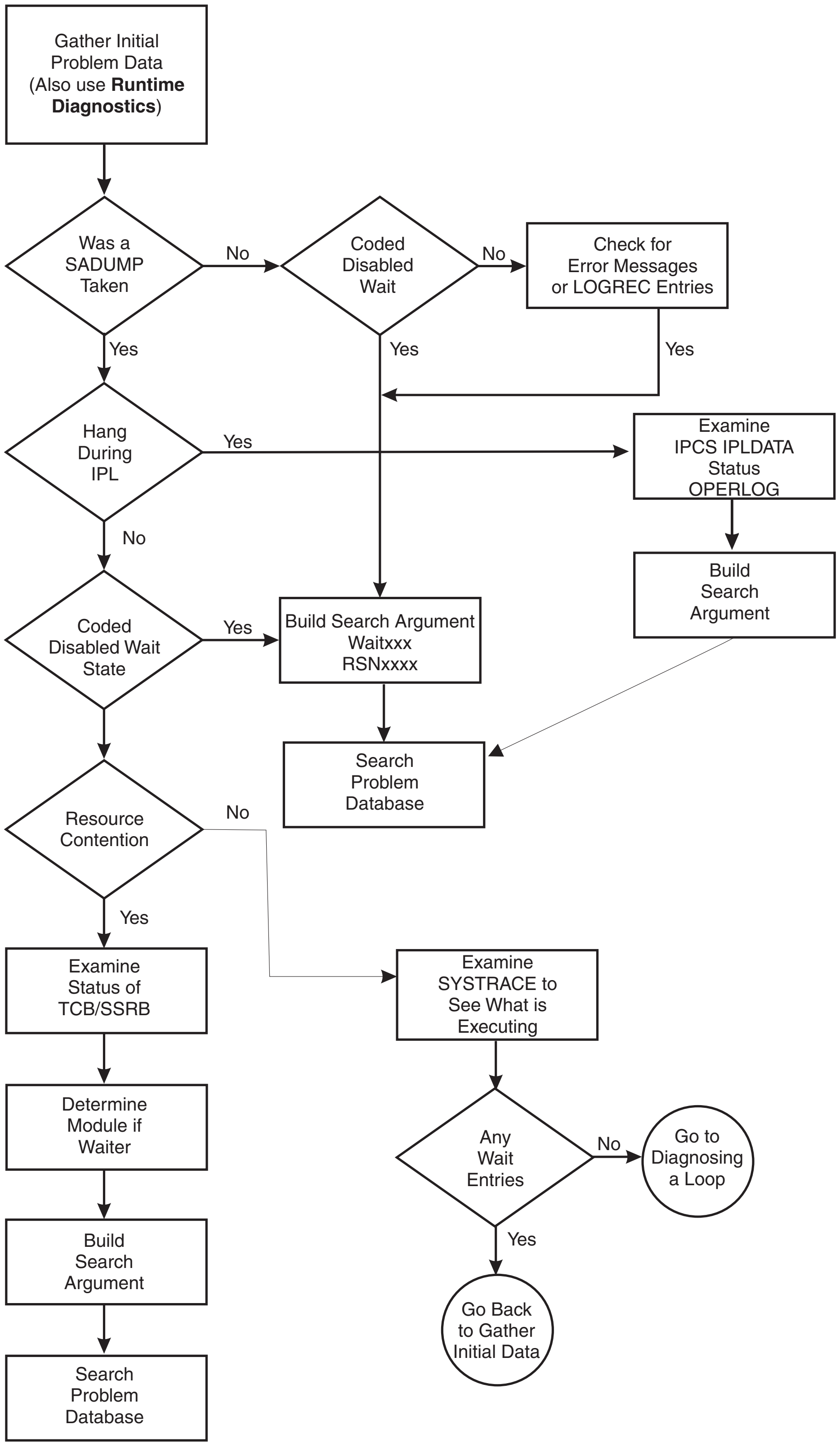

Gather Initial Problem Data (Also use **Runtime Diagnostics**)
Was a SADUMP Taken
No
Coded Disabled Wait
No
Check for Error Messages or LOGREC Entries
Yes
Yes
Yes
Hang During IPL
Yes
Examine IPCS IPLDATA Status OPERLOG
No
Build Search Argument
Coded Disabled Wait State
Yes
Build Search Argument Waitxxx RSNxxxx
Search Problem Database
Resource Contention
No
Yes
Examine Status of TCB/SSRB
Examine SYSTRACE to See What is Executing
Determine Module if Waiter
Any Wait Entries
No
Go to Diagnosing a Loop
Build Search Argument
Yes
Go Back to Gather Initial Data
Search Problem Database

Figure 40. Flowchart for system hang analysis

Collecting the problem description

About this task

The problem descriptions found in “Gathering diagnosis data” on page 5 and Table 1 on page 7 indicate you have a hang, a disabled wait state, or an enabled wait state that needs diagnosis. Before using this procedure, if possible, use Runtime Diagnostics to identify and solve the problem (see Chapter 4, “Runtime Diagnostics,” on page 35).

Steps for collecting the problem description

Perform the following steps to collect the problem description:

1. Ensure that the symptom descriptions in “Overview of a hang or wait” on page 177 and in Table 1 on page 7 identify the problem is a hang.

   If you see the system activity on the console is high and no jobs are being processed, the problem is a loop. Use the procedure in Chapter 13, “Diagnosing a loop,” on page 203.
2. Describe what was happening on the system prior to the hang or wait and record this information in Chapter 24, “Problem diagnostic worksheet,” on page 291. This includes:
   - What is the status for the system on the Hardware Management Console (HMC)?
   - What jobs were started just prior to the hang?
   - What commands were entered and responses received?
   - What recovery procedures did you attempt?
   - What error messages were received?
   - Were there environmental changes? For example, was a new device installed or software maintenance applied.
   - Was the impact to a subsystem like DB2 or the entire system workload?
3. Did the hang during IPL or system initialization? If yes, go to “Diagnosing a hang or wait during IPL” on page 182.
4. Determine the state of the system by entering the state IPCS **STATUS CPU** command. Note the PSW for each CP.
   a. If every CP is showing a no work wait, go to “Diagnosing an enabled wait state” on page 184. **For example:**
      ```
      CPU(X'00') STATUS:
      PSW=07060000 00000000 00000000 00000000
          No work wait
        CPU is in no work wait
      ```
   b. If any CP is showing a disabled coded wait state, go to “Diagnosing a coded disabled wait state” on page 186. **For example:**
      ```
      CPU(X'01') STATUS:
      PSW=000A0000 800200A2
          Disabled wait state code 00A2  SUPPLMNT INFO 80020
      Wait occurred because system monitor control
      information cannot be read or written.
        ASCB6 at F42700, JOB(XCFAS), for the home ASID
        ASXB6 at 5FDE88 and TCB6G at 5FF500 for the home ASID
        HOME ASID: 0006 PRIMARY ASID: 0006 SECONDARY ASID: 0006
      ```
   c. If the CP is not showing a no work wait or a coded disabled wait, start diagnosis by checking for resource contention. Go to “Diagnosing an enabled wait state” on page 184.

Diagnosing a hang or wait during IPL

If a hang or wait occurs during IPL or early on during system initialization, obtain a stand-alone dump, SYSLOG, and note the last message issued to the screen or log. The objectives for analyzing the output of a stand-alone dump are:

- Gather symptom data.
- Determine the state of the system.
- Analyze the preceding system activity.
- Find the failing module and component.

Steps for diagnosing a hang or wait during IPL

1. Enter the IPCS **IPLDATA STATUS** command to determine how far along the system is in the IPL or nucleus initialization program (NIP) processing. There is an entry for each initialization routine. The last entry indicates the last initialization routine to run. Use the module name in a search for a known problem. The following example indicates `IEAIPL99 ... Page frame table and cleanup` as the last entry.

```
*** IPL Statistics ***

IEAIPL10  00:00:00.000  ISNIRIM - Read SCPINFO
IEAIPL20  00:00:01.688  Test Block storage to 2G
IEAIPL11  00:00:00.018  Fast FIND service
IEAIPL31  00:00:00.002  LOAD service
IEAIPL30  00:00:00.000  IPLWTO service
IEAIPL46  00:00:00.164  Read SCHIBs into IPL workspace
IEAIPL49  00:00:00.000  Process Load and Default parameters
IEAIPL50  00:00:00.774  IPL parmlib - process LOADxx and NUCLSTxx
IEAIPL51  00:00:00.019  System architecture
IEAIPL43  00:00:00.032  Find and Open IODF data set
IEAIPL60  00:00:00.008  Read NCRs from IODF
IEAIPL70  00:00:00.208  UIM environment - load CBD and IOS services
IEAIPL71  00:00:00.120  Build DFT for each device
IEAIPL08  00:00:00.007  Read EDT information from IODF
IEAIPL40  00:00:00.093  Read MLTs from nucleus
IEAIPL42  00:00:00.018  Read NMLs from nucleus (IEANynnn modules)
IEAIPL41  00:00:01.388  Read PDS directory entries and CESD records
IEAIPL05  00:00:01.056  Build and sort NUCMAP
IEAIPL02  00:00:03.779  Load nucleus modules
IEAIPL04  00:00:00.020  Allocate PFT and SQA/ESQA
IEAIPL14  00:00:00.000  Build LSQA/ELSQA for Master
IEAIPL06  00:00:00.000  IARMI - RSM blocks, master SGT
IEAIPL09  00:00:00.054  IAXMI - PFT, master RAB,  etc.
IEAIPL07  00:00:00.037  Update AMODE for nucleus resident SVCs
IEAIPL03  00:00:00.027  Build UCBs, ULUT, etc.
IEAIPL18  00:00:00.172  Copy and relocate EDT to ESQA
IEAIPL99  00:00:00.465  Page frame table and cleanup

Total IPL Time:  00:00:10.162
```

Figure 41. IPL statistics example

The following NIP example indicates `IEAVNPFF ... Loadwait/Restart` as the last entry.

```
*** NIP Statistics ***

IEAVNIP0  00:00:00.024  NIP Base
IEAVNIPM  00:00:00.077  Invoke NIP RIMs
IEAVNPE6  00:00:03.358  Service Processor Interface
IEAVNPFF  00:00:00.023  Loadwait/Restart
```

2. If the report is complete, the last entries indicate the master scheduler initialization is complete with the total times as in the following example:

   ```
   FINSHMSI  00:00:00.001  Wait for attached CMDs

   IEEMB860  00:05:17.024      Uncaptured time:  00:00:00.810

   Total Time:  00:07:13.473
   ```

   Enter the IPCS **SELECT ALL** command to verify which system address spaces are active and have completed initialization. For example:

   ```
   ASID JOBNAME  ASCBADDR  SELECTION CRITERIA
   ---- -------- --------  ------------------
   0001 *MASTER* 00FD3400  ALL
   0002 PCAUTH   00F8DE80  ALL
   0003 RASP     00F8DD00  ALL
   0004 TRACE    00F8DB80  ALL
   0005 DUMPSRV  00F8DA00  ALL
   0006 XCFAS    00F81E80  ALL
   0007 GRS      00F81D00  ALL
   0008 SMSPDSE  00F80400  ALL
   0009 CONSOLE  00F80280  ALL
   000A WLM      00F4F300  ALL
   000B ANTMAIN  00F4F180  ALL
   000C ANTAS000 00F4F000  ALL
   000D OMVS     00F4DE80  ALL
   000F IEFSCHAS 00FC6E80  ALL
   0010 JESXCF   00F8B500  ALL
   0011 ALLOCAS  00F8B380  ALL
   0012 IOSAS    00F97280  ALL
   0013 IXGLOGR  00F97100  ALL
   0014 JES2     00FC0D80  ALL
   ```

3. Do a search using symptoms that include the last:
   - Initialization routine to run
   - Message that was issued (to the log or screen)
   - Address space to initialize

Table 19. Common wait states that occur during IPL.

Where possible, this table contains an example of the wait state code, reason code, explanation, example, and where to find more information.

Wait state code	Reason code	Explanation	Find information in:
X'064'	X'005'	Indicates an ABEND was issued during NIP. To diagnose using a SADUMP, enter the **SYSTRACE ALL** command, go to the bottom of the output and enter the **FIND *SVC PREV** command to locate the ABEND issued.	Chapter 10, “Diagnosing an abend,” on page 155

Example: WAIT X'064' RSNX'005'
`00 0001 00000000  *SVCE    D 040C1000 814E1EE2  00000010 84000000 84878000  10000201 00000000 0001 0001 BEC5A7BA2DAC255B`
`00800004                  00400000`			

X'064'	X'009'	Indicates a program check occurred during NIP. To diagnose using a SADUMP, format the LCCA by entering the **CBFORMAT** LCCA*x* command (where *x* is CP ID).	Chapter 10, “Diagnosing an abend,” on page 155

Table 19. Common wait states that occur during IPL (continued).

Where possible, this table contains an example of the wait state code, reason code, explanation, example, and where to find more information.

Wait state code	Reason code	Explanation	Find information in:

Example: WAITX'064' RSNX'009'

```
LCCA: 00F81000
+0000  LCCA..... LCCA      CPUA..... 0041      CAFM..... 4000      PGR1..... 00000000  00000000  00000000  00000000  00000000
+001C            00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000
+0048  PGR2..... 00000000  08256D70  07BCBCF8  07DFC098  00FCC990  00000903  07BB6678  07E0717C  00000005  00000005  00000000
+0074            07EB2B5F  012B2AE0  08256F90  08256000  08257148  PPSW..... 070C1000  812B2B58            PINT..... 00040011
+0094  PVAD..... 08257000  CR0...... 00000000  PGR3..... 00000000  00000000  00000000  00000000  00000000  00000000  00000000
+00BC            00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  P2A0..... 010D061C
+00E4  P2A1..... 00000002  P2A2..... 00000000  P2A3..... 00000000  P2A4..... 00000000  P2A5..... 00000000  P2A6..... 00000000
+00FC  P2A7..... 00000000  P2A8..... 00000000  P2A9..... 00000000  P2AA..... 00000000  P2AB..... 00000000  P2AC..... 00000000
+0114  P2AD..... 00000000  P2AE..... 00000000  P2AF..... 7F281180
+0120  RSGR..... 00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000
+014C            00000000  00000000  00000000  00000000  00000000  DSA2..... 00000000  P2C0..... 5FB1EE40  P2C1..... 0331B07F
+016C  P2C2..... 037B9E00  PX2K..... 8040      PX2S..... 000A      PX2A..... 0001      PX2P..... 000A      P2C5..... 04EC6280
+017C  P2C6..... FE000000  P2C7..... 0331B07F  P2C8..... 00000000  P2C9..... 00000000  P2CA..... 00000000  P2CB..... 00000000
+0194  P2CC..... 02FCA403  P2CD..... 0331B07F  P2CE..... DF884EC5  P2CF..... 7F70A0B8  PSW3..... 00000000  00000000
```

Wait state code	Reason code	Explanation	Find information in:
X'040'	Not applicable	Indicates an ABEND was issued during NIP. Gather additional information by entering the IPCS **STATUS CPU**:	From this example, a search argument including symptoms WAIT040 ABEND878 RC10 IEAVNPB2 would be built to check for a known problem.

Example:

```
             CPU (X'00') STATUS:
PSW=00020000 80000000 00000000 00878040
    (Running in PRIMARY, key 0, AMODE 31, DAT OFF)
    Disabled for PER I/O MCH
NIP RIM IEAVNPB2 has failed
ABEND=878 REASON=00000010

  Register values
      0-3  84000000  84878000  03A5EF90  0000E676
      4-7  00FCC498  00FCDC98  0393F428  00FCC000
      8-11 0187C518  00001000  00000000  00000030
     12-15 00000001  00000000  FE000424  00000010
  ASCB1 at FCC000, JOB(*MASTER*), for the home ASID
  ASXB1 at FCC370 and a local SRB for the home ASID
  HOME ASID: 0001 PRIMARY ASID: 0001 SECONDARY ASID: 0001
```

Diagnosing an enabled wait state

When the IPCS **STATUS CPU** command does not show a *no work wait* or a *coded disabled wait*, start diagnosis by checking for resource contention using the following steps.

Steps for diagnosing an enabled wait state

1. Verify that IPCS **STATUS CPU** report shows every CPU in a no work wait:

```
CPU(X'00') STATUS:
 PSW=07060000 00000000 00000000 00000000
     No work wait
   CPU is in no work wait
```

2. Enter the IPCS **ANALYZE EXCEPTION** command to look for resource contention.

```
  CONTENTION EXCEPTION REPORT

JOBNAME=PMIMTAPE  ASID=0065  TCB=007DD0F8

JOBNAME=PMIMTAPE HOLDS THE FOLLOWING RESOURCE(S):

   RESOURCE #0003:  There are 0025 units of work waiting for this resource
      NAME=MAJOR=SYSIEFSD MINOR=Q4 SCOPE=SYSTEM

 STATUS FOR THIS UNIT OF WORK:
   This address space is on the SRM IN queue.
JOBNAME=PCICBDTS  ASID=0266  TCB=008723A0

JOBNAME=PCICBDTS HOLDS THE FOLLOWING RESOURCE(S):

   RESOURCE #0002:  There are 0022 units of work waiting for this resource
      NAME=MAJOR=SYSIEFSD MINOR=Q10 SCOPE=SYSTEM
STATUS FOR THIS UNIT OF WORK:
   This address space is on the SRM IN queue.

****************************************************** END OF DATA *****
```

If resource contention exists, use the IPCS **FIND** command on the TCB or SSRB address that is identified as the holder of a resource in the analyze exception report to see if the TCB is waiting for any other resources. If found waiting, get the TCB or SSRB address of the holder of that resource and repeat the process until the bottom of the contention chain is reached.

3. Enter an IPCS SUMMARY FORMAT JOBNAME(*xyz*) for the holder of the resource in contention.
4. Use the IPCS **FIND** command to locate the TCB or SSRB that is identified as the holder of the resource. If found waiting, get the TCB/SSRB address of the holder of that resource and repeat the process, until the bottom of the contention chain is reached.
   a. If the holder of the resource is a TCB, go to "Examining the TCB status" on page 198.
   b. If the holder is an SSRB, either find the SSRB in the SUMMARY FORMAT output or format the SSRB control blocks with the IPCS **CBFORMAT srb address STR(SRB)** command. Using the PSW address from the CPSW field, use the IPCS **WHERE** command or browse storage to find the module name that determines where the SRB was last running.

      For example:

```
SSRB: 02451200
    +0000  ID....... SSRB      FLNK..... 02452900  ASCB..... 00F72D00  CPAF..... 0000      PASI..... 006C      PTCB..... 007E2250
    +0014  EPA...... 00000000  RMTR..... 8142F3D8  PARM..... 00000000  WEB...... 01DF0598  PKF...... 00        FLGS..... 08
    +0026  HLHI..... 00        FLGS..... 00        FRRA..... 00000000
    +0030  FPRS..... 00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000            TRAN..... 00000000
    +0054  SAFN..... 0000      TYPE..... 0C
    +0058                      GPR0..... 00000000  GPR1..... 00000000  GPR2..... 00000041  GPR3..... 01DDE4C8  GPR4..... 00000000
    +006C  GPR5..... 00000000  GPR6..... 00000000  GPR7..... 00000000  GPR8..... 00000000  GPR9..... 00000000  GPRA..... 00000000
    +0084  GPRB..... 014228AF  GPRC..... 814218B0  GPRD..... 7F01E388  GPRE..... 0186926E  GPRF..... 00000000

    +0098  CPSW..... 070C0000  81421ED0            CPUT..... 00FFFFFF  FF4AF580            TIME..... 00000000  009C49C0
    +00B0  XSB...... 02451838  ORMT..... 18FEF6C8  LSA1..... 0206D998  LAA...... 000A0068  LSDP..... 0206DA18  ALOV..... 00000000


    Register values
      0-3  00000000  00000000  00000000  00000000
      4-7  00000000  00000000  00000000  00000000
      8-11 00000000  00000000  00000000  00000000
     12-15 00000000  00000000  00000000  00000000

 +0108  DUCT..... 00000000  00000000  00000000  00000000  0007BF00  00000000  00000000  00000000  00000000  00000000  00000000
    +0134            00000000  00000000  00000000  00000000  00000000
    +0634                      SSD...... 19413000  OPAS..... C240      OPTC..... 00000000  SUPF..... 80000069  SUSP..... 00
    +064A  SUSP..... 00000000  00000000  00000000  00000000            SUSP..... 00000000  SUSP..... 00000000  SYNC..... 00000000
    +0668  SYNC..... 00000000  SYNC..... 00000000  SYNC..... 00000000  SYNC..... 00000000  SYNC..... 00000000  SUSP..... 00000000
    +0680            00000000  SUSP..... 00000000  SUSP..... 00000000  AFPR..... 00000000  00000000  00000000  00000000  00000000
    +06AC            00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000
    +06D8            00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  FPFL..... 00
    +0740  TRNE..... 00000000  00000000
```

5. Use the module name to search for a known problem.
6. If the search reveals no contention, use the IPCS **SYSTRACE ALL** command to examine the system trace table for the ASIDs that are executing.
7. Use the **FIND WAIT** command in the SYSTRACE report to check for any WAIT type system trace entries.

8. If no WAIT entries exist, there might be contention on CPU resources. Scroll through SYSTRACE noting the ASIDs associated with the entries. If all the entries are for a couple of ASIDs and they are mostly CLKC or EXT type entries, it might indicate a loop. Go to Chapter 13, “Diagnosing a loop,” on page 203.
9. If WAIT entries are found, then there is no contention on any CPU resource. Talk to the operator to get more specific information on what appears to be hung from an operations perspective. If a specific job or class of jobs are hung (like Batch or TSO users), then get the job name or the specific TSO userid and go to Chapter 12, “Diagnosing a job or subsystem hang,” on page 193.

Diagnosing a coded disabled wait state

When the IPCS **STATUS CPU** command shows a *disabled coded wait state*, use the following steps to start diagnosis.

Steps for diagnosing a coded disabled wait state

1. Obtain the disabled wait state code from the from the stand-alone dump, if obtained or the Hardware Management Console (HMC). The IPCS **STATUS WORKSHEET** report from a stand-alone dump often contains a corresponding wait state message. For example:

```
                            MVS Diagnostic Worksheet

Dump Title: WAIT 0A2 REASON 15A ZOS 1.7 INSTALAC SYSPLEX

CPU Model 2066 Version 00 Serial no. 0123B4 Address 00
Date: 05/22/2006     Time: 20:48:49.817141 Local

Wait State Message Issued at 20:43:48 on Day 142 of 2006:
 IXC436W THIS SYSTEM HAS LOST TIME SYNCHRONIZATION WITH THE OTHER
         SYSTEMS IN THE SYSPLEX AND
         HAS BEEN PLACED INTO A NON-RESTARTABLE
         WAIT STATE CODE: 0A2 REASON CODE: 15A


 SYSTEM RELATED DATA

 CVT  SNAME (154) IEASYSFI   VERID (-18)

      CUCB   (64) 00FD5140   PVTP  (164) 00FFB108    GDA  (230) 0210A278

      RTMCT (23C) 00F4FB20   ASMVT (2C0) 00FD75D8    RCEP (490) 01907F18

 CSD  Available CPU mask: 8000  Alive CPU mask: 80000000  00000000
      Number of active CPUs: 00000001


 PROCESSOR RELATED DATA

  NAME             OFFSET  |  CPU 00
 --------------------------+----------------------------------------
  PSW at time of dump      | 00020000
                           | 80000000
                           | 00000000
                           | 0015A0A2
```

Figure 42. IPCS STATUS WORKSHEET report from a stand-alone dump

2. Find the wait state code using *z/OS MVS System Codes*. If there is no recommended action:
   a. Look up the wait state code in the wait state code to module table in *z/OS MVS System Codes*.
   b. Use the module name to identify the component using the module identification table in *z/OS MVS Diagnosis: Reference*.
3. Perform a search using the wait state code, reason code, module name and component identifier to look for a known problem. If you cannot find a match, report the problem to IBM.

Table 20. Common disabled wait states. This table contains examples where possible for illustrative purposes.

Wait state code	Reason code	Explanation	More information:

X'01B'		This is a restartable wait state that is loaded when an active SLIP trap requests an action of WAIT. The conditions specified on the **SLIP** command are met. The system enters a wait state, as requested. Information about the environment when the slip trap matches is presented in message IEE844W. For example:
		`Message IEE844W:`
		
		`IEE844W SLIP TRAP 0001 MATCHED. ACTION=WAIT TYPE=PER.`
		` PER INFO:`
		
		`207F: 010FEED2`
		`PSW: 440cc000 810FEED6`
		`CR 3-4 80000001 00010001`
		`GR 0-3 00000000 01DA9600  00000041 01F993B8`
		`4-7: 01EA9600 00000000 010FEE80 00F8C000`
		`8-B: 00FD6F80 010FDA8 00FCD680 9598947F`
		`C-F: 01F998E0 014E7040 01EA9140 0000000C`
		`AR 0-3: 00000000 00000000 00000000 00000000`
		`4-7: 00000000 00000000 00000000 00000000`
		`8-B: 00000000 00000000 00000000 00000000`
		`C-F: 00000000 00000000 00000000 00000000`
		
		`RESTART THE SYSTEM TO CONTINUE`
			The topic on SLIP problem data in the SLIP work area in *z/OS MVS Diagnosis: Tools and Service Aids*.

X'040'		The system ended a task during nucleus initialization program (NIP) processing.	Chapter 10, "Diagnosing an abend," on page 155
X'064'	X'005'	Indicates that an ABEND was issued during NIP. To diagnose using a SADUMP, do a SYSTRACE ALL, max PF8 to the bottom of the output and do a 'F *SVC PREV' to locate the ABEND issued.	

EXAMPLE: WAITX'064' RSNX'005'
`00 0001 00000000  *SVCE    D 040C1000 814E1EE2  00000010 84000000 84878000  10000201 00000000 0001 0001 BEC5A7BA2DAC255B`
`                                    00800004                    00400000`			

X'064'	X'009'	A program check occurred during nucleus initialization program (NIP). Message IEA304W explains the wait state and entry code. If the message does not appear on the console, you can find the message in the wait state message area (WSMA). The WSMA is described in *z/OS MVS Data Areas* in the z/OS Internet library (http://www.ibm.com/systems/z/os/zos/bkserv/).To diagnose using a stand-alone dump, format the LCCA by entering the CBFormat LCCA*x* command (where *x* is the CPU ID).	Chapter 10, "Diagnosing an abend," on page 155
X'0A2'	X'004'	The operator entered the **`VARY XCF,sysname,OFFLINE`** command to remove the system from the sysplex.	Ask the operator or system programmer why the system was varied out of the sysplex before continuing with diagnosis.
	X'104'	I/O is prevented because a system is fenced.	Go to "Diagnosing a system partitioned from a sysplex because of status update missing" on page 188
	X'10C'	Cross-system coupling facility (XCF) or cross-system extended services (XES) encountered an unrecoverable error and stopped the system. The system also issues this wait state in response to an operator request to stop the system. For information about diagnosing sysplex problems see, *z/OS MVS Diagnosis: Reference*.	"Diagnosing an enabled wait state" on page 184

Diagnosing a system partitioned from a sysplex because of status update missing

A system that is partitioned from the sysplex because of status update missing is really indicating a system hang or wait. A system in a sysplex indicates its health by updating a timestamp value on the *SYSPLEX* couple dataset every second. If the timestamp is not updated for the failure detection interval as defined in the COUPLE*xx* parmlib member in use (85 seconds is the recommended value), the system will be partitioned from the sysplex. Use the following steps to guide your diagnosis:

Steps for diagnosing a system partitioned because of status update missing

When partitioned, the system is be put into a X'0A2' wait state with one of the following reason codes:

- X'104' if a System Failure Management (SFM) Policy is active and IO is fenced.
- X'10C' if the status update missing.

1. Verify the system was partitioned from the sysplex as indicated by message IXC101I, which is in the OPERLOG:

   ```
   IXC101I SYSPLEX PARTITIONING IN PROGRESS FOR SYS22 REQUESTED BY
   XCFAS. REASON: SFM STARTED DUE TO STATUS UPDATE MISSING
   ```
2. Verify the system is in a X'0A2' wait state by entering the `IPCS STATUS CPU` command:

   ```
   CPU(X'01') STATUS:
   PSW=000A0000 800200A2
       Disabled wait state code 00A2  SUPPLMNT INFO 8002
   Wait occurred because system monitor control information cannot be read or written.
     ASCB6 at F42700, JOB(XCFAS), for the home ASID
     ASXB6 at 5FDE88 and TCB6G at 5FF500 for the home ASID
     HOME ASID: 0006 PRIMARY ASID: 0006 SECONDARY ASID: 0006
   ```
3. Enter the IPCS **COUPLE SYSPLEX EXCEPTION** to identify the reason the system is being partitioned.
4. If the reason is that the system entered a coded disabled wait state prior to sysplex partitioning, go to “Diagnosing a coded disabled wait state” on page 186 otherwise, go to “Diagnosing an enabled wait state” on page 184.

Searching the problem reporting databases

Search arguments are used to search problem reporting databases. If the problem you are diagnosing was already reported and the symptoms are in the database, the search produces a match. Searching is an iterative process; you might need to use the procedures in “Gathering additional data for hangs and waits” on page 190 to gather additional data and continue your search.

Steps for searching the problem reporting databases

About this task

Use the following steps to search the problem reporting databases and determine if the problem was previously reported:

Procedure

1. Develop a free-format search argument using the symptoms obtained from the analysis performed. The free-format search argument can include any of the following symptoms:

- WAITxxx RSNyyyyyyyy (where xxx is the disabled wait state code and yyyyyyyy is the associated reason code)
- Module or CSECT name
- Resource name that was found to be in contention
- Message ID
- Component ID

2. If an argument is not provided, use the primary symptom string in VERBEXIT SYMPTOM output, if available, or use the following symptoms:
   - Program identifier: PIDS/ccccccccc
   - CSECT name or module name: RIDS/cccccccc
   - Wait state:
     – If a disabled wait, with a wait state code: WS/D0hhh
     – If an enabled wait: WS/E0000
   - If ANALYZE EXCEPTION output indicates a lockout: PCSS/LOCKOUT
   - Input request (call, command, macro, statement), if one is associated with the problem: PCSS/cccccccccc
   - Symptoms created from information in the STATUS CPU output
3. Select the problem type on the search tool panel, based on **STATUS CPU** output:

Table 21. Selecting the problem type for jSTATUS CPU output

Problem Type	STATUS CPU Output
Disabled wait	DISABLED WAIT STATE CODE
Enabled wait	NO WORK WAIT
Enabled wait	DISABLED FOR ccc (not I/O or EXT)
Hang	None of the above

4. If the search finds that the problem was previously reported, request the problem fix.

   Searching is an iterative process. If the search finds no match, you can remove some symptoms or change the symptoms and search again. Continue searching for matches by adding, removing, and changing symptoms using the steps in “Gathering additional data for hangs and waits” on page 190.
5. If you still cannot find the cause of the hang or wait or if the problem is new, report the problem to IBM using the procedure in Chapter 23, “Reporting problems to IBM,” on page 287. Record the following problem data in Chapter 24, “Problem diagnostic worksheet,” on page 291:
   - Problem type: disabled wait, enabled wait, or hang
   - Search argument
   - Dump formatted by IPCS online
   - SMF records, if obtained
   - Accompanying messages: identifiers and texts
   - Hard-copy log, beginning 15 to 30 minutes before the problem, or master trace, if not wrapped between the problem and dump
   - Logrec records, beginning 15 to 30 minutes before the problem and edited using the SPOTCHK and TIMESEQ parameters
   - All output data sets related to the problem
   - Data on any related problems

- Module name and level
- Name and level of the operating system(s) with a list of program temporary fixes (PTF) applied at the time of the problem and all installation modifications, exits, and products with other than Class A service
- Other problem data developed while using the procedures in this document or other diagnosis books for the component, subsystem, or program

Results

You know you are done when you find a match for your problem or report the problem.

Related information:

- See "Searching problem reporting databases" on page 8 for more information on developing search arguments.
- See *z/OS MVS IPCS Commands* for the IPCS subcommands:
  - ANALYZE
  - STATUS
  - VERBEXIT SYMPTOM
  - STATUS CPU

Gathering additional data for hangs and waits

Gathering additional data will increase your chances of finding a match in the problem reporting databases. Use the procedures outlined in this section to gather additional data and continue searching the problem reporting databases.

Steps for gathering messages and logrec for hangs

About this task

Use the following steps to collect and analyze messages and logrec records about the problem.

Procedure

1. Use time stamps to select messages and software, symptom, and hardware logrec records related to the problem. Look in the following:
   - OPERLOG or SYSLOG
   - VERBEXIT MTRACE dump output, which shows the buffer for system messages
   - VERBEXIT LOGDATA dump output, which formats the logrec buffer
   - Logrec data set, formatted by EREP
2. Use the **COPYCAPD** command to check for any SVC dumps captured in dataspaces that did not have a chance to get written out to a dataset prior to the system hang. For example:

```
Number  Time stamp           Title
------  -------------------  -----------------------------------------------------------------------
--------------
 1  04/10/2006 15:41:18  END OF MEMORY RESOURCE MANAGER HANG DETECTED: TCB = 009BBE88, NAME = ................
 2  04/10/2006 15:42:24  END OF MEMORY RESOURCE MANAGER HANG DETECTED: TCB = 009D64B0, NAME = ................
 3  04/10/2006 15:44:43  IXC431I XCF STALLED GROUP
 4  04/10/2006 15:46:35  JES2/XCF Env on current & other sysplex members via IEADMCJ2
 5  04/10/2006 15:51:35  END OF MEMORY RESOURCE MANAGER HANG DETECTED: TCB = 009683F0, NAME = ................
 6  04/10/2006 15:52:43  END OF MEMORY RESOURCE MANAGER HANG DETECTED: TCB = 009D17A8, NAME = ................
 7  04/10/2006 15:53:51  END OF MEMORY RESOURCE MANAGER HANG DETECTED: TCB = 009A4998, NAME = ................
 8  04/10/2006 15:55:00  END OF MEMORY RESOURCE MANAGER HANG DETECTED: TCB = 009A4758, NAME = ................
 9  04/10/2006 15:56:21  END OF MEMORY RESOURCE MANAGER HANG DETECTED: TCB = 00985CF8, NAME = ................
```

These SVC dumps can be extracted into dump data sets using the **COPYCAPD** command and then diagnosed individually. Go to Chapter 10, "Diagnosing an abend," on page 155.

Results

Related information:

- See *z/OS MVS IPCS Commands* for the IPCS subcommands.
- For formatting of logrec records, see Recording logrec error records in *z/OS MVS Diagnosis: Tools and Service Aids*.
- See *z/OS MVS Diagnosis: Reference* for logrec reports.
- For explanations of the messages, see:
  - *z/OS MVS System Messages, Vol 1 (ABA-AOM)*
  - *z/OS MVS System Messages, Vol 2 (ARC-ASA)*
  - *z/OS MVS System Messages, Vol 3 (ASB-BPX)*
  - *z/OS MVS System Messages, Vol 4 (CBD-DMO)*
  - *z/OS MVS System Messages, Vol 5 (EDG-GFS)*
  - *z/OS MVS System Messages, Vol 6 (GOS-IEA)*
  - *z/OS MVS System Messages, Vol 7 (IEB-IEE)*
  - *z/OS MVS System Messages, Vol 8 (IEF-IGD)*
  - *z/OS MVS System Messages, Vol 9 (IGF-IWM)*
  - *z/OS MVS System Messages, Vol 10 (IXC-IZP)*
  - *z/OS MVS Dump Output Messages*
  - The message book for a subsystem or program

You know you are done when your search produces a match.

Chapter 12. Diagnosing a job or subsystem hang

Overview of a hang or wait

When a job or subsystem hang occurs, you might notice part of the system is not functioning or that a job is in the system for a long time without processing.

Symptom of a job hang:

- A job remains in the system for a long time and does not end.

Symptom of a subsystem hang:

- A subsystem does not respond to any commands. For example, in a JES2 system, enter a **$DI1** command and JES2 does not respond.

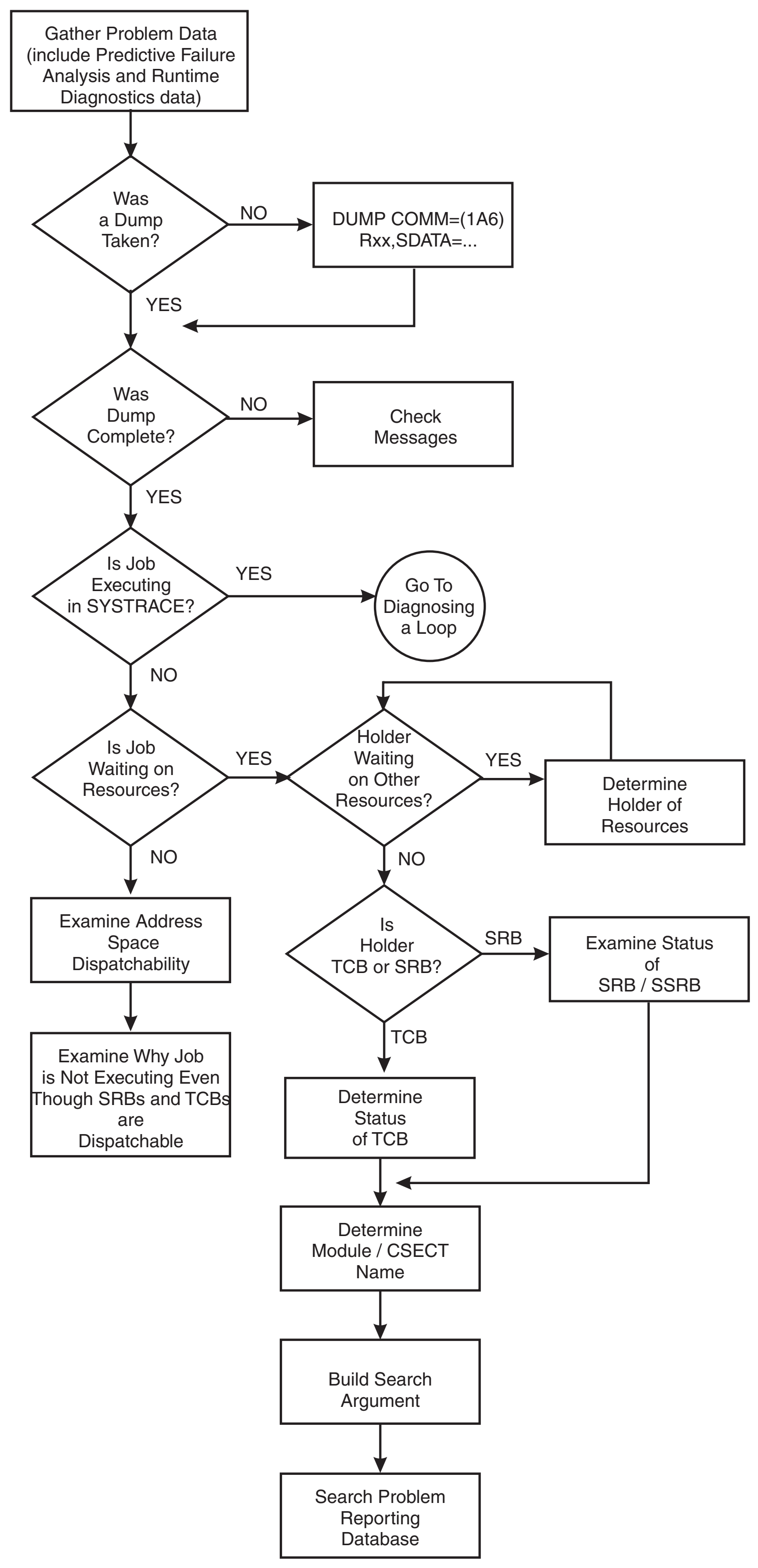
Steps for diagnosing a job or subsystem hang

The following procedures will guide you through diagnosing a job or subsystem hang:

1. "Gathering additional data for a job or subsystem hang" on page 195
2. "Determining the status of a hung job or subsystem" on page 195
3. "Determining if a job is waiting for resources" on page 196
4. "Determining address space dispatchability" on page 196
5. "Examining the SRB status" on page 197
6. "Examining the TCB status" on page 198
7. "Examining why a job is not running" on page 200

Use the following flowchart to guide diagnosis of a job or subsystem hang:

Job or subsystem hang analysis



Gather Problem Data (include Predictive Failure Analysis and Runtime Diagnostics data)
Was a Dump Taken?
NO
DUMP COMM=(1A6) Rxx,SDATA=...
YES
Was Dump Complete?
NO
Check Messages
YES
Is Job Executing in SYSTRACE?
YES
Go To Diagnosing a Loop
NO
Is Job Waiting on Resources?
YES
Holder Waiting on Other Resources?
YES
Determine Holder of Resources
NO
NO
Examine Address Space Dispatchability
Is Holder TCB or SRB?
SRB
Examine Status of SRB / SSRB
Examine Why Job is Not Executing Even Though SRBs and TCBs are Dispatchable
TCB
Determine Status of TCB
Determine Module / CSECT Name
Build Search Argument
Search Problem Reporting Database

Figure 43. Flowchart for job or subsystem hang analysis

Gathering additional data for a job or subsystem hang

About this task

It is important gather information about what the job or subsystem was doing at the time of the hang. This includes answering the following questions:

- What recovery procedures did you attempt? For example, was the **MODIFY**, **CANCEL** or **FORCE** command entered?
- What error messages were received at the time of the hang?
- What commands were entered and responses received?
- Were there environmental changes? For example, a new device installed or a software maintenance upgrade.

Record this information in Chapter 24, “Problem diagnostic worksheet,” on page 291.

Step for gathering additional data

Before you begin: You must know how to use IPCS and understand how to take an SVC dump. For complete information on SVC dump, see using the IEADMCxx parmlib member in the topic on SVC dump in*z/OS MVS Diagnosis: Tools and Service Aids*.

Request an SVC dump in one of the following ways:

- Use the **DUMP** command for the hung job and any other related ASIDs.
- If this is a subsystem or system address space, use IEADMCxx, the **DUMP** command parmlib member. IEADMCxx allows you to specify the collection of dump data without having to remember and identify all the systems, address spaces and data spaces involved. IEADMCxx from SYS1.SAMPLIB defines the dump options for specific jobs, functions, subsystem, as documented in *z/OS MVS Initialization and Tuning Reference*.
- Specify the following:

  ```
  DUMP COMM=(dumptitle)
  Rxx,ASID=(1,xx),SDATA=(GRSQ,SQA,CSA,RGN,TRT,COUPLE,XESDATA,NUC),END
  ```

  Where *xx* is the ASID associated with the hung job. You can also use JOBNAME=(xyz)).

  If using a IEADMCxx parmlib member, enter:

  ```
  DUMP COMM=(title),PARMLIB=xx
  ```

You must also determine the level of the z/OS operating system:

- Use the IPCS **CBFORMAT CVT** command. For an example, see “Steps for diagnosing a system hang” on page 178.

Determining the status of a hung job or subsystem

Sometimes a loop can appear to be a hang. Use the following steps to identify if the job is running:

Steps for determining the status of a hung job or subsystem

About this task

1. In the SVC dump, look for entries in the system trace table by issuing IPCS **SYSTRACE JOBNAME(*name*)**. If there are no entries, continue with “Steps for determining if a job is waiting for resources” on page 196.

2. If there are entries, summarize to determine if:
   - All the entries are in SRB mode or all the entries are for one or two TCB's.
   - The PSW addresses are all within a certain range and repeating. (If the PSW addresses in CLKC and EXT type trace entries are all within a specific range, go to Chapter 13, "Diagnosing a loop," on page 203.)

   If both are true, use the IPCS **WHERE** command to identify which module that the PSW address is pointing to.
3. Use the module name and the symptom *LOOP* to build a search argument to check for a known problem. For details, see "Extracting problem symptoms and search arguments" on page 9.

Determining if a job is waiting for resources

Use the following steps to identify if the job or subsystem is waiting for resources:

Steps for determining if a job is waiting for resources

About this task

1. Enter the IPCS **ANALYZE RESOURCE** command.
2. Look in the report for the job name that is hung to determine if it is waiting for system resources. If it is waiting for resource, note the resource name and examine the status of the unit of work holding the resource. **For example:**

```
JOBNAME=SDSF      ASID=003D  SSRB=02FC4900  --> unit of work holding resource
 JOBNAME=SDSF     HOLDS THE FOLLOWING RESOURCE(S):

    RESOURCE #0002:  There are 0018 units of work waiting for this resource
       NAME=LOCAL LOCK FOR ASID 003D
       DATA=INTERRUPTED AND NOW DISPATCHABLE
```

3. Do a **FIND** on the TCB or SSRB address of the holder to check if it is waiting on another resource. Repeat until the bottom of the contention chain is reached.
4. Choose from the following:
   a. If the unit of work that is holding the resource is not dispatchable, go to "Examining the SRB status" on page 197 or "Examining the TCB status" on page 198 to determine why.
   b. If the unit of work that is holding the resource is dispatchable, go to "Steps for examining why a job is not running" on page 200.
   c. If the job is not holding or waiting for a resource, go to "Steps for examining address space dispatchability."

Determining address space dispatchability

If you are working with a complete SVC dump, as indicated by messages IEE911E or IEA611I, examine address space dispatchability using the following steps.

Steps for examining address space dispatchability

Procedure

1. Enter IPCS **SUMMARY FORMAT JOBNAME(xyz)** to obtain a report for the hung job.
2. Locate the ASCB (**F** ASCB). **For example:**

```
ASCB: 00F2A280
   +0000  ASCB..... ASCB      FWDP..... 00F2A100  BWDP..... 00F2A400  LTCS..... 00000000  SVRB..... 009FD598  SYNC..... 000001DE
   +0018  IOSP..... 00000000  R01C..... 0000      WQID..... 0000      SAWQ..... 04CBAAF0  ASID..... 003D      R026..... 0000
   +0028  LL5...... 00        HLHI..... 01        DPH...... 00EF      TCBE..... 00000000  LDA...... 7FF18EA0  RSMF..... C0
   +0035  FLG3..... 00        R036..... 0000      CSCB..... 18E89D00  TSB...... 00000000  EJST..... 00000000  22CF88A0
   +0048  EWST..... BEC51B1A  4660A320            JSTL..... 000141DE  ECB...... 809FDC80  UBET..... BE89F19D  TLCH..... 00000000
   +0060  DUMP..... 009FFE88  AFFN..... FFFF      RCTF..... 01        FLG1..... 00        TMCH..... 00000000  ASXB..... 009FDE88
   +0070  SWCT..... 00A7      DSP1..... 80        FLG2..... 00        RSV...... 0000      SRBS..... 000E      LLWQ..... 00000000
   +007C  RCTP..... 009FE0A8  LOCK..... 4FFFFFFF  LSWQ..... 04CBAAF1  QECB..... 00000000  MECB..... 40000000  OUCB..... 03292100
   +0094  OUXB..... 0384E6F8  FMCT..... 0000      LEVL..... 03        FL2A..... 80        R09C..... 00000000  IQEA..... 00000000
   +00A4  RTMC..... 00000000  MCC...... 00000000  JBNI..... 00000000  JBNS..... 00FAF60C  SRQ1..... 00        SRQ2..... 00
   +00B6  SRQ3..... 00        SRQ4..... 00        VGTT..... 00000000  PCTT..... 00000000  SSRB..... 0000      SMCT..... 00
   +00C3  SRBM..... 07        SWTL..... 00000D30  SRBT..... 00000001  301DBF80            LTCB..... 03748628  LTCN..... 00000005
   +00D8  TCBS..... 00000000  LSQT..... 00000000  WPRB..... 009FE950  NDP...... EF        TNDP..... FF        NTSG..... FF
   +00E7  IODP..... FD        LOCI..... 00000000  CMLW..... 044AA6B8  CMLC..... 00000000  SSO1..... 000000    SSO4..... 00
   +00F8  ASTE..... 03F05F40  LTOV..... 7FF95000  ATOV..... 7FFF0128  ETC...... 0001      ETCN..... 0001      LXR...... 0001
   +010A  AXR...... 0000      STKH..... 009FE960  GQEL..... 19021900  LQEL..... 19021778  GSYN..... 00000000  XTCB..... 009F1598
   +0120  CS1...... C0        CS2...... 00        R122..... 0000      GXL...... 03105000  EATT..... 00000000  034077E0
   +0130  INTS..... BE89F19D  91A9EA80            LL1...... 00        LL2...... 00        LL3...... 00        LL4...... 00
   +013C  RCMS..... 00000000  IOSC..... 00000067  PKML..... 0000      XCNT..... 01F4      NSQA..... 00000000  ASM...... 03896540
   +0150  ASSB..... 03AA4280  TCME..... 00000000  GQIR..... 00000000  R15C..... 00000000  00000000  00000000  CREQ..... 00000000
   +016C  RSME..... 03807290  AVM1..... 00        AVM2..... 00        AGEN..... 0000      ARC...... 00000000  RSMA..... 03807128
   +017C  DCTI..... 0000062E
```

3. Check the following fields in the ASCB control block:

   DSP1 (+X'72')
   Address space non-dispatchability bits:
   - If **X'80'**, an SVC dump is in progress, which does not indicate a problem.
   - If X'40', an address space is failing. Check field MCC (ASCB+X'A8') for the memory termination completion code (MCC=88**40D**000). If so, this job is terminating with an ABEND40D. Extract the associated reason code from field ARC (ASCB+X'174').
   - If X'10' or X'32', the address space is logically or physically swapped out. To determine the reason for the swap, issue the IPCS **VERBX SRMDATA** and **FIND** the job name (**F** jobname).

   LOCK (+X'80')
   Address space local lock word:
   - If the word contains X'7FFFFFFF', a TCB or SSRB is suspended holding the local lock. To locate the PSW and registers for the lock holder, enter **FIND** IHSA. Use the IPCS **WHERE** command on the address from CPSW (current PSW) in the IHSA to identify the lock holder. Use this module name in a search.
   - If the word contains X'0000004X', this identifies that the lock holder is currently executing on CP *4x* at the time of the dump. Do an IPCS **SYSTRACE CPU(4x)**, go to the bottom of the output and scroll back examining the trace entries to see what unit of work is executing. For information on interpreting system trace, see system trace in *z/OS MVS Diagnosis: Tools and Service Aids*.
   - If the word contains X'4FFFFFFF' or X'FFFFFFFF' (the SRB or TCB ready to run id), then the unit of work is on the dispatching queue and waiting to get dispatched. Go to "Steps for examining why a job is not running" on page 200.

Examining the SRB status

Steps for examining the SRB status

1. Enter the IPCS **CBFORMAT** *srb* address STR(SRB) to format the SRB or SSRB and related control blocks that describe the environment of the system request block (SRB). **For example:**

```
SSRB: 02451200
   +0000  ID....... SSRB      FLNK..... 02452900  ASCB..... 00F72D00  CPAF..... 0000      PASI..... 006C      PTCB..... 007E2250
   +0014  EPA...... 00000000  RMTR..... 8142F3D8  PARM..... 00000000  WEB...... 01DF0598  PKF...... 00        FLGS..... 08
   +0026  HLHI..... 00        FLGS..... 00        FRRA..... 00000000
   +0030  FPRS..... 00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000            TRAN..... 00000000
   +0054  SAFN..... 0000      TYPE..... 0C
   +0058                      GPR0..... 00000000  GPR1..... 00000000  GPR2..... 00000041  GPR3..... 01DDE4C8  GPR4..... 00000000
   +006C  GPR5..... 00000000  GPR6..... 00000000  GPR7..... 00000000  GPR8..... 00000000  GPR9..... 00000000  GPRA..... 00000000
   +0084  GPRB..... 014228AF  GPRC..... 814218B0  GPRD..... 7F01E388  GPRE..... 0186926E  GPRF..... 00000000

   +0098  CPSW..... 070C0000  81421ED0            CPUT..... 00FFFFFF  FF4AF580            TIME..... 00000000  009C49C0
   +00B0  XSB...... 02451838  ORMT..... 18FEF6C8  LSA1..... 0206D998  LAA...... 000A0068  LSDP..... 0206DA18  ALOV..... 00000000


   Register values
     0-3  00000000  00000000  00000000  00000000
     4-7  00000000  00000000  00000000  00000000
     8-11 00000000  00000000  00000000  00000000
    12-15 00000000  00000000  00000000  00000000

   +0108  DUCT..... 00000000  00000000  00000000  00000000  0007BF00  00000000  00000000  00000000  00000000  00000000  00000000
   +0134            00000000  00000000  00000000  00000000  00000000
   +0634                      SSD...... 19413000  OPAS..... C240      OPTC..... 00000000  SUPF..... 80000069  SUSP..... 00
   +064A  SUSP..... 00000000  00000000  00000000  00000000            SUSP..... 00000000  SUSP..... 00000000  SYNC..... 00000000
   +0668  SYNC..... 00000000  SYNC..... 00000000  SYNC..... 00000000  SYNC..... 00000000  SYNC..... 00000000  SUSP..... 00000000
   +0680            00000000  SUSP..... 00000000  SUSP..... 00000000  AFPR..... 00000000  00000000  00000000  00000000  00000000
   +06AC            00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000
   +06D8            00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  FPFL..... 00
   +0740  TRNE..... 00000000  00000000
```

2. Use the PSW address from the CPSW field in the IPCS **WHERE** command or browse storage to look for a module name to determine where the SRB was last executing.
3. Use the module name and hang to search for a known problem.

Examining the TCB status

Steps for examining the TCB status

1. Enter the IPCS **SUMMARY FORMAT JOBNAME(xyz)** to generate a report of address space related control blocks representing the job, which includes the ASCB, SRB's, TCB's, RB's, and others.
2. Locate the TCB.
   a. If you have identified a TCB as holding or waiting for a resource, enter the **FIND TCB:** *xxxxxxxx* where *xxxxxxxx* is the virtual address of the TCB.
   b. If you have not identified a TCB, scroll to the bottom of the **SUMMARY FORMAT** report to locate the last TCB on the chain.
3. Examine the TCB non-dispatchability bits, ignoring the SDUMP non-dispatchability indicator:

```
TCB: 008FF2A0
     +0000  RBP...... 008FF218  PIE...... 00000000  DEB...... 00000000  TIO...... 008D1FE8  CMP...... 00000000  TRN...... 40000000
     +0018  MSS...... 7FFFE700  PKF...... 80        FLGS..... 00000004  01                  LMP...... FF        DSP...... FF
     +0024  LLS...... 00000000  JLB...... 008FCDD8  JPQ...... 008FD000
  ......
     Register values
       0-3  00000001  25400F88  00000040  00000000
       4-7  008DFAB0  0000000A  008D1FE0  25401048
       8-11 008FC138  008FF560  00000000  A5400D0A
      12-15 830048B2  00006F60  00000311  808FF218
   +012C  EAE...... 7FF4E8F8  ARC...... 00000000  GRES..... 00000000  STCB..... 7F6F8A68  TTIME.... 00000000  007A1440
   +0144  CELAP.... 00006FF0  R148..... 0000      RBYT1.... 00        LEVEL.... 03        BDT...... 00000000  NDAXP.... 00000000
   +0154  SENV..... 00000000
   Task non-dispatchability flags from TCBFLGS4:
    Top RB is in a wait                        ---> important
   Task non-dispatchability flags from TCBFLGS5:
    Secondary non-dispatchability indicator
   Task non-dispatchability flags from TCBNDSP2:
    SVC Dump is executing for another task      ----> ignore, always on in an svcdump
```

4. If RB is in a WAIT or with no flags on:

a. Do a **FIND ACTIVE** to locate the RB control blocks (these can be PRBs, SVRBs, IRBs or SIRBs).

b. Look at the last RB formatted. Verify that the wait or suspend count in the high order byte of the RBLINK is greater than zero. Extract the PSW address from the OPSW field. **For example:**

```
ACTIVE RBS

PRB: 008FF218
-0020  XSB...... 7FFFDCA0  FLAGS2... 80     RTPSW1... 00000000  00000000         RTPSW2... 00000000  00000000
-0008  FLAGS1... 40800003  WLIC..... 00020001

+0000  RSV...... 00000000  00000000       SZSTAB... 00110082  CDE...... 008FD000  OPSW..... 070C0000  A5400F00
+0018  SQE...... 00000000  LINK..... 018FF2A0
+0020  GPR0-3... FD000008  00006FF8  00000040  008DFAD4
+0030  GPR4-7... 008DFAB0  008FD0C8  008D1FE0  FD000000
+0040  GPR8-11.. 008FC138  008FF560  00000000  008FD0C8
+0050  GPR12-15. 830048B2  00006F60  008FC184  008FC168
           64-Bit GPRs from the RB/XSB


  Left halves of all registers contain zeros
  0-3  FD000008  00006FF8  00000040  008DFAD4
  4-7  008DFAB0  008FD0C8  008D1FE0  FD000000
  8-11 008FC138  008FF560  00000000  008FD0C8
  12-15 830048B2  00006F60  008FC184  008FC168
  +0060  RSV...... E2C8D9F0  F4D3D740
```

c. Enter the IPCS **WHERE** *xxxxxxxx* , using the address from the RBOPSW field with the high order addressing mode bit off, to identify which module or CSECT was last in control. **For example:**

```
Command ===> ip w 25400f00
 ********************************************************* TOP OF DATA *********
      ASID(X'0020') 25400F00. SHR04LP+01F8 IN EXTENDED PRIVATE
      ASID(X'0020') 25400F00. AREA(Subpool251Key08)+0F00 IN EXTENDED PRIVATE
 ********************************************************* END OF DATA *********
```

- If PSW points into ISGGWAIT, the TCB is waiting on an ENQ resource. Check the storage pointed to from GPR1 saved in this SVRB. It will point to storage containing the ENQ parmlist. The ENQ parmlist +4 points to the major name and +8 points to the minor name associated with the resource. For a mapping of the parmlist, see SVC 38 in *z/OS MVS Diagnosis: Reference*.
- If the PSW points to IEAVEWAT, a program call (PC) entered the WAIT as requested. To determine who the requester was, locate the current linkage stack entry for this TCB by entering **F LSE: PREV**. Enter the IPCS **WHERE** command on the PSW address from field PSWE to determine who performed the WAIT (PC 30D). Use this module name when doing a search for a known problem. **For example:**

```
LSE: 7F58B140
    GENERAL PURPOSE REGISTER VALUES
    00-01.... 00000000  00000001  00000000  80095238
    02-03.... 00000000  00FCB818  00000000  018944B8
    04-05.... 00000000  7FF6ADB0  00000000  03ECC378
    06-07.... 00000000  00FCB818  00000000  98F27718
    08-09.... 00000000  18F29088  00000000  00FEC410
    10-11.... 00000000  03ECC0D0  00000000  18F24747
    12-13.... 00000000  18F23748  00000000  7F4FF100
    14-15.... 00000000  0000030D  00000000  18F27712
    ACCESS REGISTER VALUES
    00-03.... 00000000  00000000  00000000  00000000
    04-07.... 00000000  00000000  00000000  00000000
    08-11.... 00000000  00000000  00000000  00000000
    12-15.... 00000000  00000000  00000000  00000000
    PKM...... 8040      SASN..... 0006      EAX...... 0000      PASN..... 0006
    PSW...... 07041000  80000000            PSWE..... 00000000  18F27752--->  this PSW address indicates
    TARG..... 00000000  0000030D            MSTA..... 00000000  00000000 who did a PC 30D WAIT
    TYPE..... 0D
     PC STATE ENTRY
    RFS...... 0D80      NES...... 0000
```

5. If the *abnormal wait* non-dispatchability flag is on, check for a subtask that is also abending. Do a **FIND** on TCB examining the CMP field in the TCB for one that is non-zero. Next, repeat step 3 examining the TCB non-dispatchability bits.
6. If any other TCB non-dispatchability flags are on, determine the name to the TCB non-dispatchability flag set by examining the value of TCBFLGS field, bytes 4 and 5, and field TCBNDSP. Interpret bit settings using the mapping of the TCB in the version of *z/OS MVS Data Areas* that corresponds to the release of z/OS you are running in your environment. See the z/OS Internet library (http://www.ibm.com/systems/z/os/zos/bkserv/). Use the TCB non-dispatchability bit name in a search for a known problem.
7. Build a symptom string including:
   - The word WAIT
   - The module name
   - The component ID identified
   - Any other symptoms. For example, the resource waited on (SYSZTIOT or CMSEQDQ lock) or the TCB non-dispatchability bit name (TCBSTP).

   Search for a known problem.

Examining why a job is not running

When a job is not running, but the TCBs or SRBs are dispatchable, determine what is preventing the job from being dispatched. This often occurs when higher priority work is monopolizing the CPs. Use the following steps to examine the activity in the system trace.

Steps for examining why a job is not running

1. Enter the IPCS **SYSTRACE** ***ALL*** command.
2. Enter **FIND WAIT** in the SYSTRACE report and identify WAIT type trace entries.
   a. If you find WAIT entries, there are intervals when the dispatcher did not find any dispatchable work to dispatch. Review the previous sections of Chapter 12, “Diagnosing a job or subsystem hang,” on page 193 to ensure nothing overlooked.
   b. If you do not find any WAIT entries, examine which ASIDs are running by scrolling through the IPCS **SYSTRACE ALL** report.
3. In a IPCS **VERBX SRMDATA** report, compare the service classes and periods of the jobs executing in SYSTRACE to those that are hung.

   For example:

```
JOB   ZFS
ASID  01F5
OUCB  02700780 IN      QUEUE
                +11 (NSW)  NONSWAPPABLE
                +11 (PVL)  PRIVILEGED PROGRAM
                (ASCBRSME) RAX ADDRESS IS 020848C8
                           SERVICE CLASS = SYSSTC
                           WORKLOAD = SYSTEM
                           INTERNAL CLASS= $SRMGOOD
                           PERIOD = 01
```

If these are of higher priority, examine where they are running by using the IPCS **WHERE** on several of the PSW addresses from the SYSTRACE report.

4. Search for a known problem using the module, CSECT, or both.

Chapter 13. Diagnosing a loop

Note: Use Runtime Diagnostics to diagnose and possibly solve the loop condition before continuing these steps. See "CPU analysis" on page 40 in Chapter 4, "Runtime Diagnostics," on page 35.

Overview of a loop

A loop is a repetitive set of instructions being performed by a job or unit of work. A job or function that is looping can appear to be hung or can use a high amount of CP resource and lock out other work from getting service.

There are three types of loops and symptoms for each type:

Disabled loop
A disabled loop is repetitive execution, usually in system level code, with the IO and EXT type interrupts prevented with a PSW mask of X'x4' in the high order byte of the PSW. A disabled loop is bound to one CP in the system. If in a multi-processor environment and resources are held, a spin loop is detected. If on a uniprocessor, a disabled loop will result in a system outage.

Enabled loop
An enabled loop occurs under a unit of work (TCB or SRB) that is executing on behalf of a job or function. It is executing with a PSW that is enabled for I/O and external interrupts with a mask of X'x7' in the high order byte. A unit of work that is looping enabled, is interrupted periodically for IO, EXT or CLKC type interrupts, which are traced in the system trace table.

Spin loop
A spin loop is a timed disabled loop in system code controlled by the installation with specifications in the EXSPATxx (excessive spin condition actions) parmlib member. The system can spin or loop disabled waiting for a resource, such as a lock, to be released by another CP in a multi-processing environment. See *z/OS MVS Initialization and Tuning Reference*for more information on the EXSPATxx parmlib member.

Disabled loop symptoms

Disabled loops are easier to identify than enabled loops. Symptoms include:

- System CP usage increases for unexplained reasons.
- There is no communication with the system through the master and alternate consoles.
- Console communications are locked out. To check for communication with the console, enter **DISPLAY T** command and the system will not respond.

Enabled loop symptoms

Enabled loops allow some or all interrupts. The loops are usually caused by an error in an application program. All or most of the loop is in code running in problem state, but the loop can include system code if any instructions in the loop request system services. An enabled loop can run on more than one central processor. The loop will uselessly consume resources and might take over all system operation.

Additional symptoms include:
- A bottleneck, indicating that the system slows down periodically, thus creating a performance problem.
- A job stays in the system for a long time without changing status or ending.
- Low priority work slows down or stops running (a result of a higher priority enabled loop).
- System CP usage increases for unexplained reasons or CP usage of an address space is much higher than normal.

Spin loop symptoms

A spin loop occurs when one processor in a multiprocessor environment is unable to communicate with another processor or requires a resource currently held by another processor. The processor that has attempted communication is the *detecting* or *spinning* processor. The processor that has failed to respond is the *failing* processor.

The detecting processor continuously attempts its communication with the failing processor until either:
- It is successful.
- A specified time interval has passed.

When the communication is not successful within this interval, an excessive spin loop *time out* exists. The detecting processor then initiates recovery processing for the condition.

MVS processing for excessive spin-loop conditions can provide recovery without any operator prompts or actions required. The following recovery actions can be defaulted to or specified in the EXSPATxx parmlib member:

SPIN Continue spinning for another interval to allow the event to complete

ABEND
End the current unit of work on the failing processor but allow the recovery routines to retry

TERM End the current unit of work on the failing processor and do not allow the recovery routines to retry

ACR Invoke alternate CP recovery (ACR) to take the failing processor offline.

- The system chooses the appropriate action without requiring any decision or action. If an action taken in response to an occurrence of an excessive spin loop does not resolve the condition, the system takes the next action when the next excessive spin loop time out occurs. The default order in which the system takes the actions is SPIN, ABEND, TERM, and ACR.
- An installation can change the order of the actions, except the first one, that the system takes.
- For hardware-related errors that formerly caused message IEA490A, the system immediately initiates ACR processing without working through the sequence of actions and without requiring any intervention.

- There is a default spin loop time-out interval. You can change this interval through the combination of a parameter in EXSPATxx parmlib member and entering the SET command.
- To avoid unnecessary recovery actions, system functions that can validly exceed the interval are exempt from excessive spin-loop processing. If they exceed the time out interval, the system functions do cause an excessive spin loop record to be written to the log recording data set (LOGREC) data set.
- The installation can still control excessive spin loop recovery through operator actions.

See EXSPATxx (excessive spin condition actions) in *z/OS MVS Initialization and Tuning Reference*.

Steps for diagnosing a loop

The following steps guide you through diagnosing a loop:

1. “Gathering additional data for a loop” on page 206
2. “Analyzing the dump to determine the type of loop” on page 207
3. “Diagnosing a disabled loop” on page 208
4. “Diagnosing an enabled loop” on page 209
5. “Diagnosing an excessive spin (spin loop)” on page 212
6. “Analyzing a logrec error record” on page 214
7. “Searching the problem reporting databases” on page 215

Use the following flowchart to guide diagnosis of a loop:

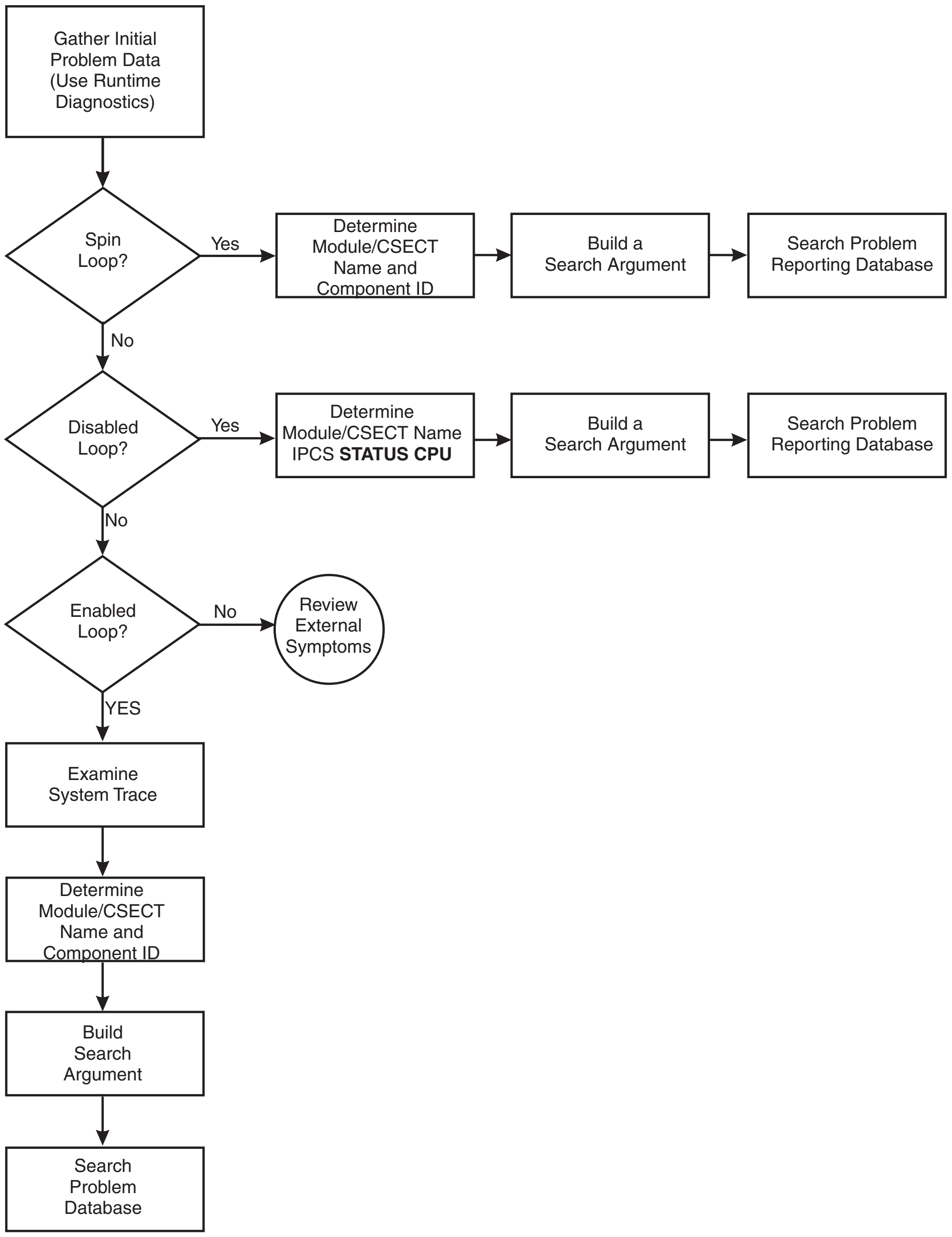

Gather Initial Problem Data (Use Runtime Diagnostics)
Spin Loop?
Yes
Determine Module/CSECT Name and Component ID
Build a Search Argument
Search Problem Reporting Database
No
Disabled Loop?
Yes
Determine Module/CSECT Name IPCS STATUS CPU
Build a Search Argument
Search Problem Reporting Database
No
Enabled Loop?
No
Review External Symptoms
YES
Examine System Trace
Determine Module/CSECT Name and Component ID
Build Search Argument
Search Problem Database

Figure 44. Flowchart for diagnosis of a loop

Gathering additional data for a loop

By gathering the correct data you can determine what recovery actions are necessary.

Steps for gathering loop data

Gather the following types of data:

1. The description of the external symptoms, including any software or hardware changes.
2. What recovery actions were attempted? Was a **MODIFY**, **CANCEL** or **FORCE** command entered?

3. Request an SVC dump using the system **DUMP** command for the job or jobs involved in the loop. If this is a subsystem or system address space, define the dump options using the applicable dump parmlib member, IEADMCxx documented in *z/OS MVS Initialization and Tuning Reference*.

   Otherwise, specify the following:

   ```
   DUMP COMM=(dumptitle) Rxx,ASID=(1,xx),SDATA=(GRSQ,SQA,CSA,RGN,TRT,COUPLE,XESDATA,NUC),END
   ```

   Where **xx** is the ASID associated with the looping job. You can also specify it with JOBNAME=(xyz). If an SVC dump is not possible or the system is hung, request a stand-alone dump.
4. If the **DUMP** command cannot be entered because the system is hung, or does not complete, request a stand-alone dump of the system. See the topic on stand-alone dump in *z/OS MVS Diagnosis: Tools and Service Aids*.
5. If this is an enabled loop and there is a SYSMDUMP DD coded in the JCL, you can enter a **CANCEL** command for the looping job with the DUMP option.

Related information

- See *z/OS MVS Diagnosis: Tools and Service Aids* for information about stand-alone dumps, SVC, SYSMDUMP, and stand-alone dumps.
- See *z/OS MVS JCL Reference* for the SYSMDUMP DD statement.
- See *z/OS MVS System Commands* for information about the **DUMP** and **CANCEL** commands.

Analyzing the dump to determine the type of loop

You can determine what type of loop you have by analyzing the dump.

Step for analyzing the dump for loop type

Before you begin: You need access to IPCS.

Analyze any dump for the type of loop.

Format the dump with an IPCS **STATUS CPU** subcommand. Under the heading CPU(X'nn') STATUS, look for the following:

```
DISABLED FOR cccccccc
```

- System processing was disabled for one or more types of interrupts for the module running at the time of the dump. The system can be disabled for program event recording (PER), I/O, external interrupts (EXT), and machine checks (MCH).

The type is:

- **Disabled loop**: If the system was disabled for I/O or EXT or both.
- **Enabled loop**: If the system was not disabled for I/O or EXT.

In this example, the statement DISABLED FOR PER indicates an ENABLED loop.

```
Example: STATUS CPU Subcommand Output
CPU STATUS:
Warnings regarding STRUCTURE(ASCB) at ASID(X'0001') 00FD5F00:
Located via STRUCTURE(ASVT) at ASID(X'0001') 00F336D0
   Storage not in dump

 PSW=070C2000  8AC2CB2A  (RUNNING IN PRIMARY, KEY 0,  AMODE 31, DAT ON)
```

```
    DISABLED FOR PER
  ASID(X'0001') 0AC2CB2A. AREA(PRIVATEX)+02CB2A IN EXTENDED PRIVATE
 ASCB76 at F52080, JOB(CATALOG), for the home ASID
```

Related information:

See *z/OS MVS IPCS Commands* for the STATUS subcommand.

Diagnosing a disabled loop

A disabled loop is not visible in the system trace output because disabled routines do not take interrupts. Normally, a disabled loop results in a spin loop in a multiprocessor environment. When analyzing a stand-alone dump for a disabled loop, use the stored status data to determine the module involved in the loop. Disabled loops often result in a system outage if they persist. Therefore, you usually be working with a stand-alone dump.

Steps for diagnosing a disabled loop

1. In an IPCS session, enter the IPCS **STATUS CPU** command to examine status of each CP. **For example:**

```
CPU(X'02') STATUS:
PSW=04042000 80000000 00000000 011E8592
(Running in PRIMARY, key 0, AMODE 31, DAT ON)
Disabled for PER I/O EXT
NOCPU ASID(X'0001') 011E8592. IEANUC01.IEAVELKX+073A IN READ ONLY NUCLEUS
ASCB1 at FD3400, JOB(*MASTER*), for the home ASID
ASXB1 at FD3598 and a local SRB for the home ASID
HOME ASID: 0001 PRIMARY ASID: 0001 SECONDARY ASID: 0001


CLTE: 01F76020
+0000 BLSD..... 00000000 XDS...... 00000000 XRES..... 00000000 XQ....... 00FD3018 ESET..... 00FD3028 IXSC..... 00000000
+0018 IXSH..... 00FD3040 IXDS..... 00000000 IXLL..... 00000000 ULUT..... 00FD3030 IXRE..... 00000000 WLMR..... 00000000
+0030 WLMQ..... 00FD3050 REGS..... 00000000 CNTX..... 00FD3060 SSD...... 00000000
HOLDING LOCK(S): CPU
CURRENT FRR STACK IS: NORMAL
CPU(X'04') STATUS:
PSW=04040000 80000000 00000000 017D5126
(Running in PRIMARY, key 0, AMODE 31, DAT ON)
Disabled for PER I/O EXT
NOCPU ASID(X'0001') 017D5126. IEANUC01.IAXRC+034E IN READ ONLY NUCLEUS
ASCB1 at FD3400, JOB(*MASTER*), for the home ASID
ASXB1 at FD3598 for the home ASID. No block is dispatched
HOME ASID: 0001 PRIMARY ASID: 0003 SECONDARY ASID: 0367

CLTE: 01B19000
+0000 BLSD..... 00000000 XDS...... 00000000 XRES..... 00000000 XQ....... 00FD3018 ESET..... 00FD3028 IXSC..... 00000000
+0018 IXSH..... 00FD3040 IXDS..... 00000000 IXLL..... 00000000 ULUT..... 00FD3030 IXRE..... 00000000 WLMR..... 00000000
+0030 WLMQ..... 00FD3050 REGS..... 00000000 CNTX..... 00FD3060 SSD...... 00000000
HOLDING LOCK(S): CPU RSM RSMCM RSMST RSMAD
CURRENT FRR STACK IS: NORMAL
```

   In the preceding example, the PSW for CP 2 indicates that it is executing disabled in CSECT **IEAVELKX**, the lock manager. The status for CP 4 indicates several locks are held, therefore, this is a normal spin in **IEAVELKX**. The loop that is causing the problem is in IAXRC.

2. Use the name of the CSECT executing on CP 4, IARXC and the symptom *LOOP* to check for a known problem.
3. From the module prefix, identify the component, subsystem, or product, if provided by IBM.

   Use the module name to query the SMP/E zone for a module entry with that module name. If the search does not find a match, the module is not an IBM module. If the search indicates a match, use the FMID to positively identify the product.
4. Continue diagnosis as follows:

a. If all the addresses are in components of z/OS, continue with “Searching the problem reporting databases” on page 215.
b. If all the addresses are in an IBM subsystem or product, continue diagnosis with the diagnosis publication for the subsystem or product. See Chapter 22, “Diagnosis information for z/OS base elements and features,” on page 285 for the correct publication.
c. If all the addresses are in components of z/OS and in an IBM subsystem or product, continue with “Searching the problem reporting databases” on page 215 and with the diagnosis book for the subsystem or product. See Chapter 22, “Diagnosis information for z/OS base elements and features,” on page 285 for the correct publication.
d. If any of the addresses are in an installation-provided program, including an installation exit routine, continue diagnosis with that program, using the dump.

   If some addresses are in the program or routine and some in system modules, the loop is probably in the program or routine and includes one or more requests for system services.

Diagnosing an enabled loop

Enabled loops are often quite large and can include several distinct operations, such as I/O, SVCs, and module linkages. Because the loop is enabled, it is interrupted, preempted, and resumed many times. This makes the loop pattern difficult to recognize. “Steps for diagnosing an enabled loop” can help make that identification easier.

For an enabled loop you should have either an SVC dump or a stand-alone dump.

Steps for diagnosing an enabled loop

Verify the system is in an enabled loop by examining the activity in the system trace table.

1. Enter the IPCS **SYSTRACE ALL TIME(LOCAL)** command. The following is an example of the system trace table output:

```
--------------------------------------------------- SYSTEM TRACE TABLE ---------------------------------------------------
------------------
---- ------------------ --
PR ASID WU-ADDR- IDENT CD/D PSW----- ADDRESS- UNIQUE-1 UNIQUE-2 UNIQUE-3 PSACLHS- PSALOCAL PASD SASD TIMESTAMP-LOCAL
UNIQUE-4 UNIQUE-5 UNIQUE-6 PSACLHSE DATE-05/08/2006
05-004C 00000000 EXT 1005 077C6000 951AFBD0 00001005 00000000 00000000 0301 004C 11:00:24.184339 00000000
05-004C 00000000 EXT 1005 077C6000 951AFBB2 00001005 00000000 00000000 0301 004C 11:00:24.184681 0000000
05-004C 031F6478 SSRB 077C6000 951AFBB2 00000000 04979298 00 00000000 0301 004C 11:00:24.184683 00000000
05-004C 00000000 EXT 1005 077C6000 951AFBA6 00001005 00000000 00000000 0301 004C 11:00:24.185024 00000000
05-004C 00000000 EXT 1005 077C6000 951AFBD8 00001005 00000000 00000000 0301 004C 11:00:24.185366 00000000
05-004C 00000000 EXT 1005 077C6000 951AFBD0 00001005 00000000 00000000 0301 004C 11:00:24.185708 00000000
05-004C 031F6478 SSRB 077C6000 951AFBD0 00000000 04979298 00 00000000 0301 004C 11:00:24.185710 00000000
05-004C 00000000 EXT 1005 077C6000 951AFBC0 00001005 00000000 00000000 0301 004C 11:00:24.186051 00000000
05-004C 00000000 EXT 1005 077C6000 951AFBD0 00001005 00000000 00000000 0301 004C 11:00:24.186393 00000000
05-004C 00000000 EXT 1005 077C6000 951AFBAC 00001005 00000000 00000000 0301 004C 11:00:24.186734 00000000
05-004C 031F6478 SSRB 077C6000 951AFBAC 00000000 04979298 00 00000000 0301 004C 11:00:24.186737 00000000
05-004C 00000000 EXT 1005 077C6000 951AFBD0 00001005 00000000 00000000 0301 004C 11:00:24.187082 00000000
05-004C 00000000 EXT 1005 077C6000 951AFBAC 00001005 00000000 00000000 0301 004C 11:00:24.187423 00000000
05-004C 00000000 EXT 1005 077C6000 951AFBB2 00001005 00000000 00000000 0301 004C 11:00:24.187765 00000000
05-004C 031F6478 SSRB 077C6000 951AFBB2 00000000 04979298 00 00000000 0301 004C 11:00:24.187767 00000000
05-004C 00000000 EXT 1005 077C6000 951AFBD4 00001005 00000000 00000000 0301 004C 11:00:24.188108
```

Figure 45. System trace table entry

   The columns represent:
   - The PR, ASID, and TCB-ADDR columns tell you who is executing
   - The IDENT CD/D columns tell you the type of event
   - The PSW column tells you where
   - The UNIQUE columns tell you how

- The PASD tells you which ASID is being addressed
- The TIMESTAMP tells you when.

2. To diagnose an enabled loop, look for patterns of repetitive entries in system trace. Typically, the bounds of an enabled loop are identified by the PSW addresses in EXT, CLKC, and IO events. In the preceding example, the bounds of the loop appear to go from PSW address 151AFBA6 to 151AFBD0.
3. Enter the IPCS **WHERE** command on these PSW addresses to determine which module or CSECT the loop is in or browse storage at that location. **For example,** under IPCS Option 1:

```
Command ==>

CURRENT DEFAULTS:
Source ==> DSNAME('H44IPCS.PMR59356.B019.DUMP')
Address space ==> ASID(X'0001')

OVERRIDE DEFAULTS:
Source ==> DSNAME('H44IPCS.PMR59356.B019.DUMP')
Address space ==> ASID(X'301')

Password ==>
POINTER:
Address ==> 151AFBA6
Remark ==> ASID(X'0301')
```

   Browse backward (PF 7) looking for an module or CSECT identifier.

```
Command ===>
151AEEA0 1606B1F8 00000000 00000000 00000000 | ...8............ |
151AEEB0 00000000 04000005 00000000 00000000 | ................ |
151AEEC0.:151AEECF. LENGTH(X'10')--All bytes contain X'00'
151AEED0 00000000 80000000 00000000 00000000 | ................ |
151AEEE0 00000000 00000000 03002100 30004001 | .............. . |
151AEEF0.:151AEF0F. LENGTH(X'20')--All bytes contain X'00'
151AEF10 40000000 00000000 00000000 00000000 | ............... |
151AEF20.:151AEFFF. LENGTH(X'E0')--All bytes contain X'00'
151AF000 90ECDDB4 A7C50046 151B0688 C3D8D4D6 | ....xE.....hCQMO |
151AF010 D9C5C4C3 40404040 4040F0F3 61F0F261 | REDC 03/02/ |
151AF020 F0F6F1F9 4BF4F740 40404040 40404040 | 0619.47 |
151AF030 40404040 4040D6E2 40404040 F2F2F040 | OS 220 |
151AF040 D7D2F2F0 F6F0F040 F5F6F9F7 60C9F0F3 | PK20600 5697-I03 |
151AF050 404D835D 40D99683 9285A340 E29686A3 | ............... |
151AF060 A6819985 6B40C995 834B40F1 F9F9F96B | ............... |
151AF070 40F2F0F0 F640C193 9340D989 8788A3A2 | ............... |
151AF080 40D985A2 8599A585 844B4000 00000000 | ............... |
```

   Browse backward (PF 7) looking for an module or CSECT identifier.
4. Obtain the PSW addresses from the system trace entries involved in the loop. For ANALYZE output and SUMMARY KEYFIELD CURRENT output, use the TCB address.

   If using a PSW address, ignore the leftmost bit of the leftmost digit. The leftmost bit of the leftmost digit denotes addressing mode and is not part of the address.
5. Do one of the following actions, for each address in the loop:
   - If analyzing the dump interactively, use the address in a WHERE subcommand to obtain the name of the load module.
   - If analyzing printed output, find the address:
     – In dump output from the LIST or VERBEXIT SUMDUMP subcommand. Look for the CSECT name eye-catcher. IBM module names are generally

followed by an assembly date and a product identifier or PTF level, such as HBB7720 or UY01234; most eye-catchers are at the beginning of a module, but some are at the end.

– In a module listed for the LPAMAP or VERBEXIT NUCMAP subcommand. LPAMAP will list load modules. Use AMBLIST to obtain the offsets of CSECTS within those load modules. NUCMAP lists CSECTs with offsets, but can only be used for modules within the nucleus.

In the following example, **STATUS CPU** output and **WHERE** subcommand, the PSW identifies the address as X'13206AA'.

```
CPU STATUS:

 PSW=040C2000  813206AA  (RUNNING IN PRIMARY, KEY 0,  AMODE 31, DAT ON)
     DISABLED FOR PER I/O EXT
   ASID(X'0006') 013206AA. IEANUC01.IXLM2SP+07AA IN READ ONLY NUCLEUS
  ASCB1 at FD1780, JOB(*MASTER*), for the home ASID
  ASXB1 at FD1A30 for the home ASID. No block is dispatched
  HOME ASID: 0001 PRIMARY ASID: 0006 SECONDARY ASID: 0006
```

Using the WHERE subcommand, the load module name is IXLM2SP plus an offset of 7AA.

```
ASID(X'0006') 013206AA. IEANUC01.IXLM2SP+07AA IN READ ONLY NUCLEUS
```

6. From the module prefix, identify the component, subsystem, or product, if provided by IBM.

   Use the module name to query the SMP/E zone for a module entry with that module name. If the search does not find a match, the module is not an IBM module. If the search indicates a match, use the FMID to positively identify the product.

7. Continue diagnosis as follows:
   - If all the addresses are in components of z/OS, continue with “Searching the problem reporting databases” on page 215.
   - If all the addresses are in an IBM subsystem or product, continue diagnosis with the diagnosis book for the subsystem or product.
   - If all the addresses are in components of z/OS and in an IBM subsystem or product, continue with “Searching the problem reporting databases” on page 215 and with the diagnosis book for the subsystem or product.
   - If any of the addresses are in an installation-provided program, including an installation exit routine, continue diagnosis with that program, using the dump.

     If some addresses are in the program or routine and some in system modules, the loop is probably in the program or routine and includes one or more requests for system services.

Related information:

- See the topics on IBM component, subsystem, product program identifier or module prefix in *z/OS MVS Diagnosis: Reference*.
- See *z/OS MVS IPCS Commands* for the IPCS subcommands.
- See *z/OS MVS Diagnosis: Tools and Service Aids*
- See *z/OS MVS Diagnosis: Reference* for the SVC summary.
- For the format and contents of the EED and TCB, see *z/OS MVS Data Areas* in the z/OS Internet library (http://www.ibm.com/systems/z/os/zos/bkserv/).

Diagnosing an excessive spin (spin loop)

To avoid unnecessary recovery actions, system functions that can validly exceed the interval are exempt from excessive spin-loop processing, so that they will not cause any recovery actions. If they exceed the time-out interval, these system functions do cause an excessive spin loop record to be written to the logrec data set.

You can examine the in-storage logrec buffer for entries that recovery routines have made, but which were not written to the logrec data set because of a system problem. Very often it is these records that are the key to the problem solution. See the topic on obtaining information from the logrec in *z/OS MVS Diagnosis: Tools and Service Aids*.

When the system writes a dump, the dump includes the records in the logrec buffer in storage; the buffer records have been either written to the logrec data set or are queued to be written to the logrec data set. When you begin to diagnose a dump for a system problem, you can use IPCS to view the system records in the logrec recording control buffer.

Before you begin You need to have IPCS installed and the following information:

- An EREP report of SYS1.LOGREC
- Any SVC dumps, stand-alone dumps showing the ABEND071, or SYSMDUMP dump
- SYSLOG and OPERLOG. Look for spin loop messages like IEE331A or IEE178I. Note that IEE331A is rare; IEE178I is more likely to occur. For example:

```
00 IEE331A PROCESSOR (0) IS IN AN EXCESSIVE
DISABLED SPIN LOOP WAITING FOR CPU IN
STOPPED STATE
REPLY U OR SPIN TO CONTINUE SPIN
REPLY ABEND TO TERMINATE WORK ON
PROCESSOR (1) WITH RETRY,
REPLY TERM TO TERMINATE WORK ON
PROCESSOR (1) WITHOUT RETRY,
OR STOP PROCESSOR (1) AND
REPLY ACR
(AFTER STOPPING THE PROCESSOR, DO NOT START IT)
```

```
IEE178I AUTOMATIC RECOVERY IS IN PROGRESS NO OPERATOR ACTION IS REQUIRED
PROCESSOR (00) DETECTED AN EXCESSIVE DISABLED SPIN LOOP
WAITING FOR LOCK RELEASE FROM PROCESSOR (03).
AUTOMATIC RECOVERY ACTION IS SPIN
```

Steps for diagnosing an excessive spin

Use the following steps:

1. In an EREP report of SYS1.LOGREC or an IPCS VERBX LOGDATA report in a dump, enter **FIND** S0071 (X'071'). You might find multiple records for S0071 with different reason codes.
2. The following example is for an ABEND071 RSN10, which is the first record written to logrec for a spin condition:

   Example: VERBEXIT LOGDATA Output

```
  SEARCH ARGUMENT ABSTRACT

  PIDS/5752SC1CM RIDS/IEANUC01#L RIDS/IEAVTEXS AB/S0071 PRCS/00000010
  REGS/0E68E REGS/0C7AA RIDS/IEAVTEXS#R

  SYMPTOM            DESCRIPTION
```

```
-------             -----------
PIDS/5752SC1CM      PROGRAM ID: 5752SC1CM
RIDS/IEANUC01#L     LOAD MODULE NAME: IEANUC01
RIDS/IEAVTEXS       CSECT NAME: IEAVTEXS
AB/S0071            SYSTEM ABEND CODE: 0071
PRCS/00000010       ABEND REASON CODE: 00000010
REGS/0E68E          REGISTER/PSW DIFFERENCE FOR R0E: 68E
REGS/0C7AA          REGISTER/PSW DIFFERENCE FOR R0C: 7AA
RIDS/IEAVTEXS#R     RECOVERY ROUTINE CSECT NAME: IEAVTEXS
```

Look for a SOFTWARE RECORD for abend X'071' with the following reason codes:

- Reason code X'10': Recovery action of SPIN was taken, and a logrec entry was written to provide information about the CSECT/module that was detected as looping. No attempt was made to abend the unit of work.
- Reason code X'20': Recovery action of ABEND was taken, targeting the looping program with a retryable ABEND071.
- Reason code X'30': A recovery action of TERM was taken, targeting the looping program with a non-retryable ABEND071.

In a logrec record for an abend X'071', look in the SEARCH ARGUMENT ABSTRACT for the symptom RIDS/IEAVTEXS. If found, look under the heading VARIABLE RECORDING AREA (SDWAVRA) for the following problem data:

- EX SPIN RECORD in the EBCDIC text
- An array of up to 16 pointers: addresses of the functional recovery routines (FRR) on the interrupted FRR stack
- A binary number: index into the FRR stack of the current FRR
- Array of processors causing the spin from SVTSPCP
- Spin loop timeout interval
- Spin loop recovery actions
- Control registers

In the following output, the SDWAVRA indicates an excessive spin record and contains an array of pointers to the FRR stack.

Example: VERBEXIT LOGDATA Output –– Variable Recording Area

```
 VARIABLE RECORDING AREA (SDWAVRA)

+000    KEY: 39     LENGTH: 0E
+002    C5E740E2    D7C9D540    D9C5C3D6    D9C4        |EX SPIN RECORD  |

+010    KEY: 37     LENGTH: 04
+012    C6D9D9E2                                        |FRRS            |

+016    KEY: 38     LENGTH: 40
+018    811F43F8    811D74ED    81319571    81086D05    |A..8A...A.N.A._.|
+028    811D74ED    00000000    00000000    00000000    |A...............|
+038    00000000    00000000    00000000    00000000    |................|
+048    00000000    00000000    00000000    00000000    |................|
```

a. Enter the IPCS **WHERE** command on the PSW address at the time of error from the logrec entries found for an S0071. This should identify the CSECT/module that is excessively looping.
b. Search for a known problem using the symptoms: ABEND071 and the module or CSECT name identified. If IPCS **WHERE** fails to identify a CSECT/module name, browse storage preceding the psw address looking for a module eye catcher using IPCS Option 1.

3. Identify the modules containing each instruction in the loop, as follows:For **automatic spin loop** recovery, if the dump is an **SVC dump**: Use a STATUS

WORKSHEET subcommand to obtain the dump title. Obtain the component name, component identifier, or module name from the title.

In the following example, STATUS WORKSHEET output, the dump title is for the XES component.

```
                             MVS Diagnostic Worksheet
 Dump Title: COMPON=IXL,COMPID=5752SCIXL,ISSUER=IXLM1REC,MODULE=IXLM2SP
              ,ABEND=S0071,REASON=00000030

 CPU Model 9021 Version A6 Serial no. 300359 Address 03
 Date: 03/30/93     Time: 10:32:38  Local

 Original dump dataset: SYS1.DUMP32
```

4. From the module prefix, identify the component, subsystem, or product, if provided by IBM.

   Use the module name to query the SMP/E zone for a module entry with that module name. If the search does not find a match, the module is not an IBM module. If the search indicates a match, use the FMID to positively identify the product.
5. Continue diagnosis as follows:
   - If all the addresses are in components of z/OS, continue with "Searching the problem reporting databases" on page 215.
   - If all the addresses are in an IBM subsystem or product, continue diagnosis with the diagnosis book for the subsystem or product.
   - If all the addresses are in components of z/OS and in an IBM subsystem or product, continue with "Searching the problem reporting databases" on page 215 and with the diagnosis book for the subsystem or product.
   - If any of the addresses are in an installation-provided program, including an installation exit routine, continue diagnosis with that program, using the dump.

     If some addresses are in the program or routine and some in system modules, the loop is probably in the program or routine and includes one or more requests for system services.

Related information:
- See *z/OS MVS IPCS Commands* for the IPCS subcommands.
- See the IBM component, subsystem, product program identifier or module prefix in *z/OS MVS Diagnosis: Reference*.
- See *z/OS MVS Diagnosis: Tools and Service Aids*for information about analyzing an SVC dump for a problem in an installation-provided program.

Analyzing a logrec error record

Analyze a logrec error record for a disabled loop. The excessive-spin logrec error record can identify the module running on the processor causing the spin condition.

Steps for analyzing a logrec error record

Before you begin: You need access to the logrec error record.

Use the following steps to identify the module running on the processor causing the spin condition, as follows:

1. Locate the 16 FRR addresses from the stack that was current when the target processor was restarted. These addresses appear after the identification text EX SPIN RECORD at the start of the VRA.
2. Identify the current FRR on the stack from the INDEX=*x* value that follows the sixteen addresses. The value of *x* can be 0 through 16.

   If *x* is 0, the stack contains no current FRRs. Otherwise *x* is an index indicating which of the 16 addresses points to the current FRR. For example, if *x* is 2, the second address points to the current FRR.
3. Use a storage map to identify the component that owns the FRR at this address.

For example: In the following output example, the current FRR is 81319571, which is the third FRR in the stack. This is the current FRR because INDEX=03.

VERBEXIT LOGDATA output — FRR Stack

```
+012    C6D9D9E2                                       |FRRS            |

+016    KEY: 38     LENGTH: 40
+018    811F43F8    811D74ED    81319571    81086D05    |A..8A...A.N.A._.|
+028    811D74ED    00000000    00000000    00000000    |A...............|
+038    00000000    00000000    00000000    00000000    |................|
+048    00000000    00000000    00000000    00000000    |................|

+058    KEY: 37     LENGTH: 06
+05A    C9D5C4C5    E77E                               |INDEX=          |

+060    KEY: 38     LENGTH: 01
+062    03                                             |.               |
```

Searching the problem reporting databases

Search arguments are used to search problem reporting databases. If the problem you are diagnosing was already reported and the symptoms are in the database, the search produces a match. Searching is an iterative process; you might need to gather additional data and continue your search.

Steps for searching the problem reporting databases

Use the following steps to search the problem reporting databases and determine if the problem was previously reported:

1. Create a search argument using the symptoms applicable for the type of loop being diagnosed:
   a. Disabled loop search argument - LOOP module/CSECT name
   b. Enabled loop search argument - LOOP module/CSECT name
   c. Spinloop search argument - ABEND071 module/CSECT name.

   Tip: Use free-format search arguments. For more information, see "Searching problem reporting databases" on page 8.
2. If the search finds no match, remove some symptoms or add some symptoms. Search again. Continue searching for matches by adding and removing symptoms.

   If the search finds that the problem was previously reported, request the fix.
3. If the search does not produce a match, continue with the next step. Use problem data from the preceding steps to create more symptoms; use these symptoms in later searches.

4. If you still cannot find the cause of the loop or if the problem is new, report the problem to IBM using the procedure in Chapter 23, “Reporting problems to IBM,” on page 287. Provide the following problem data as listed in Chapter 24, “Problem diagnostic worksheet,” on page 291.
   - Problem type: type of loop
   - Search argument
   - Dump, formatted by IPCS, online or printed
   - Range of loop
   - SMF records, if obtained
   - Accompanying messages: identifiers and texts
   - Hard-copy log, beginning 15 to 30 minutes before the problem, or master trace, if not wrapped between the problem and dump
   - All printed output and output data sets related to the problem
   - Data on any related problems
   - Module name and level
   - Name and level of the operating system(s) with a list of program temporary fixes (PTF) applied at the time of the problem and all installation modifications, exits, and products with other than Class A service
   - Other problem data developed while using the procedures in this document or other diagnosis books for the component, subsystem, or program

Related information

- See “Searching problem reporting databases” on page 8 for more information.
- See *z/OS MVS IPCS Commands* for the IPCS subcommands.

Chapter 14. Diagnosing an output problem

Overview of analyzing output problems

Most output problems occur during installation and testing of new functions or applications.

This chapter explains what information you will need to properly diagnose an output problem.

Symptoms of output problems: Your output is incorrect, incomplete, or missing, but messages indicate successful processing.

- **Incorrect output**: The processing produced all the expected output, but some output is incorrect. For example:
  - Some values in a report are wrong
  - The text in a message is incorrect
  - The return or reason code is not valid
  - The records in are not producing expected records out.
- **Incomplete output**: The processing did not produce all the expected output. For example, a column is missing from a report.
- **Missing output**: Some or all of the expected output is missing. For example, a report is missing.

Steps for diagnosing output problems

Before you begin: You need to have IPCS installed and have access to the following information:

- Job log
- Input and output data sets for the program
- The JCL for the program
- System log
- Logrec data set

Use the following flowchart to guide you through diagnosing an output problem:

Output problem analysis

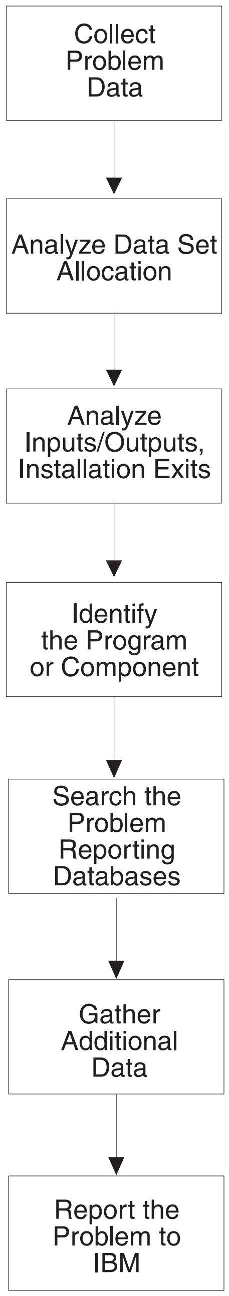

Collect Problem Data
Analyze Data Set Allocation
Analyze Inputs/Outputs, Installation Exits
Identify the Program or Component
Search the Problem Reporting Databases
Gather Additional Data
Report the Problem to IBM

Figure 46. Flowchart for output problem analysis

The following steps will guide you through diagnosing output problems:

1. "Collecting problem data for an output problem" on page 219
2. "Analyzing data set allocation for an output problem" on page 219
3. "Analyzing the inputs and outputs" on page 220
4. "Analyzing installation exits for an output problem" on page 221
5. "Identifying the program or component" on page 221
6. "Searching the problem reporting databases for an output problem" on page 223
7. "Gathering additional data for output problems" on page 223
   - "Messages and logrec for output problems" on page 223

- “Determine path for output problems” on page 224
- “Teleprocessing for output problems” on page 224

8. “Reporting output problems to IBM” on page 225

Collecting problem data for an output problem

To properly diagnose an output problem you need to collect as much information as possible.

Step for collecting problem data

- If the output is a data set, collect the following:
  - All input data sets for the program
  - Input macros, commands, and statements that are used to request output from the program
  - All output data sets produced by the program
  - The job log
- If the output is a message, return code, or reason code, collect the incorrect or incomplete message or code.

Analyzing data set allocation for an output problem

For missing data set output, analyze data set allocation.

Steps for analyzing data set allocation

1. Look for messages in the job log indicating that all data sets used by the program were properly allocated and unallocated. Look for:
   - The data set that should have contained the output
   - The data set that contained the input used by the program to create the output
   - Indications that the output was sent to another data set or different system
2. If problems are found, correct the JCL.
3. If the problem is not found, continue diagnosis with “Analyzing the inputs and outputs” on page 220.

Related information:

- See *z/OS MVS JCL Reference* for JCL coding.
- For explanations of the messages, see:
  - *z/OS MVS System Messages, Vol 1 (ABA-AOM)*
  - *z/OS MVS System Messages, Vol 2 (ARC-ASA)*
  - *z/OS MVS System Messages, Vol 3 (ASB-BPX)*
  - *z/OS MVS System Messages, Vol 4 (CBD-DMO)*
  - *z/OS MVS System Messages, Vol 5 (EDG-GFS)*
  - *z/OS MVS System Messages, Vol 6 (GOS-IEA)*
  - *z/OS MVS System Messages, Vol 7 (IEB-IEE)*
  - *z/OS MVS System Messages, Vol 8 (IEF-IGD)*
  - *z/OS MVS System Messages, Vol 9 (IGF-IWM)*
  - *z/OS MVS System Messages, Vol 10 (IXC-IZP)*

Analyzing the inputs and outputs

For incorrect or incomplete data set output, analyze the inputs and outputs.

Steps for analyzing the inputs and outputs

Perform the following steps:

1. Compare the input and the output. For example, if a device model number is wrong in an IOS report, compare it to the model number specified on the in the IODF.
2. Compare the output received to examples of the output shown in the user's guide for the request.
3. Check the call, command, macro, or statement used to request the output. Make sure that all fields contain desired values.
4. Match your findings to the following:
   - For missing data set output, check the macro, command, or statement used to request the output. Make sure that the missing output was supposed to be received.
   - For an incorrect or incomplete message, match the message received to the message in the book with its explanation.
   - For incorrect return and reason codes, match the codes to the expected codes. Make sure that the code received and the code in the book are the same type: both hexadecimal or both decimal.

   For the message text or a return or reason code in a message, see:
   - *z/OS MVS System Messages, Vol 1 (ABA-AOM)*
   - *z/OS MVS System Messages, Vol 2 (ARC-ASA)*
   - *z/OS MVS System Messages, Vol 3 (ASB-BPX)*
   - *z/OS MVS System Messages, Vol 4 (CBD-DMO)*
   - *z/OS MVS System Messages, Vol 5 (EDG-GFS)*
   - *z/OS MVS System Messages, Vol 6 (GOS-IEA)*
   - *z/OS MVS System Messages, Vol 7 (IEB-IEE)*
   - *z/OS MVS System Messages, Vol 8 (IEF-IGD)*
   - *z/OS MVS System Messages, Vol 9 (IGF-IWM)*
   - *z/OS MVS System Messages, Vol 10 (IXC-IZP)*
   - *z/OS MVS Dump Output Messages*
   - The message book for a subsystem or program.

   For a return or reason code with an abend code, see
   - *z/OS MVS System Codes*

   For a return or reason code for a macro, see:
   - *z/OS MVS Programming: Authorized Assembler Services Reference ALE-DYN*
   - *z/OS MVS Programming: Authorized Assembler Services Reference EDT-IXG*
   - *z/OS MVS Programming: Authorized Assembler Services Reference LLA-SDU*
   - *z/OS MVS Programming: Authorized Assembler Services Reference SET-WTO*
   - *z/OS MVS Programming: Authorized Assembler Services Guide*
   - *z/OS MVS Programming: Assembler Services Reference ABE-HSP*
   - *z/OS MVS Programming: Assembler Services Reference IAR-XCT*
   - *z/OS MVS Programming: Assembler Services Guide*.

Analyzing installation exits for an output problem

Analyze all installation exit routines used in obtaining the output.

Steps for analyzing installation exits

1. Check for problems in the logic of each exit routine. Many installation exits are invoked using the dynamic exits facility, which allows an updated exit to be refreshed into the system.
2. If no logic problems are found, remove the options that cause each exit routine to be invoked.
3. Rerun the program.
4. If this action stops the problem, the problem is in the exit routine that was eliminated. Continue diagnosis with that routine.

Related information:

See *z/OS MVS Installation Exits* for more information on coding installation exit routines.

Identifying the program or component

Identifying the component that is involved with the output enables you to determine the source of the problem.

Steps for identifying the program or component

Identify the program or component involved with the output from one of the following steps:

1. For output from a batch job, obtain the program name from the PGM parameter on the JCL EXEC statement. In the following example the program name is obtained from the JCL EXEC statement:

   In the following example, the name of a program involved with the output is IKJEFT01. It can be found on the PGM parameter of the EXEC statement:

   ```
   //*
   //*=====================================================
   //* Batch TSO job (PGM=IKJEFT01)
   //*=====================================================
   //IKJEFT01 EXEC PGM=IKJEFT01,REGION=4096K,DYNAMNBR=50
   //*
   ```
2. Identify the program or component involved with the output from one of the following:
   - For output from interactive work, use the command being processed to identify the program.
   - For an error message, use the message prefix to identify the program or component or look for the component listed in the message explanation. Look at the preface of any of the z/OS MVS system messages books to find the element or component that the message is associated with and the book where you can find the explanation of the message:

     For example, if you have an incorrect message with message number AHL002A, you can tell from the preface of *z/OS MVS System Messages, Vol 1 (ABA-AOM)* that the message was issued by GTF, and the explanation can be found in the same book:

Table 22. Example of using the message prefix to identify the component

Prefix	Component	Book Title	Order Number
AHL	Generalized trace facility (GTF)	• *z/OS MVS System Messages, Vol 1 (ABA-AOM)*, • *z/OS MVS Dump Output Messages*	• SA22-7631 • SA22-7590

3. For a return or reason code accompanying an abend, see the component listed in the explanation of the code. The following example identifies the component issuing an abend:

   00D

   Explanation: An error occurred during processing of a CTRACE or CTRACEWR macro. Register 15 contains xxnnnnxx where nnnn is a reason code that further describes the error.

   Reason Code (hex)
   Explanation

   0001 For the CTRACE macro, the parameter list version number is incorrect.

   0002 For the CTRACE macro, the component name either does not begin with an alphabetic or national character, or it contains one or more characters that are not alphanumeric or national characters.

   Source:
   Component trace
4. For a component of z/OS, continue with “Searching the problem reporting databases for an output problem” on page 223.
5. For an IBM subsystem or product, continue diagnosis with the subsystem or product.
6. For an installation-provided program, including an installation exit routine, continue diagnosis with that program.

Related information:
- See *z/OS MVS JCL Reference* for the EXEC statement.
- For message prefixes for IBM components, subsystems, and products, see the following books:
  – *z/OS MVS System Messages, Vol 1 (ABA-AOM)*
  – *z/OS MVS System Messages, Vol 2 (ARC-ASA)*
  – *z/OS MVS System Messages, Vol 3 (ASB-BPX)*
  – *z/OS MVS System Messages, Vol 4 (CBD-DMO)*
  – *z/OS MVS System Messages, Vol 5 (EDG-GFS)*
  – *z/OS MVS System Messages, Vol 6 (GOS-IEA)*
  – *z/OS MVS System Messages, Vol 7 (IEB-IEE)*
  – *z/OS MVS System Messages, Vol 8 (IEF-IGD)*
  – *z/OS MVS System Messages, Vol 9 (IGF-IWM)*
  – *z/OS MVS System Messages, Vol 10 (IXC-IZP)*
- See *z/OS MVS System Codes* for explanations of the abend codes.

- See *z/OS MVS Diagnosis: Reference* to relate an IBM component, subsystem or product to a program identifier or module prefix.

Searching the problem reporting databases for an output problem

Use free-format search arguments to search the problem reporting databases. If the problem you are diagnosing was already reported and the symptoms are in the database, the search produces a match. Searching is an iterative process; you might need to gather additional data and continue your search.

Step for searching the problem reporting database

Use the following steps to search a problem reporting database to determine if this is a known problem.

1. Use a free-format search argument developed from the symptoms. For more information, see "Searching problem reporting databases" on page 8.
2. If the search finds no match, remove some symptoms or change the symptoms. Search again. Continue searching for matches by adding, removing, and changing symptoms.
3. If the search finds that the problem was previously reported, request the problem fix. If not, continue with the next step. Use problem data from following steps to create more symptoms; use these symptoms in later searches.

Related information:

See "Searching problem reporting databases" on page 8 for more information on developing a search argument.

Gathering additional data for output problems

Gathering additional data will increase your chances of finding a match in the problem reporting databases. Use the procedures outlined in this section to gather additional data and continue searching the problem reporting databases.

Steps for gathering additional information for output problems

Use the following steps to collect additional information about:

- "Messages and logrec for output problems"
- "Determine path for output problems" on page 224
- "Teleprocessing for output problems" on page 224

Messages and logrec for output problems

1. Collect and analyze messages and logrec records about the problem. Look at any messages or software, symptom, and hardware records for logrec around the time of the problem.
2. Look in the following:
   - Job log
   - TSO/E user's ISPF transaction log or session manager log
   - System log (SYSLOG) for the console with master authority or the alternate console
   - Logrec data set, formatted by EREP.
3. Check for:
   - I/O errors that could affect the output

- Operator interactions that could affect the output
- Problems with the access method or function involved: For example, VSAM, BTAM, JES, or WTO.

Determine path for output problems

1. Analyze the path the data should take.
2. For the request being processed, determine the correct path for the data from input to output.
3. Determine each program and component involved.
4. Check for messages about problems in these programs and components.
5. If an installation-provided exit routine receives control during the processing, check the routine.
6. Look at the environment. Specifically, look for recent hardware and software changes to the system and to any applications. A change in one program can affect others; for example, a change to an application that updates a database affects all other users of the database.
7. If needed, recreate the problem, using a SLIP trap and traces to obtain the data needed to isolate the problem.

Teleprocessing for output problems

1. Analyze the path the data should take.
2. Determine how the data flows through the programs and components that process it and through the systems and hardware. Use this knowledge to recreate the problem, using traces to checkpoint the data at certain spots along the path.
3. Track the data from a point where it was correct to a point where the data stopped or became incorrect.

Related information:

- For formatting of logrec records, see Recording logrec error records in *z/OS MVS Diagnosis: Tools and Service Aids*.
- See *z/OS MVS Diagnosis: Tools and Service Aids* for the logrec records.
- For explanations of the messages, see:
  - *z/OS MVS System Messages, Vol 1 (ABA-AOM)*
  - *z/OS MVS System Messages, Vol 2 (ARC-ASA)*
  - *z/OS MVS System Messages, Vol 3 (ASB-BPX)*
  - *z/OS MVS System Messages, Vol 4 (CBD-DMO)*
  - *z/OS MVS System Messages, Vol 5 (EDG-GFS)*
  - *z/OS MVS System Messages, Vol 6 (GOS-IEA)*
  - *z/OS MVS System Messages, Vol 7 (IEB-IEE)*
  - *z/OS MVS System Messages, Vol 8 (IEF-IGD)*
  - *z/OS MVS System Messages, Vol 9 (IGF-IWM)*
  - *z/OS MVS System Messages, Vol 10 (IXC-IZP)*
  - *z/OS MVS Dump Output Messages*
  - The message book for a subsystem or program
- See *z/OS Communications Server: SNA Diagnosis Vol 1, Techniques and Procedures* and *z/OS Communications Server: SNA Diagnosis Vol 2, FFST Dumps and the VIT* to diagnose VTAM® problems.

- See the SLIP chapter in *z/OS MVS System Commands* for information on setting a SLIP trap.
- See *z/OS MVS Diagnosis: Tools and Service Aids* for requesting dumps and traces.
- See *z/OS MVS Installation Exits* for exit routines.

Reporting output problems to IBM

If you have completed the procedures listed in this chapter, cannot find a match in the problem reporting databases, and you believe the failure was caused by a defect in IBM code, call the IBM Support Center.

Provide the following problem data:

- Problem type: INCORROUT, that is, incorrect, incomplete, or missing output in a data set, message, return code, or reason code
- Search argument
- All input associated with the problem, including all data sets, commands, macros, and statements
- All output associated with the problem, including data sets, reports, and records
- JCL for all data sets involved
- Source code for any exit routine involved
- Accompanying messages: identifiers and texts
- Hard copy log, beginning 30 to 60 minutes before the problem, or master trace, if not wrapped between the problem and dump
- Name and level of the operating system(s) with a list of program temporary fixes (PTF) applied at the time of the problem and all installation modifications, exits, and products with other than Class A service

Related information:

- See Chapter 23, “Reporting problems to IBM,” on page 287 for more information on reporting a problem.
- For formatting of logrec records, see the topic on recording logrec error records in *z/OS MVS Diagnosis: Tools and Service Aids*.

Chapter 15. Diagnosing a performance problem

Overview of a performance problem

Most performance problems appear as unacceptable response times or resource usage. For example, system processing is slow because a program is using an excessive amount of system resources.

Symptoms of a performance problem:
- Jobs take more time to execute than normal
- Poor or erratic response time exists
- Service level objectives are being exceeded
- Users complain about slow response time
- Unexpected changes occur in response times or resource utilizations
- Other indicators show stress
- Monitor III Workflow/Exceptions occur.
- System resource indicators occur (for example, paging rates, DASD response)
- Expected throughput on the system is not being attained
- CP utilization for a job or address space is higher than normal.

Steps for diagnosing a performance problem

Before you begin: You need access to following types of information:
- System messages
- SVC dump
- Job log
- TSO/E user's ISPF transaction log or session manager log
- System log
- Logrec data set
- z/OS Resource Measurement Facility (RMF), SMF, or other system monitoring programs, such as IBM OMEGAMON z/OS Management Console.

When diagnosing a performance problems, using Predictive Failure Analysis (PFA) and Runtime Diagnostics can help you eliminate whether the problem is a damaged job or system. See Chapter 4, “Runtime Diagnostics,” on page 35 and Chapter 7, “Predictive Failure Analysis overview and installation,” on page 65.

RMF also provides comprehensive guidance for diagnosing performance problems. Diagnosing a problem: the first steps in *z/OS RMF User's Guide* offers a practical, task-oriented approach to analyzing performance issues using RMF.

Use the following flowchart to guide you through diagnosing a performance problem:

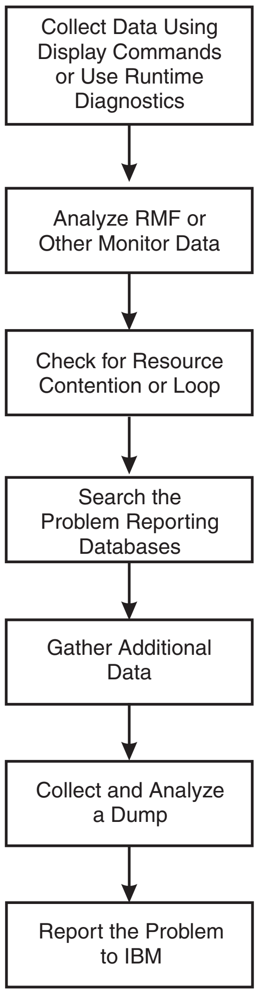

Collect Data Using Display Commands or Use Runtime Diagnostics
Analyze RMF or Other Monitor Data
Check for Resource Contention or Loop
Search the Problem Reporting Databases
Gather Additional Data
Collect and Analyze a Dump
Report the Problem to IBM

Figure 47. Flowchart for performance problem analysis

Use the following steps to guide you through diagnosing a performance problem:

1. "Collecting data using commands"
2. "Checking for resource contention or loop" on page 231
3. "Searching the problem reporting database" on page 232
4. "Gathering additional data for performance problems" on page 233
5. "Analyzing a dump for performance problems" on page 234
6. "Reporting performance problems to IBM" on page 235

Collecting data using commands

Collecting information with commands will give you a better understanding of the source of the problem. In a JES2 system, use JES2 commands to determine why JES2 is not able to schedule work. This topic is divided into two sections:

- "Steps for collecting data using DISPLAY"
- "Steps for using JES2 commands to collect data" on page 230

Steps for collecting data using DISPLAY

Use the following `DISPLAY` commands in the order listed to find the source of your problem:

1. Determine if the system is waiting for an operator action, for example, mounting of a volume. Other jobs might have to wait until the action is completed. In this case, the operator should perform the action. The following command displays outstanding messages requiring operator action:

   ```
   DISPLAY R,LIST
   ```

   In the following example of DISPLAY R,LIST output, there is one message, IEF434D, requiring immediate operator action. The system waits until a valid reply is entered. The operator should enter a valid reply to this message.

   ```
   IEE112I 13.39.37 PENDING REQUESTS    FRAME LAST   F      E   SYS=SY1
   RM=0    IM=0     CEM=2     EM=0     RU=0    IR=0    AMRF
   ID:R/K     T JOB ID   MESSAGE TEXT
            3 C          *ILR005E PLPA PAGE DATA SET FULL, OVERFLOWING TO
                          COMMON DATA SET
            5 R          *IEF434D CRITJOB - INVALID REPLY. REPLY 'HOLD'
                          OR ' OHOLD'
   ```
2. Look for the generalized trace facility (GTF) in the started tasks. GTF may slow performance. A job step name of STARTING indicates that the system has not yet successfully completed initiation of the first step. If unsuccessful initiation of the job continues, diagnose this problem. The following command displays detailed information for active jobs and started tasks.

   ```
   DISPLAY A,LIST
   ```

   In the following example of DISPLAY A,LIST output, all of the steps have been successfully initiated. However, the performance problem could be due to GTF being active:

   ```
   IEE114I 14.51.49 93.181 ACTIVITY     FRAME LAST   F      E   SYS=SY1
    JOBS     M/S    TS USERS    SYSAS    INITS   ACTIVE/MAX VTAM     OAS
   00000    00003    00000      00016    00000    00000/00000       00000
    LLA      LLA      LLA      NSW  S  VLF      VLF      VLF      NSW  S
    JES2     JES2     IEFPROC  NSW  S
    GTF      GTF      IEFPROC  NSW  S
   ```
3. Look for the loss of a hardware component indicated by a message in the hard copy log. This loss might be causing jobs to wait. In this case, correct the hardware problem. The `DISPLAY M` command displays the hardware configuration.

   ```
   DISPLAY M
   ```
4. Look for the name and status of current SLIP traps.
   a. Enter the DISPLAY SLIP command for a summary of SLIPs that are running on the system:

      ```
      DISPLAY SLIP
      DISPLAY SLIP=xxxx
      ```

      In the Figure 48 output using the `DISPLAY SLIP` command shows the name and status of the current SLIP traps.

```
IEE735I 13.42.00 SLIP DISPLAY        FRAME LAST   F      E   SYS=SY1
ID   STATE    ID   STATE    ID   STATE    ID   STATE    ID   STATE
X013 ENABLED  X028 ENABLED  X0E7 ENABLED  X0F3 ENABLED  X13E ENABLED
X222 ENABLED  X322 ENABLED  X33E ENABLED  X622 ENABLED  X804 ENABLED
X806 ENABLED  X80A ENABLED  X9FB ENABLED  XB37 ENABLED  XD37 ENABLED
XE37 ENABLED
```

Figure 48. Output from the DISPLAY SLIP command

   b. Pick a SLIP ID that is `ENABLED` and enter `DISPLAY SLIP=`*xxxx* to check for enabled PER traps. In this case, disable the traps.

In Figure 48 on page 229, all the SLIPs appear enabled. Entering `DISPLAY SLIP=X013` yields the following enabled PER trap with an action of STRACE, which can slow performance:

```
IEE735I 09.14.03 SLIP DISPLAY        FRAME LAST   F      E   SYS=SY1
ID=X013,PER-SB,ENABLED(ACTIVE),ACTION=STRACE,SET BY TSO KLOGAN
```

Related information:

- See *z/OS MVS System Commands* for the SLIP and DISPLAY commands.
- See *z/OS MVS Diagnosis: Tools and Service Aids* for information about GTF.

Steps for using JES2 commands to collect data

Use the following procedures to collect problem data with JES2 commands:

1. Use the job entry subsystem display commands to find the status of jobs, queues, printer setups, requirements of SYSOUT data sets, and other problem data.

   ```
   $D J1-9999
   ```

   $D J1-99 output displays the status of jobs. If the display shows that a range of jobs has been held, use the JES2 $A J command to release specific jobs. Or, use the JES2 $A A command to release all jobs in the system.

   In the following **$D J1-99** output, all of the displayed jobs are being held:

   ```
   $D J1-99

   JOB00005  $HASP608 IEBGENER AWAITING HARDCOPY         PRIO HELD 15 ANY
   JOB00006  $HASP608 XEQN1    AWAITING XMITTER POK      PRIO HELD 9 ANY
   JOB00007  $HASP608 IEBGENER AWAITING HARDCOPY         PRIO HELD 15 ANY
   JOB00008  $HASP608 IEBGENER AWAITING EXECUTION A      PRIO HELD 9 ANY
   ```

2. Use the following command to display the status of tasks started under JES2:

   ```
   $D S1-9999
   ```

   The following **$D S1-99** output shows tasks started under JES2:

   ```
   $D S1-99
    STC00001  $HASP608 SYSLOG   EXECUTING $              PRIO 15 IBM2
    STC00002  $HASP608 $MASCOMM AWAITING HARDCOPY        PRIO 15 ANY
    STC00003  $HASP608 INIT     EXECUTING $              PRIO 15 IBM2
     INITASID=0015
    STC00004  $HASP608 IRRDPTAB ON PRT1                  PRIO  1 IBM2
   ```

3. Use the following command to display the status of time-sharing users:

   ```
   $D T1-9999
   ```

4. Use the following command to display the status of data set groups queued for output and the status of JES2-controlled local printers.

   ```
   $D F
   $D U,PRTS
   ```

   If these displays show that no printers are set up with the needed forms, use the **JES2 $T PRTnnnn** command to change a printer's setup to the needs of the output forms queue.

   In the following **$D F and $D U,PRTS** output, there is only one item queued for the printers. All of the printers, however, are either drained or halted, which means that none of the printers are started. No printing can occur.

   ```
   $D F

   $HASP621 OUT R=LOCAL            F=STD      C=**** T=**** W= (NONE)
   $HASP621 PRMODE=LINE     CLASS A=3

   $D U, PRTS
   ```

```
$HASP603 PRT1      UNIT=0017,STATUS=HALTED,(STC00004 IRRDPTAB)
$HASP603 PRT2      UNIT=0002,STATUS=DRAINED
$HASP603 PRT3      UNIT=0017,STATUS=DRAINED
$HASP603 PRT4      UNIT=000E,STATUS=DRAINED
$HASP603 PRT5      UNIT=000F,STATUS=DRAINED
$HASP603 PRT6      UNIT=000F,STATUS=DRAINED
```

5. Use the following command to display the number of queued jobs.

   ```
   $D Q
   ```

   If the problem is that jobs are not running because they are held, you can use the JES2 `$A` command with appropriate parameters to release the jobs.

Related information:

See *z/OS JES2 Commands* for the JES2 commands.

Checking for resource contention or loop

You can analyze output from RMF, SMF or another system monitoring program to look for resource contention and loops.

Steps for checking resource contention

Before you begin:

Perform the following steps to check for resource contention:

1. Use output from RMF, SMF or another system monitoring program to look for problems. Find someone in your installation who is familiar with the program and can interpret the output. The following lists some of the potential problems to look for:
   - A program using a lot of storage, whether it is real, virtual, auxiliary or extended storage.
   - Data set contention
   - ENQ contention
   - Tuning problems
   - System running over capacity
2. Identify the program, job or function that is involved in the performance problem. If using more resources than normal, gather RMF, trace output, or both from the time frame of when the slowdown occurred.

   For a problem caused by resource contention, use dump output from “Analyzing a dump for performance problems” on page 234 or the following two choices:
   - Use the ANALYZE output to identify the problem causing the contention:

     In the following output, resource #0002, which is device 687, shows an intercept condition and is not running. There is one unit of work waiting for this device.

     ```
                      CONTENTION EXCEPTION REPORT

      JOBNAME=IOS.      ASID=0001  UCBTAPE=00FC72C0

      JOBNAME=IOS.     HOLDS THE FOLLOWING RESOURCE(S):

         RESOURCE #0002:  There are 0001 units of work waiting
                               for this resource
           NAME=I/O Device  687 (TAPE    ) VOLSER=......
     ```

```
              DATA=(IOS) Active I/O with ASSIGN held.
                   (IOS) Device not ready.
                   (IOS) Intercept condition.

         STATUS FOR THIS UNIT OF WORK:
           IRA10102I This address space is on the SRM IN queue.
```

- For a problem caused by a batch program and identified through JES2 commands, obtain the program name from the PGM parameter on the JCL EXEC statement.

  In the following example for obtaining the program name, the name of the program involved with the output is IKJEFT01. It can be found on the PGM parameter of the EXEC statement:

  ```
  //*
  //*===================================================
  //* Batch TSO job (PGM=IKJEFT01)
  //*===================================================
  //IKJEFT01 EXEC PGM=IKJEFT01,REGION=4096K,DYNAMNBR=50
  //*
  ```

3. Continue diagnosis depending on what program caused the problem.
   - For a component of z/OS, continue with “Searching the problem reporting database.”
   - For an IBM subsystem or product, continue diagnosis with the subsystem or product.
   - For an installation-provided program, including an installation exit routine, continue diagnosis with that program.

Related information:

- See *z/OS MVS System Management Facilities (SMF)* for the SMF reports.
- See *z/OS RMF User's Guide* for information.
- See *z/OS MVS IPCS Commands* for the ANALYZE subcommand.
- See *z/OS MVS JCL Reference* for the EXEC statement.
- See *z/OS MVS Diagnosis: Reference* to find the IBM component, subsystem, or product for a program identifier or module prefix.

Searching the problem reporting database

Search arguments are used to search problem reporting databases. If the problem you are diagnosing was already reported and the symptoms are in the database, the search produces a match. Searching is an iterative process; you might need to gather additional data and continue your search.

Steps for searching the problem reporting databases

1. Search a problem reporting database to determine if the problem was previously reported.

   Include any of the following free-format search arguments that apply to the performance problem, and the word **PERFORMANCE**. For example:
   - System address space name
   - Function name
   - Product involved
   - Module or CSECT name
   - Resource names
   - Message ID

- Symptoms created from information in STATUS CPU output

For more information, see “Searching problem reporting databases” on page 8.

2. If the search finds no match, remove some symptoms or add some symptoms. Search again. Continue searching for matches by adding or removing symptoms.
3. If the search finds that the problem was previously reported, request the problem fix. If not, continue with “Gathering additional data for performance problems.” Use the problem data gathered there to create more symptoms; use these symptoms in later searches.

Related information:

- See “Searching problem reporting databases” on page 8.
- See *z/OS MVS IPCS Commands* for the STATUS and VERBEXIT SYMPTOM subcommands.

Gathering additional data for performance problems

Gathering additional data will increase your chances of finding a match in the problem reporting databases. Use the procedures outlined in this section to gather additional data and continue searching the problem reporting databases.

Steps for gathering additional information for performance problems

1. Collect and analyze messages and logrec records about the problem. Use time stamps to select messages and software, symptom, and hardware records for logrec. Look at any messages or records that occurred before and during the time the problem was first reported. Look in the following:
   - The job log
   - A TSO/E user's ISPF transaction log or session manager log
   - The system log (SYSLOG) for the console with master authority or the alternate console
   - VERBEXIT MTRACE dump output, which shows the buffer for system messages
   - VERBEXIT LOGDATA dump output, which formats the logrec buffer
   - Logrec data set, formatted by EREP
2. Use IPCS to look at the dump.

Related information:

- See *z/OS MVS IPCS Commands* for the VERBEXIT LOGDATA and VERBEXIT MTRACE subcommands.
- For formatting of logrec records, see Recording logrec error records in *z/OS MVS Diagnosis: Tools and Service Aids*.
- For explanations of the messages, see:
  - *z/OS MVS System Messages, Vol 1 (ABA-AOM)*
  - *z/OS MVS System Messages, Vol 2 (ARC-ASA)*
  - *z/OS MVS System Messages, Vol 3 (ASB-BPX)*
  - *z/OS MVS System Messages, Vol 4 (CBD-DMO)*
  - *z/OS MVS System Messages, Vol 5 (EDG-GFS)*
  - *z/OS MVS System Messages, Vol 6 (GOS-IEA)*

- *z/OS MVS System Messages, Vol 7 (IEB-IEE)*
- *z/OS MVS System Messages, Vol 8 (IEF-IGD)*
- *z/OS MVS System Messages, Vol 9 (IGF-IWM)*
- *z/OS MVS System Messages, Vol 10 (IXC-IZP)*
- See the message book for a subsystem or program.
- See *z/OS MVS Dump Output Messages* for dump output messages.

Analyzing a dump for performance problems

When all else fails, analyzing a dump might yield some additional information about performance problems. A dump captures a moment in time during the system execution and potentially a small amount of SYSTRACE data. This is typically not enough data to diagnose performance problems, but you might find a loop or a resource contention for the jobs that are performing poorly.

Steps for collecting and analyzing a dump for performance problems

Before you begin: Request a dump. If you did not obtain a dump, recreate the problem and request the dump using the following steps:

1. Enter a **DUMP** command to request an SVC dump and reply with SDATA options to dump global resource serialization control blocks and the nucleus.
2. Specify the address spaces that are experiencing performance reduction. For example, if the problem appears to be in JES2 or JES3, specify the JES address space in the reply.

   The following example shows a request for an SVC dump with a sample of the parameters you might use:

   ```
   DUMP COMM=(text)

   REPLY id,ASID=1,SDATA=(GRSQ,NUC,CSA,SQA,TRT),END
   ```

   Tip: Be sure to give your dumps meaningful names.
3. Use IPCS to format the dump.
4. If your dump contains the JES address space and your problem appears to be in JES2 or JES3, format the dump using IPCS as follows:
   - For a JES2 system, select JES2 in the Component Analysis panel. Then select JES2 Control Blocks from the JES2 Component Data Analysis panel. From the JES2 Control Block List panel, select the control blocks you wish to format.
   - For a JES3 system, select JES3 in the Component Analysis panel. Then select JES3 Control Block Information from the IPCS JES3 - Primary Options panel. You can also use JMF to analyze performance problems for JES3.

Related information:

- See *z/OS MVS Diagnosis: Tools and Service Aids* for requesting an SVC dump.
- See *z/OS MVS System Commands* for the DUMP operator command.
- See *z/OS JES2 Diagnosis* for information on diagnosing JES2 problems and using IPCS for JES2 diagnosis.
- See *z/OS JES3 Diagnosis* for information on diagnosing JES3 problems, using IPCS for JES3 diagnosis, and JMF.

Reporting performance problems to IBM

If you have completed the procedures listed in this chapter, cannot find a match for your performance problem in the problem reporting databases, and believe the problem is a defect in IBM code, call the IBM Support Center.

Provide the following problem data:

- Problem type: performance
- Search argument
- Dump, formatted by IPCS, online or printed
- System responses to DISPLAY and JES commands
- Parmlib members analyzed
- SMF records, if obtained in steps from the hang or wait procedure
- Hard copy log, beginning 30 to 60 minutes before the problem, or master trace, if not wrapped between the problem and dump
- Logrec records, beginning 30 to 60 minutes before the problem
- All printed output and output data sets related to the problem
- Name and level of the operating system(s) with a list of program temporary fixes (PTF) applied at the time of the problem and all installation modifications, exits, and products with other than Class A service

Related information:

- See Chapter 23, “Reporting problems to IBM,” on page 287 for more information.
- For formatting of logrec records, see Recording logrec error records in *z/OS MVS Diagnosis: Tools and Service Aids*.

Part 5. Diagnosing component-specific problems

After you identify a component-specific problem, use this information as a guide for operational problem determination of the problem. Each component section contains basic commands and function you can use for determining problems and collecting and sending documentation to IBM support. Additionally, where possible each component identifies component-specific problems, and supplies:

- Problem symptoms
- Investigation techniques
- Recovery actions
- Actions to avoid recurrence

See:

- Chapter 16, “Catalog component operational problem determination,” on page 239
- Chapter 17, “PDSE operational problem determination,” on page 243
- Chapter 18, “RRS operational problem determination,” on page 251
- Chapter 19, “System Data Mover (SDM) operational problem determination,” on page 265
- Chapter 20, “VSAM component operational problem determination,” on page 271
- Chapter 21, “VSAM record-level sharing (RLS) operational problem determination,” on page 277

.

Chapter 16. Catalog component operational problem determination

The major commands used to help in problem determination are: F CATALOG,LIST – Which shows the active tasks in the catalog address space D GRS,CONTENTION – displays resource contention F CATALOG,TAKEDUMP – Takes a dump of the Catalog address space F CATALOG,RESTART – Restarts the catalog address space

Catalog messages in general start with IEC3

Basic Catalog problem determination functions

Catalog provides the following functions to help with problem determination, data collection, and recovery:

- Use the DISPLAY GRS,CONTENTION command to display resource contention.
- Use the MODIFY CATALOG,LIST command to show the active tasks in the catalog address space.
- Use the MODIFY CATALOG,RESTART command to restart the catalog address space.
- Use the MODIFY CATALOG,TAKEDUMP command to take a dump of the Catalog address space.
- Make sure that all catalog checks are activated and running to warn you of impending catalog problems. For details, see Catalog checks (IBMCATALOG) in *IBM Health Checker for z/OS User's Guide*.

Catalog component-specific problems and recovery

- "Hang in the Catalog address space or in the user address waiting on a request to the Catalog address space"
- "Damaged or broken catalogs" on page 240
- "Slow performance in various address spaces due to requests to the catalog address space taking excessive time" on page 241

Hang in the Catalog address space or in the user address waiting on a request to the Catalog address space

Symptoms

- Jobs hung in open or close processing
- TSO users hang when doing a simple ISPF 3.4 data set list
- Specific requests for a data set hang
- ISV monitoring products may indicate a problem accessing catalogs

How to investigate

1. Issue the command F CATALOG,LIST, wait 30 seconds to a minute, and reissue the command. The list of active tasks should have changed. Any tasks waiting over 5 seconds should be investigated. The output from the command will give the requesting address space name, type of request, how long the request has been waiting and in some cases the reason the request is waiting.

2. If any of the requests indicate waiting for an enqueue or reserve or latch issue the command D GRS,C . The resources that are likely to be in contention are SYSIGGV2, SYSZVVDS, SYSVTOC, SYS.IGG.CATALOGLATCHSET. If any of these resources are in contention you should obtain a dump of the catalog address space on each system that could be sharing catalogs. Issuing the command F CATALOG,TAKEDUMP and routing that to all systems in the sysplex will enable in depth problem determination. If there are systems outside the scope of a sysplex that may share some catalogs, data sets or volumes, then dumps should be obtained from those systems also.

Recovery actions

1. Issue the command F CATALOG,RESTART on the systems that appear to be hung.
2. F CATALOG,RESTART will be followed by an IEC363D if the F CATALOG,TAKEDUMP was not used to obtain a dump prior, answer 'y' to IEC363D and 'n' to the following IEC364D query to obtain a dump. Otherwise, reply 'n' to IEC363D to proceed with the restart. For more information, see "Restarting the Catalog Address Space" in *DFSMS Managing Catalogs* .
3. If a restart of catalog is not desirable, use instructions listed in "Ending a Catalog Request Task" in *DFSMS Managing Catalogs* to specifically end a service task believed to be hanging while holding a resource. Ensure that F CATALOG,TAKEDUMP is issued prior to taking action to obtain a dump

Actions to avoid recurrence

Issue the D GRS,C on a periodic basis (perhaps 15 minute intervals). Concern yourself with any of the resources listed above if the queuing increases and the same TCBs are holding resources.

Damaged or broken catalogs

Symptoms

- Jobs hung in open or close processing or other hang symptoms
- Unexplainable catalog errors during locate errors

How to investigate

1. Run the following commands in an IDCAMS batch job to identify errors:

```
//STEP1    EXEC  PGM=IDCAMS,REGION=0M
//SYSPRINT DD    SYSOUT=A //SYSIN    DD *
DIAGNOSE ICFCATALOG INDATASET(catalog.name
EXAMINE NAME(catalog.name) ITEST NODTEST
EXAMINE NAME(catalog.name) NOITEST DTEST
LISTCAT ENT(catalog.name) CAT(catalog.name) ALL
LISTCAT CAT(catalog.name) ENTRIES
```

2. If DIAGNOSE or EXAMINE indicate a condition code 8 or greater, run the following DSS PRINT jobs to capture a picture of the catalog:

```
//STEP2    EXEC PGM=ADRDSSU,REGION=0M
//SYSPRINT DD   SYSOUT=*
//SYSUDUMP DD   SYSOUT=*
//SYSIN    DD *
   PRINT DATASET('catalog.name') -
     INDYNAM(volser)
   PRINT DATASET('catalog.name.CATINDEX') -
     INDYNAM(volser)
```

Recovery actions

Catalog breakages can range from minor logical DIAGNOSE errors to major structural EXAMINE errors.

- For only DIAGNOSE errors, follow recovery actions listed under IDC21364I in *MVS System Messages, Vol 6*.
- If errors include EXAMINE errors, attempt the following actions:
  1. Determine if a cc=0 backup can be taken by using IDCAMS EXPORT. Ensure the number of records exported resembles a number close to the number of records you expect in the catalog.
     ```
     //STEP1    EXEC  PGM=IDCAMS,REGION=0M
     //CATBKUP  DD DSNAME=CAT.BACKUP,UNIT=SYSDA,
     //            DISP=(NEW,CATLG),SPACE=(CYL,(10,10))
     //SYSPRINT DD    SYSOUT=A
     //SYSIN    DD *
        EXPORT catalog.name OUTFILE(CATBKUP) TEMPORARY
     ```
  2. If EXPORT runs fine, refer to instructions listed in Chapter 6 "Backing Up and Recovering Catalogs" in *DFSMS Managing Catalogs* section "Recovering a BCS" to IMPORT this back-up, which will rebuild the catalog index.
  3. If EXPORT fails, recovery from back-up will be necessary. With IBM products, this can be performed by locating the last good back-up taken of the catalog and then forward recovering the back-up to the current time. This will require a back-up produced by EXPORT, SMF 61,65, and 66 records, and use of the IBM supplied forward recovery utility ICFRU. Refer the chapter "Integrated Catalog Forward Recovery Utility" in *DFSMS Managing Catalogs* for instruction in using ICFRU.
  4. If EXPORT fails and a good catalog back-up cannot be located, salvaging entries will be necessary. Refer to "Merging Catalogs" in *DFSMS Managing Catalogs* for direction on how to use REPRO MERGECAT to move entries from the broken catalog to a new catalog.

Actions to avoid recurrence

To help avoid breakages, ensure that catalogs being shared between systems reside on DASD generated as shared and that the catalog is defined with share options (3 4). If the catalog is accessed from outside of the sysplex, ensure sharing is performed with RESERVEs rather than through a convert RNL for SYSIGGV2.

To ensure catalog recovery can be performed in a timely manner, take regular back-ups via IDCAMS EXPORT. Ensure DIAGNOSE and EXAMINE are ran prior to each EXPORT to ensure a valid back-up is obtained. If back-ups are taken with another product like DSS, an additional step of restoring the back-up and then exporting that copy will be required in order to use the ICFRU product since that product only accepts EXPORT output for input.

Slow performance in various address spaces due to requests to the catalog address space taking excessive time

Symptoms

Timeout events such as Abend522 or application specific errors.

How to investigate

1. Issue the command F CATALOG,LIST wait 30 seconds to a minute and reissue the command. The list of active tasks should have changed. Any tasks waiting over 5 seconds should be investigated. The output from the command will give the requesting address space name, type of request, how long the request has been waiting and in some cases the reason the request is waiting.
2. Issue D GRS,C to identify any contention on a particular catalog (SYSIGGV2) or volume (SYSZVVDS)
3. Issue F CATALOG,TAKEDUMP during time of slow performance to obtain a dump in case support must be engaged

Recovery actions

1. Cancel non-critical jobs that are involved in the contention
2. F CATALOG,RESTART. This will be followed by an IEC363D if the F CATALOG,TAKEDUMP was not used to obtain a dump prior, answer ‘y’ to IEC363D and ‘n’ to the following IEC364D query to obtain a dump. Otherwise, reply ‘n’ to IEC363D to proceed with the restart. For more information, see "Restarting the Catalog Address Space" in *DFSMS Managing Catalogs*.

Actions to avoid recurrence

1. Review "Diagnosing a Catalog Performance Problem" in *DFSMS Managing Catalogs* for possible solutions and best practices
2. Investigate any jobs or users that may be issuing generic requests such as locates for *. These processes may hold a shared SYSIGGV2 ENQ for a longer time than typical locate requests.
3. If performance is concerning a particular catalog, research whether certain aliases directed to that catalog can be split off into a separate catalog as this may reduce contention during peak periods
4. If performance is concerning a particular volume, research whether there are other volume processing is occurring at that time such as defragmentation jobs or back-up processing.

Chapter 17. PDSE operational problem determination

Partitioned Data Set Extended (PDSE) is an advanced implementation of the MVS partitioned (PO) data set organization. PDSE files:

- Have a directory structure which does not require pre-allocation and which will dynamically expand to fit the number of members stored within.
- Store members in a manner where periodic data set compression is not needed to reclaim "dead" space.
- Automatically reclaim space when a member is deleted or replaced.

PDSE can be used in place of a PDS to store data, or to store executable programs in the form of program objects. PDSE data sets are processed using most of the same access methods and macros as PDS files.

PDSE data set access is accomplished by means of a client server relationship between the PDSE user (batch jobs, started tasks, or TSO users) and the SMSPDSE and, optionally, the SMSPDSE1 address space.

Basic PDSE problem determination functions

PDSE provides the following functions to assist with first failure data capture and problem determination.

- System generated ABEND0F4 dumps
- EREP reports
- SYSLOG messages
- The SMSPDSE and the SMSPDSE1 monitor task
- PDSE system commands
- The IEBPDSE PDSE Verification Utility program.

Collecting documentation for PDSE

The support center routinely requests a standard set of diagnostic data for PDSE issues, included is:

- The first system generated ABEND0F4 dump.
- Detailed EREP report or raw LOGREC data that brackets the dump by two hours. Raw LOGREC is the preferred format.
- SYSLOG or OPERLOG for the same time frame as LOGREC/EREP.
- In the event of PDSE data set corruption, an ADRDSSU physical dump of the affected PDSE data set.

In event that a system generated dump was not taken or if the dump was DAE suppressed then an operator initiated dump of the PDSE address space or spaces may be required.

Pre-allocating an IEADMCxx PARMLIB member, to dump the PDSE processing environment, will save time. Create the member as follows:

```
JOBNAME=(*MASTER*,SMSPDSE*),
SDATA=(PSA,CSA,SQA,GRSQ,LPA,LSQA,RGN,SUM,SWA,TRT,COUPLE,XESDATA)
```

Issue the following command when you need to obtain a dump:

```
DUMP COMM=(title),PARMLIB=xx
```

Where, ‘xx’, is the suffix of the IEADMCxx parmlib member that contains the SMSPDSE dump options.

PDSE specific problems

Common PDSE issues that might require Support Center assistance:

- “ABEND0F4 failures”
- “MSGIGW038A possible PDSE problems” on page 245
- “PDSE data set corruption” on page 247
- “Failure of the SMSPDSE or SMSPDSE1 address space” on page 249

ABEND0F4 failures

PDSE code performs extensive validity checking and will issue a system 0F4 abnormal termination (ABEND0F4) completion code for internally detected errors. These ABEND0F4 are accompanied by unique return and reason codes that identify the module where the failure occurred and the reason for the abnormal termination.

By default the ABEND0F4 dump will include the primary, home and secondary address spaces insuring that both the SMSPDSE/SMSPDSE1 server and client address spaces are captured in the dump.

If more than one dump is generated then the first dump in the series is usually the most pertinent to the cause of the failure, as subsequent dumps may be recording issues that were encountered during recovery processing.

Note: The first dump in a series can be something other than a S0F4 ABEND, such as ABEND0C1, ABEND0C4. or ABEND30D.

Symptoms

ABEND0F4 failures in batch jobs, TSO users, or started task address spaces.

How to investigate

- Note the DUMP TITLE from the system IEA794I (msgIEA794I) message and record the following data:
  - The failing CSECT name
  - Offset within the CSECT
  - Maintenance level
  - Return code
  - Reason code.

    Note: The reason code is recorded in general purpose register 0 and the return code is recorded in general purpose register 15. Check these two registers if the reason and return codes are not readily apparent.

  In most instances this information is sufficient enough to identify a matching APAR.

  The following is a sample S0F4 DUMP TITLE, for an ABEND0F4 RC24 RSN1467A02B failure out of IGWISRCH+0E92.

```
DUMP TITLE=COMPID=DF115,CSECT=IGWISRCH+0E92,DATE=03/18/11,MAINTID=
NONE,ABND=0F4,RC=00000024,RSN=1467A02B
```

- Search the Technical Support database or IBMLINK for:
  - Available APARs to resolve known issues
  - Suggested recovery actions.

Recovery actions

- Implement the suggested recovery action or actions and schedule the install of all maintenance that was identified in your problem search. Note, that the implementing all PDSE maintenance requires an IPL with CLPA.
- Gather the standard set of PDSE diagnostic data and contact the Support Center if additional assistance is required.

Actions to avoid recurrence

MSGIGW038A possible PDSE problems

PDSE employs an optional monitor that checks PDSE resources on regular intervals for possible problems. The monitor reports potential PDSE problems in an IGW038A messages.

Note: The IGW038A message only warns of a possible problem. Further actions must be taken to determine if a problem actually exists.

A PDSE ANALYSIS command should be issued as soon as possible after the receipt of an IGW038A message. The IGW038A message is issued from either the SMSPDSE or SMSPDSE1 address space. The issuing address space is identified in the first line of the message:

```
IGW038A POSSIBLE PDSE PROBLEM(S). (SMSPDSE|SMSPDSE1)
RECOMMEND ISSUING V SMS,PDSE,ANALYSIS
```

The recommended action is to issue a PDSE ANALYSIS command. IBM strongly suggests that clients employ their system automation package to respond to IGW038A messages.

Symptoms

IGW038A messages

How to investigate

1. The target address space for the PDSE ANALYSIS command depends on the address space that issued the IGW038A message:
   - ANALYSIS command for IGW038A issued from the SMSPDSE address space:
     ```
     VARY SMS,PDSE,ANALYSIS
     ```
   - ANALYSIS command for IGW038A issued from the SMSPDSE1 address space:
     ```
     VARY SMS,PDSE1,ANALYSIS
     ```
     Route the command to all LPARs in the SYSPLEX.
2. The VARY ANALYSIS command output is displayed in an IGW031I message. This message identifies any latch/lock contention problems, latch/lock hangs, and outstanding cross system messages. (PDSE uses XCF messaging for serialization between systems in a SYSPLEX.) Refer to the manual *z/OS*

DFSMSdfp Diagnosis, Chapter "PDSE Diagnostic Aids", section "Analysis Command", subheading "Recommended Usage", for a description of the IGW031I command output.

Note that a single IGW038A message is not an indication of a problem, especially if the IGW031I message states:

```
++ No exceptional data set conditions detected
```

In all likelihood, this message indicates that the detected exception condition was temporary in nature. Proceed to the next step if IGW038A messages continue to occur.

3. Parse the IGW031I message for an indication of cross system issues. Make note of any remote subsystem IDs (ssssssss)

   ```
   ++ Message to ssssssss pending for nnnnnnnn seconds
   ```

4. Obtain a dump of the identified PDSE operating environment. The recommended method for obtaining a dump, in this situation, is to use the Monitor's DUMPNEXT processing option. This dump will automatically include both the SMSPDSE or SMSPDSE1 server and the client address space.

   The DUMPNEXT processing option should be enabled on the LPAR where the IGW038A message was issued and any remote systems identified in the IGW031I message.

   Issue the following command to enabling DUMPNEXT processing for the SMSPDSE address space:

   ```
   VARY SMS,PDSE,DUMPNEXT
   ```

   Issue the following command to enabling DUMPNEXT processing for the SMSPDSE1 address space:

   ```
   VARY SMS,PDSE1,DUMPNEXT
   ```

Recovery actions

1. After obtaining the dump or dumps CANCEL or CANCEL and FORCE, if necessary, any TSO users or jobs identified in the IGW031I message. In most instances the termination of the address space that is holding the contested PDSE resource will clear the latch.

   Note: In most instances, ASRBULCH latches are secondary issues and not the primarily contested PDSE resource.

2. Reissue the ANALYSIS command after terminating job or jobs. If the IGW031I shows the same contested latch then the FREELATCH command should be issued. Again the format of the FREELATCH command is dependent on the address space where the resource issue was detected.

   ```
   V SMS,PDSE(1),FREELATCH(latchaddr,ASID,tcbaddr)
   ```

   where latchaddr, ASID, and tcbaddr are from the IGW031I output.

   Note: The FREELATCH command can only be used to free latches. This command will have no effect against PDSE data set locks.

   Important: If this command is used to release a latch held by a process (on the same system or on another system in a multisystem environment) that is still running, it could result in breaking the PDSE and it could cause system abends.

3. Run the ANALYSIS command again after issuing the FREELATCH command. If problems still exist, there is a good possibility that an SMSPDSE1 address space restart or an IPL will be required to clear the problem.

   Consideration should be given to engaging the service center at this point.

Important: All support releases of PDSE have the option of running with a re-startable SMSPDSE1 address space. A restart of the SMSPDSE1 address space should be attempted prior to an IPL.

Refer to the manual *DFSMSdfp Diagnosis Reference*, chapter "PDSE Diagnostics Aids", for a discussion of consideration for restarting the SMSPDSE1 address space.

Issue the following command if it been determined that the PDSE1 address can and should be restarted:

```
V SMS,PDSE1,RESTART
```

4. An IPL will be required if:
   - The restart of the SMSPDSE1 fails to free the contested PDSE resource
   - If the SMSPDSE1 address space is not activated.

   Consideration should be given to engaging the support center before an IPL is attempted as the support center may be able to offer other recovery options.
5. Open an incident with the support center, COMPID 5695DF115, after all recovery actions have been completed or if attempted recovery actions fail to resolve the PDSE resource contention issue. Submit the following diagnostic data to the support center:
   - All dumps that were captured
   - Detailed EREP report or raw LOGREC data that brackets the dumps by two hours. Raw LOGREC data is preferred format.
   - SYSLOG or OPERLOG for the same time frame as LOGREC/EREP.

Actions to avoid recurrence

PDSE data set corruption

Symptoms

PDSE corruption is normally discovered through an operator message or an ABEND0F4 failure.

Common operator messages which indicate possible PDSE data set corruption include:
- IEC036I 002-A4 IGC0005E
- IEC143I 213-50
- IGW01330T DIAG 271B0409, 281C014A, 270C03F7,
- CSV031I LIBRARY SEARCH FAILED FOR MODULE xxxxxxx, 27080409
- 'I/O error reading data' when using ISPF Browse

Common ABEND0F4 reason codes that may indicate PDSE corruption are:

Reason Code	Module	Explanation
010E5AB8	IGWDARD1	
IGWDADC0	JCDM_NOT_ALL_SARS_RETURNED	
050A0046	IGWBEXT2	BDS_INVALID_DSCB_EXTENTS
06215910	IGWBVLP1	JCDM_BAD_VDF
070A0021	IGWBVLP1	CDS_INVALID_LSCB
130B0203	IGWLH3AB	CLM_FIND_HL3B_HASH_FAILED
1451A030	IGWDDCR2	IMF_DUPLICATE
145AA033	IGWISRCH	IMF_INDEX_PAGE_DAMAGED

Reason Code	Module	Explanation
1465A03E	IGWISUBS	IMF_ABEND_DURING_PROC
150A001E	IGWBITX1	INVALID_DIRECTORY_SUFFIX
150BC008	IGWBIEX1	RSNS_IO_ERROR_DIRECTORY
150BC009	IGWBIEX1	RSNS_IO_ERROR_MEMBER
1C0752EE	IGWDRRRC	JCDM_RECORD_PREFIX_ERROR
270C03F7	IGWCDGTA	DESRS_IEWBXILO_ERROR
271B0409	IGWCDGTL	DESRS_UNEXPECTED_JCDM_ERROR

PDSE corruption can be caused by improper sharing or serialization of the PDSE resources in the user's processing environment. Most PDSE sharing problems violate one of the following rules:

- GRS must be active on any system that shares PDSEs with another system.
- In a PDSESHARING(NORMAL) environment all sharing PDSE must be in the same GRSPLEX.
- In PDSESHARING(EXTENDED) environment all sharing systems PDSE must be in the same SYSPLEX. (Extended PDSE sharing requires XCF signaling between the sharing systems and XCF signaling is restricted to systems within a single SYSPLEX.)

How to investigate

Identify the problem PDSE data set.

1. The data set name will appear in the following operator messages:
   - IEC036I
   - IEC143I
   - CSV031I – will contain the failing DDNAME

   Proceed to the next step if PDSE data set name cannot be identified from an operator message.
2. Format the dump and issue the following command:
   ```
   IPCS STATUS
   ```
   Some of the PDSE subcomponents will record the PDSE data set name in the SDWA Variable Recording Area of this report. Proceed to the next step if the data set name was not recorded in the SDWA Variable Recording Area.
3. A more detailed analysis of the ABEND0F4 dump will be needed if a PDSE data set is not identified by either of the previous two steps. Open an incident with the service center and transfer the ABEND0F4 dump in for assistance in identifying the failing PDSE.

Recovery actions

1. Skip this step if the LPAR where the failure is occurring is a monoplex. Otherwise, attempt an ISPF browse against the identified PDSE data set from another LPAR in the SYSPLEX. Proceed to step 4 if the browse fails. If the ISPF browse from another LPAR in the SYSPLEX succeeds then the DASD copy of the file is intact and the most likely problem is that the cached directory pages for the PDSE data set are out of sync with the DASD copy of the file. Proceed to step 3 if the failing LPAR is not release at z/OS 1.13 or above.
2. 2. z/OS 1.13 and above. Issue the following command to discard the cached directory pages and reread the directory from DASD VARY SMS,PDSE|PDSE1,REFRESH,DSN(dsname)[,VOL(volser)] Attempt to browse

the identified PDSE data set on the failing LPAR after the command completes. Proceed to the next step if the failure still occurs.

3. If implemented, restart the SMSPDSE1 address space.

   ```
   VARY SMS,PDSE1,RESTART
   ```

   Restarting the PDSE address space will rebuild the connections to all opened PDSE data sets. Proceed to the next step if the SMSPDSE1 address space is not implemented or if the file still cannot be browsed after the restart completes.
4. Open an incident with the service center. Gather and transfer in for review the previously defined basic PDSE diagnostic data.
5. The PDSE data set is corrupted, begin data set recovery efforts.
6. IEBCOPY the corrupted data set into a newly allocated PDSE data set. Proceed to the next step if the IEBCOPY job fails.
7. If the IEBCOPY fails, then the data set will have to be recovered from a viable back up. Open an incident with the service center and gather and transfer in for review the previously defined basic PDSE diagnostic data. In addition to the basic PDSE diagnostic data, collect and transfer in the following additional data:
   - A DFDSS physical dump copy of the corrupted PDSE data set.
   - An IEHLIST LISTVTOC FORMAT report or a DFDSS PRINT of the VTOC from the volume where the PDSE resides.
   - SMF TYPE14/15 records from all sharing LPARs back to the last successful access.

Actions to avoid recurrence

Failure of the SMSPDSE or SMSPDSE1 address space

Symptoms

The deactivation of the SMSPDSE is indicated by an IGW007E message.

```
IGW007E SMS HAS BEEN DEACTIVATED DUE TO A FATAL ERROR RETURN CODE (IN
HEX): return-code REASON CODE (IN HEX): reason-code {A DUMP HAS BEEN
TAKEN|NO DUMP HAS BEEN TAKEN}
```

The deactivation of the SMSPDSE1 address space is indicated by an IGW077E message.

```
IGW077E SMSPDSE1 DEACTIVATED DUE TO A FATAL ERROR RETURN CODE:
return-code REASON CODE: reason-code A RESTART OF SMSPDSE1 IS
RECOMMENDED. TO RESTART ENTER: VARY SMS,PDSE1,RESTART
```

External symptoms include, but are not limited to, delays in job initialization and job termination, delays in job processing, and S30D abends.

How to investigate

- The deactivation of the SMSPDSE is indicated by an IGW007E message.
- The deactivation of the SMSPDSE1 is indicated by an IGW077E message.

These messages are issued as a highlighted operator action message that requires immediate attention. Both messages will remain on the console until recovery action is taken.

In the case of the deactivation of the SMSPDSE1 address space PDSE processing will be held or corralled until such point that the SMSPDSE1 address space is restarted. This corralling of PDSE activity will have an adverse affect on LPAR where the error occurred.

Recovery actions

1. Proceed to step 2 if the SMSPDSE address space was deactivated. Continue with this step if the SMSPDSE1 address space was deactivated. Attempt to restart the SMSPDSE1 address space by issuing the following command

   ```
   VARY SMS,PDSE1,RESTART
   ```

   A restart of the SMSPDSE1 address space is almost always successful. Proceed to the next step in the event that the address space fails to restart.
2. Obtain a stand alone dump.
3. IPL the LPAR.
4. Open an incident with the service center and gather and transfer in for review the SADUMP and previously defined basic PDSE diagnostic data.

Actions to avoid recurrence

- Enable the SMSPDSE1 address space when executing in a PDSESHARING(EXTENDED) environment.
- Run with the SMSPDSE and SMSPDSE1 Monitor task enabled.
- Automate the response to the IGW038A message.
- Conform to PDSE sharing rules. See the following URL for a summary of PDSE sharing rules:

  http://publib.boulder.ibm.com/infocenter/ieduasst/stgv1r0/index.jsp?topic=/com.ibm.iea.zos/zos/1.0/DFSMS/V1R0-PDSE-SharingRules/player.html

Chapter 18. RRS operational problem determination

Resource Recovery Services (RRS) helps installations protect resources. RRS is a syncpoint manager that consists of protocols and program interfaces which, when requested, can coordinate consistent changes to one or more protected resources. These resources can be accessed through different resource managers and can be on different systems within a sysplex. For more information, see *z/OS MVS Programming: Resource Recovery*.

Basic RRS problem determination functions

RRS provides the following functions to help with problem determination, data collection, and recovery:

- Use the `DISPLAY RRS` command to query status of resource managers and units of recovery (UR). Units of recovery represent transactions.
- Use CTRACE options specific to SYSRRS to gather detailed information about RRS requests and events.
- Use the SETRRS command to cancel or shut down RRS, and enable and disable archive logs. RRS issues messages with the ATR prefix.
- Make sure that all RRS checks are activated and running to warn you of impending storage usage and other RRS problems. For details, see RRS checks (IBMRRS) in *IBM Health Checker for z/OS User's Guide*.

Collecting documentation for RRS

This section contains the following topics:

- "Dumping RRS information"
- "Important RRS CTRACE information" on page 252

Dumping RRS information

When the IBM Support Center requests an RRS dump, they typically ask for the same data every time. To save time, create two IEADMCxx parmlib members.

Note: If your installation setup RRS with a job name other than 'RRS', replace that value in both the JOBNAME and DSPNAME parameters in the following examples:

- **To dump RRS:** Create an IEADMCxx parmlib member with the requested RRS dump parameters as follows:

  ```
  JOBNAME=(RRS),DSPNAME=('RRS'.*),
  SDATA=(ALLNUC,LPA,LSQA,PSA,RGN,SQA,TRT,CSA,GRSQ,COUPLE)
  ```

  When you need a dump of RRS, use the following command:

  ```
  DUMP COMM=(title),PARMLIB=xx
  ```

  xx is the suffix of the IEADMCxx parmlib member that contains the RRS dump options.
- **To dump RRS and system logger:** Create an IEADMCxx parmlib member with the requested RRS and system logger dump parameters as follows:

  ```
  JOBNAME=(RRS,IXGLOGR),DSPNAME=('RRS'.*,'IXGLOGR'.*),
  SDATA=(ALLNUC,LPA,LSQA,PSA,RGN,SQA,TRT,CSA,GRSQ,COUPLE)
  ```

Then, when you need a dump of RRS and system logger, use the following command:

```
DUMP COMM=(title),PARMLIB=xx
```

xx is the suffix of the IEADMCxx parmlib member that contains the RRS and system logger dump options.

If exploiters are involved, you can include them in the JOBNAME list in your IEADMCxx parmlib members.

Important RRS CTRACE information

RRS strongly suggests that you have the following CTRACE options set at all times. This ensures you have the most trace information available in the RRS dumps. Create a CTIRRSxx parmlib member that contains the following options:

```
TRACEOPTS
  ON
  BUFSIZE(500M)
  OPTIONS('EVENTS(URSERVS,LOGGING,CONTEXT,EXITS,STATECHG,RRSAPI,RESTART)')
```

To ensure that these CTRACE settings are always active:

- Issue the `S RRS` command, specifying the CTIRRSxx parmlib member.
- Update the RRS procedure with the CTMEM value to ensure that RRS CTRACE is running on each startup.

If RRS is already started, to set these options, issue the following z/OS command:

```
TRACE CT,ON,COMP=SYSRRS,PARM=CTIRRSxx
```

RRS CTRACE in a sysplex cascaded transaction environment and problem determination mode: If you are running a sysplex cascaded transaction environment and are in problem determination mode, use the external writer to trace more data. In addition, update the RRS trace to use the following option:

```
OPTIONS('EVENTS(ALL)')
```

You are in problem determination mode when you are collecting documentation for a recurring problem, such has a transaction hang or delay.

RRS recovery options

This section contains the following topics:

- "RRS warm start"
- "RRS cold start" on page 253

RRS warm start

An RRS warm start is also named recycling RRS. For an RRS warm start:

1. Whenever doing a warm start of RRS, first shut down its exploiters whenever possible, as documented in*z/OS MVS Programming: Resource Recovery*. For example, WebSphere® does not handle RRS termination and terminates itself when it detects RRS is not available. Additional exploiters might encounter abends when RRS terminates if they had outstanding processing activity at that time. IMS can terminate for some abend conditions.
2. Terminate RRS with one of the following commands:
   - `SETRRS CANCEL`, which terminates RRS with an abend code X'222'.
   - `SETRRS SHUTDOWN`, which first stops each resource manager and then terminates RRS without an abend (normal termination).

3. Restart RRS with the Automatic Restart Manager (ARM) or by issuing the `START RRS` command.

RRS cold start

A cold start of RRS requires deletion and redefinition of the RRS RM.DATA log stream with the IXCMIAPU utility. You can find sample JCL for an RRS cold start in the ATRCOLD member of SYS1.SAMPLIB.

When unrecoverable corruption is detected in either the RRS RM.DATA or RESTART log streams, process an RRS cold start. For instructions about RRS cold start processing, see Cold start in *z/OS MVS Programming: Resource Recovery*.

RRS component-specific problems and recovery

- "RRS resource contention"
- "RRS suspended, waiting for signal from system logger" on page 254
- "RRS log stream gap condition" on page 255
- "RRS log stream data loss condition" on page 256
- "RRS high processor usage" on page 257
- "RRS address space hang" on page 257
- "RRS high storage usage" on page 257
- "Resource manager is unable to start with RRS" on page 259
- "Resource manager termination delay" on page 259
- "RRS transaction hang" on page 260
- "RRS severe error on RRS RM.DATA log stream, message ATR250E" on page 261
- "Resolving RRS problems in a sysplex cascaded transaction environment" on page 262
  - "Sysplex cascaded transaction hang" on page 263
  - "Sysplex cascaded transaction hang messages ATR246I and ATR247E" on page 264

RRS resource contention

Symptoms

RRS resource contention shows up in DISPLAY GRS command output as contention for resource SYSZATR *sysplex_name*-RESTART. RRS obtains this global resource when manipulating the ATR.l*ogging_group_name*.RM.DATA log stream. Operations such as RRS start/stop, resource manager start/unset, and internal log stream management obtain this resource exclusively. If any of these operations hang while holding this resource, all other operations in the sysplex that require this resource also hang.

How to investigate

Look for these symptoms:

- Message ATR248E or ATR249E to identify a potential hang in log stream processing. See "RRS suspended, waiting for signal from system logger" on page 254.
- ATR* and IXG* messages that involve the resource manager data log stream.

Recovery actions

Periodically monitor for SYSZATR RESTART resource contention with the DISPLAY GRS,CONTENTION command. If you detect contention, Follow the action specified in the system logger service return and reason codes shown in message ATR248E or ATR249E. See the IXGWRITE return and reason codes in *z/OS MVS Programming: Authorized Assembler Services Reference EDT-IXG*.

If the action in the return and reason code does not resolve the contention, or if you cannot detect the cause of the contention:

1. Capture a dump of RRS, system logger, and GRS on the system that is holding the resource.
2. Recycle RRS on the holding system to free the resource.

Note: If the issue that causes the holder to hang persists, the resource contention can continue until the issue is resolved.
Search problem reporting databases for a fix for the problem. If no fix exists, report the problem to the IBM Support Center and provide the dump, system logs, and formatted LOGREC data set. (A symptom record is created on the system experiencing the failure and stored in the LOGREC data set.)

Actions to avoid recurrence

Periodically monitor for SYSZATR RESTART resource contention with DISPLAY GRS,CONTENTION.

RRS suspended, waiting for signal from system logger

RRS uses system logger services to manage resource manager and unit of recovery (UR) status information. Sometimes RRS ends up suspended while waiting for a signal from system logger.

Symptoms

The system issues one of the following messages when an RRS function is suspended in Logger processing:

-
  ```
  ATR248E RRS IS WAITING FOR SIGNAL FROM LOGGER TO RESUME PROCESSING
                   RETURN CODE: returncode REASON CODE: reasoncode DIAGNOSTIC
                   INFORMATION: diag1 diag2 diag3 diag4
  ```

-
  ```
  ATR249E RRS IS WAITING FOR SIGNAL FROM LOGGER TO RESUME PROCESSING
              LOGSTREAM NAME: logstreamname RETURN: returncode REASON: reasoncode
              DIAGNOSTIC INFORMATION: diag1 diag2 diag3 diag4
  ```

The messages indicate that RRS has a suspended TCB waiting for a signal from system logger before it can resume processing. The TCB is suspended for at least 30 seconds. The problem occurs when RRS is in the process of updating an RRS log stream and encounters a temporary error condition. The error condition is identified by the return and reason codes from the IXGWRITE service shown in the messages.

Generally, RRS is holding the SYSZATR RESTART global resource when suspended. The system obtains this resource when RRS updates the ATR.logging_group_name.RM.DATA log stream, which it can do during:

- Internal RRS housekeeping
- Restart or unset processing

All processing that involves the RM.DATA log stream throughout the RRS logging group (generally the sysplex) is suspended while waiting for the signal from system logger.

How to investigate

To investigate:

- Review the meaning of the return and reason codes found in message ATR248E or ATR249E. , which are found in the topic about IXGWRITE - Write log data to a log stream in *z/OS MVS Programming: Authorized Assembler Services Reference EDT-IXG*.
- Review SYSLOG for prior messages that show an issue with system logger processing. For example, look for IXG messages that involve the RM.DATA log stream.

Recovery actions

Check for and correct problems with system logger by reviewing the reported diagnostic information. If the problem cannot be identified or resolved, capture a dump of RRS and Logger to provide to the IBM Support Center. Provide the dump along with SYSLOG and formatted LOGRED data set.

If additional RRS work is suspended, as indicated by SYSZATR RESTART global resource contention, you might need to recycle RRS. Recycle on the system that is receiving the ATR248E message or ATR249E message. When the temporary error condition reported by system logger remains persistent, the problem reappears until you resolve the log stream issue.

RRS log stream gap condition

RRS uses system logger to manage resource manager and UR status information. RRS encountered a gap condition when browsing the log stream, which can lead to a loss of data.

Symptoms

The following message is issued when the system detects a log stream gap condition.

```
ATR202D GAP FOUND IN logstreamname. REPLY RETRY TO RETRY OR ACCEPT TO
        ACCEPT THE DATA LOSS
```

| When the gap condition is against the RRS RM.DATA log stream, the following
| message may also be issued.

```
| ATR212I RRS DETECTED LOG DATA LOSS ON LOGSTREAM logstreamname
|         DUE TO INACCESSIBLE LOG DATA. LOG DATA FROM lowGMT TO
|         highGMT ARE AFFECTED.
```

See "RRS severe error on RRS RM.DATA log stream, message ATR250E" on page 261 for more information.

How to investigate

To investigate:

- Review SYSLOG for prior messages that report the gap condition.
- Sometimes, when an HSM backup occurs against the volume that contains the RRS log stream data set, it prevents system logger from opening the data set. This condition can result in a gap condition for the log stream. To determine whether the gap condition is the cause, look for the following accompanying messages:

```
ARC0722I BACKUP STARTING ON VOLUME vvvvv(SMS) AT
ARC0722I (CONT.) hh:mm:ss ON yyyy/mm/dd SYSTEM ssss

IEC161I 052(015)-084,jjj,sss,,,
IEC161I hlq.ATR.logging_group_name.logstream.Axxxxxxx,
IEC161I hlq.ATR.logging_group_name.logstream.Axxxxxxx,.DATA,ICFCAT.SYSCAT03

IXG268I LOGSTREAM DATASET hlq.ATR.logging_group_name.logstream.Axxxxxxx
```

```
                  CAN NOT BE OPENED FOR JOB RRS DUE TO
                  INCORRECT VSAM SHAREOPTIONS OR OTHER ERROR,
                  REQUESTED DATA MAY NOT BE AVAILABLE.

ATR202D GAP FOUND IN ATR.logging_group_name.logstream.
                    REPLY RETRY OR ACCEPT TO ACCEPT THE DATA LOSS

ARC0723I BACKUP ENDING ON VOLUME vvvvv AT hh:mm:ss
```

Recovery actions

- If you can ascertain the cause of the gap, correct the problem and reply RETRY to the message. If the gap is caused by HSM backup activity, reply RETRY to message ATR202D after you get accompanying message ARC0723I.

  Note: Replying ACCEPT to message ATR202D hardens the gap as a data loss to RRS. Internal flags are set that cannot be updated without a warm start.

  When the gap condition is against the RRS RM.DATA or RESTART log stream and:
  1. **Can** be corrected, use an RRS warm start to reset the internal flag settings.
  2. **Cannot** be corrected, use an RRS cold start to recover.
- If you cannot find the cause of the gap condition, capture a dump of RRS and system logger to provide to the IBM Support Center. Provide the dump along with SYSLOG and formatted LOGREC data.

RRS log stream data loss condition

RRS uses system logger to manage resource manager and unit of recovery (UR) status information.

Symptoms

The system issues the ATR210E message when it detects a log stream data loss condition when RRS attempted to browse its RM.DATA log stream.

```
ATR210E INACCESSIBLE LOG DATA DETECTED ON THE RRS RM DATA LOGSTREAM logstreamname
```

The system can also issue accompanying message ATR218I to describe an RRS process that failed because of the inaccessible log data.

| When the data loss condition is against the RRS RM.DATA log stream, the
| following message may also be issued.

```
| ATR212I RRS DETECTED LOG DATA LOSS ON LOGSTREAM logstreamname
|         DUE TO INACCESSIBLE LOG DATA. LOG DATA FROM lowGMT TO
|         highGMT ARE AFFECTED.
```

See “RRS severe error on RRS RM.DATA log stream, message ATR250E” on page 261 for more information.

How to investigate

Review SYSLOG for prior messages from system logger that reports the data loss condition.

Recovery actions

After recovering the log stream data, correct the data loss condition and recycle RRS. When data loss occurs against the RRS RM.DATA or RESTART log stream, and cannot be corrected, use an RRS cold start to recover.

If you cannot determine the cause of the data loss condition, capture a dump of RRS and system logger to provide to the IBM Support Center. Provide the dump along with SYSLOG and formatted LOGREC data.

Actions to avoid recurrence

Use duplexing for both the RM.DATA and RESTART log streams to protect against data loss. For additional details, see APAR II12276.

RRS high processor usage

Symptoms

You detect High processor usage for RRS.

Recovery actions

High processor usage detected for RRS processing suggests a loop. Collect a dump of RRS. If this condition is a sysplex cascaded transaction processing environment, collect dumps on each system in that environment. Recycle RRS on the system with high processor usages. Search the problem reporting data bases for a fix for the problem. If no fix exists, report the problem to the IBM Support Center, providing the dump or dumps along with SYSLOG and formatted LOGREC data.

RRS address space hang

Symptoms

RRS is not processing work.

How to investigate

Look for exception messages for RRS check RRS_Storage_NumServerReqs. For information about the check, see RRS_Storage_NumServerReqs in *IBM Health Checker for z/OS User's Guide*. This check monitors the number of server requests within RRS. The system issues exception message ATRH016E if the number of server requests exceeds the threshold monitored:

```
ATRH016E The current number of server task requests in RRS is
       curreqs which exceeds the threshold
```

If this number continues to grow, RRS might have a hang. RRS has a limited number of TCBs to process each server request.

You can also observe transaction hangs by looking at the output of the following DISPLAY command:

```
D RRS,UREX
```

Recovery actions

If RRS_Storage_NumServerReqs check exception message ATRH016E shows the number of server requests growing, check for persistent RRS resource contention. Capture a dump of RRS and the holder of RRS resources, if applicable, and provide them to IBM Support Center. Recycle RRS to address the suspected hang processing server requests.

RRS high storage usage

Symptoms: High storage usage detected in the RRS address space.

How to investigate: Look for RRS IBM Health Checker for z/OS check exception messages that warn of growth in RRS storage use. See RRS checks (IBMRRS) in *IBM Health Checker for z/OS User's Guide* for a description of RRS checks:

- RRS_Storage_NumLargeLOGBlks - Monitors the count of large log buffer blocks in RRS. The system issues exception message ATRH020E if the count exceeds the threshold monitored, which can affect RRS private storage. (The default threshold is 1,000.)
- RRS_Storage_NumLargeMSGBlks - Monitors the count of large message buffer blocks in RRS. The system issues exception message ATRH018E if the count exceeds the threshold monitored, which can affect RRS private storage. (The default threshold is 1,000.)
- RRS_Storage_NumTransBlks - Monitors the count of active URs (transactions) with RRS. The system issues exception message ATRH014E if the count exceeds the threshold monitored, which can impact common storage usage. (The default threshold is 10,000.) In addition to the UR control block created in private storage, RRS creates additional control block data in common storage.

How to investigate

Look for RRS IBM Health Checker for z/OS check exception messages that warn of growth in RRS storage use. See RRS checks (IBMRRS) in *IBM Health Checker for z/OS User's Guide* for a description of RRS checks:

- RRS_Storage_NumLargeLOGBlks - Monitors the count of large log buffer blocks in RRS. The system issues exception message ATRH020E if the count exceeds the threshold monitored, which can affect RRS private storage. (The default threshold is 1,000.)
- RRS_Storage_NumLargeMSGBlks - Monitors the count of large message buffer blocks in RRS. The system issues exception message ATRH018E if the count exceeds the threshold monitored, which can affect RRS private storage. (The default threshold is 1,000.)
- RRS_Storage_NumTransBlks - Monitors the count of active URs (transactions) with RRS. The system issues exception message ATRH014E if the count exceeds the threshold monitored, which can impact common storage usage. (The default threshold is 10,000.) In addition to the UR control block created in private storage, RRS creates additional control block data in common storage.

Recovery actions

If the amount of storage that is being monitored continues to grow, determine whether it is expected behavior. Use the available RRS data collection techniques such as panels, console display command, or batch program. These techniques can help you assess the level of transaction activity in RRS and determine if it is unusual or unexpected. The threshold values can be updated for each IBM Health Checker for z/OS check to values more appropriate to the environment. Follow the action based on:

- RRS_Storage_NumLargeMSGBlks exception messages:
  - Look for hangs with sysplex cascaded transactions and base actions on the hang reported
  - Use the DISPLAY RRS,UREX command to list all active units of recovery (UR) waiting for an event.
  - If possible, resolve the hang. Otherwise, capture a dump of RRS and the associated resource manager to provide to the IBM Support Center for further assistance.
- RRS_Storage_NumTransBlks exception messages:
  - Enter the `DISPLAY RRS,UR` command to list all active UR.
  - Investigate which application is creating these URs and check for a problem with that application, such as looping, not cleaning up transactions, or recursive errors.

  – If you do not understand the increase in transaction activity:
    - Collect a dump of RRS and any resource manager or work manager identified as the source of the increased transaction activity.
    - Provide the dump to the IBM Support Center for further assistance.
- For any high storage count:
  – Look for the RRS_Storage_NumServerReqs check exception message, ATRH016E. The message suggests a possible hang in RRS processing, which can cause storage usage to grow.
  – Capture a dump of RRS and provide it to the IBM Support Center for further assistance.

If you cannot find the cause of storage growth, collect a dump of RRS and any involved exploiter for analysis. Provide the dump along with SYSLOG and formatted LOGREC data set to the IBM Support Center. If the growth continues, recycle RRS to clean up the storage usage after collecting documentation to avoid reaching a critical storage shortage.

Actions to avoid recurrence

Make sure that all RRS checks are activated and running to warn you of impending storage usage and other RRS problems.

Resource manager is unable to start with RRS

Symptoms

In a known example of this problem, when DB2 fails to restart with RRS, it issues one of the following messages:

-
  ```
  DSNX982I db2regionWLM attempt to perform RRS attach function failed with
                 RRS RC8 RSN 00F30091
  ```

-
  ```
  DSN3031I *DBVC DSN3RRSI RRS ATTACH INITIALIZATION FAILED, RRS CALL=CRGSEIF, RRS RETURN CODE=X'00008004'
  *DSNV086E *DBVC DB2 ABNORMAL TERMINATION REASON=00F30095
  ```

How to investigate

Issue the `DISPLAY RRS,RM` command to determine the state of the resource manager.

Recovery actions

Use one of the following recovery actions:

- When the resource manager is in SET state, use the RRS ISPF panels to “Unregister RM”. You can enter “n” next to the RM in SET state in the resource manager list. You can also use the ATRSRV utility with REQUEST(UNREGRM) to unregister the specific RMNAME. This state might be observed when the DSN* DB2 messages are observed.
- When the resource manager is in UNSETINPROG state, check for RRS global resource contention. If observed, collect a dump of the holding address space, RRS, system logger, and GRS. Provide the dump along with SYSLOG and formatted LOGREC data to the IBM Support Center. Recycle RRS on the holding system.
- When the resource manager is in RUN or UNSET state, work with the exploiter to determine why it is failing to restart. Also determine if it involves RRS.

Resource manager termination delay

Symptoms

Resource manager termination delay can be indicated by the following WTOR messages:

- These messages can delay the termination of resource manager until they receive a reply.

```
ATR225D   CANCEL DELAYED. REPLY WAIT, BACKOUT, OR COMMIT TO RESOLVE
                   INDOUBT UR. URID=uridentifier
ATR226D   MEMTERM DELAYED. REPLY WAIT, BACKOUT, OR COMMIT TO RESOLVE
                   INDOUBT UR. URID=uridentifier
ATR227D   CANCEL DELAYED. REPLY WAIT, BACKOUT, OR COMMIT TO RESOLVE
                   INDOUBT UR. URID=uridentifier
ATR228D   MEMTERM DELAYED. REPLY WAIT, BACKOUT, OR COMMIT TO RESOLVE
                   INDOUBT UR. URID=uridentifier
ATR229D   CANCEL DELAYED. REPLY WAIT, BACKOUT, OR COMMIT TO RESOLVE
                   INDOUBT UR. URID=uridentifier
ATR230D   MEMTERM DELAYED. REPLY WAIT, BACKOUT, OR COMMIT TO RESOLVE
                   INDOUBT UR. URID=uridentifier
ATR231D   CANCEL DELAYED. REPLY WAIT, BACKOUT, OR COMMIT TO RESOLVE
                   INDOUBT UR. URID=uridentifier
ATR232D   MEMTERM DELAYED. REPLY WAIT, BACKOUT, OR COMMIT TO RESOLVE
                   INDOUBT UR. URID=uridentifier
ATR233D   CANCEL DELAYED. REPLY BACKOUT, OR COMMIT TO RESOLVE INDOUBT
                   UR.  URID=uridentifier
ATR234D   MEMTERM DELAYED. REPLY BACKOUT, OR COMMIT TO RESOLVE INDOUBT
                   UR.  URID=uridentifier
```

- Task or memory termination of the resource manager address space is delayed because of an outstanding transaction in the "in-doubt" state. RRS forces the operator to manually resolve the "in-doubt" transaction before allowing termination to continue.

How to investigate

If a resource manager address space termination is not completing, review SYSLOG for messages ATR225D - ATR234D.

Recovery actions

Reply to the outstanding WTOR to commit or back out the "in-doubt" units of recovery.

If the "in-doubt" units of recovery is not expected, enter `D RRS,UR,DETAILED,URID=`*`uridentifier`* to determine the resource managers with an interest in the units of recovery. Provide a dump of RRS and the resource managers involved in the units of recovery. Include SYSLOG and formatted LOGREC data to the IBM Support Center

Actions to avoid recurrence

Replying to these WTORs immediately can prevent RRS caused delay to address space termination. Consider setting up automation to ensure that the messages are responded to when they are issued.

RRS transaction hang

Symptoms

The RRS exploiter reports one or more outstanding transactions exist.

How to investigate

Enter the following display command, which issues message ATR624I showing RRS units of recovery (UR) exceptions:

```
DISPLAY RRS,UREXceptions
```

```
ATR624I  hh.mm.ss
         RRS UR EXCEPTION id | SYSTEM URID | WAIT FOR |
         sysname urid waitfortext
```

Message ATR624I displays all the units of recovery that are waiting for completion of other tasks on the specified system. A UR is the RRS representation of a transaction. Examine each exception to determine whether it represents a problem.

Identify persistent exceptions in the display by looking for units of recovery that show the same status in consecutive display output. This condition suggests a possible problem with the transaction. For example, look at the following `D RRS,UREX` example output.

```
D RRS,UREX
ATR624I 14.57.34  RRS UR EXCEPTION
SYSTEM   URID                             WAIT FOR
SYS1     C4D2B56B7EB296E80000432201010000
    SYS2     C4D2B56B7EB3CA5C0000728801020000 RMGR2 PREPARE Exit
SYS1     C4D2B56B7EB29A5C00003E0601010000
    SYS2     C4D2B56B7EB3D4B800006B0A01020000 RMGR2 PREPARE Exit
```

This output from a `DISPLAY` command issued from system SYS1 lists the exception URID followed by the `WAIT FOR` text indented on the next line.

- The two SYS1 exception units of recovery are waiting for cascaded, subordinate units of recovery on SYS2.
- The units of recovery on SYS2 are waiting for resource manager `RMGR2` to complete its `PREPARE Exit`.

Recovery actions

1. Try to resolve the transaction hang.
2. If you cannot get the resource manager that is using RRS to resolve the transaction hang, capture a dump of RRS and the exploiter or exploiters involved in the hang on the systems involved.
3. Provide the dumps with SYSLOG and formatted LOGREC data on the systems involved to the IBM Support Center.

RRS severe error on RRS RM.DATA log stream, message ATR250E

RRS uses system logger to manage resource manager information. A gap or data loss condition against the RRS RM.DATA log stream could lead to a log stream error.

Symptoms

A gap or data loss condition against the RRS RM.DATA log may result in the following message being issued.

```
ATR212I RRS DETECTED LOG DATA LOSS ON LOGSTREAM logstreamname
        DUE TO INACCESSIBLE LOG DATA. LOG DATA FROM lowGMT TO
        highGMT ARE AFFECTED.
```

When the condition is against the RRS RM.DATA log, the following message will also be issued when all RRS systems in the sysplex are at version z/OS V2R1 or higher.

```
ATR250E RRS LOGSTREAM ERROR FOUND. CORRECT THE ERROR OR
        OPTIONALLY REPLY COLDSTART TO BEGIN A RRS INTERNAL COLD
        START.
```

How to investigate

Review SYSLOG for prior messages from system logger that reports the data loss condition.

Recovery actions

RRS needs to be terminated on all systems in the RRS group in order to resolve this problem. Once terminated, request a cold start of RRS using the ATRCOLD procedure and then restart RRS on each system in the RRS group. This can be done manually and will result in all outstanding transactions being lost and not recoverable. Optionally, a reply of COLDSTART may be given to instruct RRS to attempt an Internal Cold Start. RRS will remain active, but new work will not be accepted until the cold start is complete. An attempt will be made to save the in storage transactions which will be relogged as part of the Internal Cold Start procedure.

Actions to avoid recurrence

Create an automation routine that looks for message ATR250E and take the COLDSTART action as described in "Recovery actions."

Resolving RRS problems in a sysplex cascaded transaction environment

The sysplex cascaded transactions environment is complex. When problems occur, determining what actions to take can also be complex. This section is intended to help you:

- Identify the problem
- Identify the systems involved
- Gather the relevant documentation
- Recover transaction processing

This section also covers the following sysplex-specific cascaded transaction environment problems:

- "Sysplex cascaded transaction hang" on page 263
- "Sysplex cascaded transaction hang messages ATR246I and ATR247E" on page 264

RRS sysplex cascaded transaction processing exploiters include IMS TM, OTMA, and APPC synchronous, shared queue, cross queue transaction processing.

Collecting documentation for a sysplex cascaded transaction environment

- **RRS documentation:** Collect an RRS dump and CTRACE as covered in "Collecting documentation for RRS" on page 251. Use a IEADMCxx parmlib member for the RRS dump options and, if possible, use an external writer to trace to a data set as documented in "Important RRS CTRACE information" on page 252.
- **Dumping IMS:** Prepare to dump IMS by creating an IEADMCyy parmlib member and include the following statements:

  ```
  JOBNAME=(IMScontrolregion),
  SDATA=(ALLNUC,LPA,LSQA,PSA,RGN,SQA,TRT,CSA,GRSQ,SUM)
  ```

  Then, when a dump of IMS is necessary, enter the following z/OS command, referencing your IEADMCyy parmlib member:

  ```
  DUMP COMM=(whatever title you choose),PARMLIB=yy
  ```

When you want to dump both RRS and IMS, use the following z/OS command, where *xx* and *yy* are the suffix of the IEADMCxx members created for RRS and IMS:

```
DUMP COMM=(whatever title you choose),PARMLIB=(xx,yy)
```

- **Tracing IMS::** To trace IMS, either run the following traces or specify them in the DFSVSMxx member of SYS1.PROCLIB. If possible, use option log and external trace data sets to externalize the trace data.
  - **For IMS data,:** run or specify IMS RRST trace:
    ```
    /TRA SET ON TABLE RRST
    ```
  - **For OTMA users,** run or specify the OTMT trace:
    ```
    /TRA SET ON TABLE OTMT
    ```
  - **For APPC users,** run or specify IMS LUMI trace:
    ```
    /TRA SET ON TABLE LUMI
    ```

  If you are using external trace data sets, the data can wrap and that IMS does not provide archive support. It is the users responsibility to ensure that the data is not overwritten. Consider automation that copies the inactive data set when IMS switches from one data set to another, or preserve the data manually in problem situations.

Sysplex cascaded transaction hang

Symptoms

Any of the following can alert you to a sysplex cascaded transaction hang:

- IMS Display Region command shows hung threads:
  ```
  /DIS ACT REG
  ```
- IMS shared message queue growth
- RRS display command output shows UR exceptions:
  ```
  DISPLAY RRS,UREXceptions
  ```

How to investigate

- Use the IMS `/DIS ACT REG` command to display status of regions that seem to be hung. If a thread is active in the same state for two consecutive displays, that suggests that the thread is hung. You can also use the `DISPLAY RRS,UREXceptions` command to display status of transaction exceptions.
- When a transaction is hung, obtain a dump of RRS and IMS on both the frontend (FE) and back-end (BE) systems. The FE and BE system names can be determined by the /DIS ACTIVE REGION output. The following z/OS command can be used to obtain the dumps:
  ```
  RO (fesysname,besysname),DUMP COMM=(title you choose),PARMLIB=(xx,yy)
  ```
- If you cannot determine the systems that are involved in the hang, request dumps on each system in the sysplex. Also collect the IMS SLDS and IMS control region job logs for all systems dumped, OPERLOG or SYSLOG, and LOGREC. Use the following command to dump all systems:
  ```
  RO *ALL,DUMP COMM=(title you choose),PARMLIB=(xx,yy)
  ```
- If you find IMS regions that are hung with a status **other than** what is in this list, dump all the IMS and RRS pairs that are participating in the shared queue group:
  - WAIT-SYNCPOINT
  - WAIT-RRS/PC
  - TERM-WAIT SYNCPT

– TERM-WAIT RRS

Recovery actions

If the IMS /DIS ACTIVE REGION display shows a hung dependent region:

- Collect the following documentation from the back-end and frontend systems:
  ```
  RO (fesysname,besysname),DUMP COMM=(title),PARMLIB=(xx,yy)
  ```
- Try to terminate the hung dependent region with the following command:
  ```
  /STO REG ABDUMP
  ```
- If you cannot terminate the hung thread, you might recycle the IMS control region that owns the hung dependent region.
- If the problem persists, check for RRS high processor usage spikes, which suggest that RRS must be recycled.

Sysplex cascaded transaction hang messages ATR246I and ATR247E

Symptoms

Look for the following messages:

- `ATR246I RRS HAS DETECTED A` *`controlblockname`* `CONTROL BLOCK ERROR - UNEXPECTED ERROR DUMP REQUESTED`
- `ATR247E RRS HAS DETECTED A SEVERE ERROR – TERMINATE RMS AND OPTIONALLY REPLY SHUTDOWN TO SHUTDOWN RRS.`

How to investigate

- Message ATR246I indicates that RRS detected a potential problem. RRS takes an unexpected error dump and continues processing without impact. Collect the dump and report this issue to the IBM Support Center.
- Message ATR247E indicates that RRS detects a more severe error, and is expecting that action be taken to terminate RRS. This error might be due to a corrupted control block chain and RRS cannot continue processing.

Recovery actions

For **ATR246I,** no recovery action is necessary.

For **ATR247E,** take the following actions:

1. Collect dumps of RRS and IMS on all systems:
   ```
   ROUTE *ALL,DUMP COMM=(title),PARMLIB=(xx,yy)
   ```
2. Cancel IMS on this system:
   ```
   F IMS,STOP
   ```
3. After IMS is stopped, terminate RRS replying to ATR247E with `SHUTDOWN`. After RRS is terminated, the message is DOMed.
4. Start RRS with the following command:
   ```
   S RRS
   ```
5. After RRS finishes initializing, start IMS with the following command:
   ```
   S IMS
   ```

Actions to avoid recurrence

Create an automation routine that looks for the message ATR247E message and take actions in "Recovery actions."

Chapter 19. System Data Mover (SDM) operational problem determination

The system data mover (SDM) is a DFSMS/MVS component that interacts with data storage subsystems and with various advanced copy services functions to efficiently move large amounts of data. As updates occur to primary volumes, the SDM manages the process of copying those updates to secondary volumes.

The SDM ensures that updates to secondary volumes are made in the same order in which they were made to the primary volumes, maintaining sequence consistency.

Basic SDM problem determination functions

For problem determination with SDM related issues, the following items are used:
- CQUERY DEVN('XXXX') PATHS
- F ANTAS000,LISTSESS ALL (to list XRC sessions on the HW)
- XQUERY ssid SC DETAIL (reports all primary storage controls with volumes in XRC session)
- SYSLOG, LOGREC, JOBLOG, System dumps (General documentation to collect, if available)

SDM specific problems

- "ANTP0095I Unable to determine PPRC paths"
- "ANTX5104E RC=0901 (XRC)" on page 266
- "ANTX5104E RC=0647 REASON=0053 (XRC)" on page 267
- "ANTX5104E RC=0647 REASON=0002 (XRC)" on page 267
- "ANTAS00* ASIDs consuming excessive storage below 2GB" on page 268
- "Converting to IR, RC=1017" on page 269
- "Microcode issue impacting concurrent copy" on page 269

ANTP0095I Unable to determine PPRC paths

While querying paths using TSO CQUERY DEVN(xxxx) PATHS to PPRC secondary devices, the path status reports as "UNABLE TO DETERMINE".

Symptoms

```
  CQUERY DEVN(X'dddd') PATHS.
ANTP0095I CQUERY FORMATTED LVL 3 .
PATHS REPORT.
*************** PPRC REMOTE COPY CQUERY - PATHS ********************.
* PRIMARY UNIT: SERIAL#= 000000serial SSID= ssid SS= 2107 LSS= ls  *.
*              FIRST        SECOND        THIRD        FOURTH     *.
*            SECONDARY     SECONDARY     SECONDARY     SECONDARY   *.
*SERIAL NO: 000000serial  ............  ............  ............ *.
* SSID LSS:   ssid ls       .......       .......       .......    *.
*    PATHS:      0             0             0             0       *.
*          SAID DEST S*  SAID DEST S*  SAID DEST S*  SAID DEST S* *.
*          --------- --  --------- --  --------- --  --------- -- *.
*       1: XXXX XXXX FF  ---- ---- 00  ---- ---- 00  ---- ---- 00 *.
*       2: XXXX XXXX FF  ---- ---- 00  ---- ---- 00  ---- ---- 00 *.
*       3: XXXX XXXX FF  ---- ---- 00  ---- ---- 00  ---- ---- 00 *.
*       4: XXXX XXXX FF  ---- ---- 00  ---- ---- 00  ---- ---- 00 *.
*  SUBSYSTEM        WWNN                 LIC LEVEL              *.
* -----------  ----------------          -----------            *.
* PRIMARY....  (primary wwnn  )          x.x.xx.xxx             *.
* SECONDARY.1  (secondary wwnn)                                 *.
*                                                               *.
* S* = PATH STATUS:                                             *.
* 00=NO PATH         01=ESTABLISHED ESCON   02=INIT FAILED       *.
* 03=TIME OUT        04=NO RESOURCES AT PRI 05=NO RESOURCES AT SEC*.
* 06=SERIAL# MISMATCH 07=SEC SSID MISMATCH   08=ESCON LINK OFFLINE *.
* 09=ESTABLISH RETRY  0A=PATH ACTIVE TO HOST 0B=PATH TO SAME CLUSTR*.
* 10=CONFIG ERROR     FF=UNABLE TO DETERMINE                     *.
****************************************************************** .
ANTP0001I CQUERY COMMAND COMPLETED FOR DEVICE dddd. COMPLETION CODE: 00 .
```

How to investigate

This query output is a normal response from the PPRC secondary devices. Although the paths are defined, they are not recognized until they are actually in use.

Recovery actions

No applicable recovery.

Actions to avoid recurrence

None.

ANTX5104E RC=0901 (XRC)

This error is issued if the data mover detects either a No Record Found or Invalid Track Format error on a volume. The data mover will automatically reinitialize the track on which the error occurred. If a subsequent error occurs during resynchronization for the same volume, the volume pair will be suspended.

Symptoms

This error is only reporting the “out of synch” condition when XRC detects it. Post analysis is needed to determine how the volume pair got out of synch.

How to investigate

Run the Dynamic Volume Compare Utility to scan for any discrepancies between volumes. See Technote #T7000248 at

`http://www-01.ibm.com/support/docview.wss?uid=isg3T7000248`

for instructions on using the utility.

FTP the following documentation to the IBM support center for further diagnostics:

- DUMP (if available)

  If not, ensure the following diagnostic flags are enabled to collect a dump on any future hits:

  ```
    F ANTAS00n,CTFLG NRFITF ON
    F ANTAS00n,CTFLG TIF ON
    F ANTAS00n,CTFLG TIF_ERROR ON
    F ANTAS00n,CTFLG ABEND_LIC ON
  ```

- SYSLOG
- LOGREC

Recovery actions

The user should XDELPAIR, then XADDPAIR the volume pair that received the RC901 error. This will perform a full copy of the data, ensuring the volume is 100% mirrored. After the mirror is up,

Technote: See: http://www-01.ibm.com/support/docview.wss?uid=isg3T1018957

Actions to avoid recurrence

ANTX5104E RC=0647 REASON=0053 (XRC)

XRC has detected this condition during normal data mover processing. The scope of this condition is for a single storage control session. After you determine the reason for the error and correct the condition, you can issue an XADDPAIR command to add the suspended volumes back to the session. This is for the 5695DF117 System Data Mover (SDM) component.

Symptoms

Error message reported:

```
ANTX5119E XRC SESSION(session) ENCOUNTERED AN ERROR PROCESSING STORAGE CONTROL
ssid SESSION ##, RC=0647 REAS=0053 SRVC=0106
```

If a SUSSESS ALL command was entered either directly by the user, or by automation, then this is an expected condition for every session in the data mover.

If unexpected, look for other ANTXxxxxE messages containing a different return and reason code in the SYSLOG around the time of the suspend. This will be the original reason for the suspend. The REAS53 messages are symptoms of the original suspend

How to investigate

This message is a result of the suspension and a preceding message should be searched for.

FTP the following documentation to the IBM support center for further diagnostics: SYSLOG

Recovery actions

No applicable recovery.

Actions to avoid recurrence

ANTX5104E RC=0647 REASON=0002 (XRC)

An attempt to issue I/O from the SDM host to the primary storage subsystem has timed out. The timeout value is stored in the user defined StorageControlTimeout value. This is for component 5695DF117 System Data Mover (SDM).

Symptoms

Error messages:
- ANTX5119E XRC SESSION(session) ENCOUNTERED AN ERROR PROCESSING STORAGECONTROL ssid SESSION ##, RC=0647 REAS=0002 SRVC=0106
- Possible IOS messages

How to investigate

Typically, the presence of IOS messages indicate a networking issue. The I/O timeout is justified by the loss of connectivity from the SDM hosts to the primary storage subsystem. Check the SYSLOG for IOS messages. If present, users should follow up with their networking vendor. Once the network is stabilized, the user can resume the mirror.

FTP the following documentation to the IBM support center for further diagnostics: SYSLOG

Recovery actions

Correct any connectivity issues in the hardware network.

Actions to avoid recurrence

ANTAS00* ASIDs consuming excessive storage below 2GB

When running close to the real storage installation, you may experience messages indicating storage shortages on SDM LPARs. Because SDM address spaces use a large amount of fixed storage, the SDM address spaces will always report first in the IRA404I messages as users of large amounts of storage.

Symptoms

Error messages reported:
- IRA400E 04,PAGEABLE STORAGE SHORTAGE
- IRA404I ANTAS00x ASID 00xx OWNS 0000###### PAGES, 0000###### FIXED, 0000###### FIXED IN SHORTAGE AREA
- IRA401E 04,CRITICAL PAGEABLE STORAGE SHORTAGE (sometimes)

How to investigate

If you aren't witnessing a *IEA602I ADDRESS SPACE CREATE FAILED message, then this can be considered only a warning until storage is adjusted.

F ANTAS000,DUMP (generates dump, gathering storage usage from all address spaces)

Note: If no impact to environment, and only warnings are displayed, follow-up during next business day.

What to send:
- DUMPs (Generated by previous command, from all address spaces)
- SYSLOG (Including timestamps leading up to and from reported error)

Recovery actions

Reduce storage demand by terminating job tasks until shortage is relieved.

Actions to avoid recurrence

Converting to IR, RC=1017

While converting from normal XRC to XRC-IR (PPRC and XRC), clients received error messages indicating there was a mismatch of storage control sessions on the SWAP (PPRC secondary) storage controls. This is for component 5695DF117 System Data Mover (SDM).

Symptoms

Error messages reported:
- ANTX5129E XRC SESSION(session) ERROR ASSOCIATING SWAP VOLUME(####) SCSESSION(aa) WITH PRIMARY VOLUME(volume) SCSESSION(aa), ANTX5129E (CONT) RC=1017 REAS=0
- ANTA5107E XADDPAIR FAILED FOR VOLUME PAIR(volume,XRCUTL) FOR SESSION(session), RC=1017 REAS=0

How to investigate

FTP the following documentation to the IBM support center for further diagnostics:
- F ANTAS000,LISTSESS #### (for both primary and swap controllers, where #### is the device number)
- F ANTAS000,LISTDVCS #### ss (for all devices in affected session, #### is the device number, and ss is the storage control session number)
- F ANTAS00n,DUMP (Dump of the impacted data mover address space)
- SYSLOG
- LOGREC

Recovery actions

Consult IBM Software Technical Support.

Actions to avoid recurrence

Microcode issue impacting concurrent copy

A known open microcode defect causes a counter on the box to reach its maximum limit without clearing itself. This counter keeps track of active sessions on a control unit (in hardware). There is a limit to how many active sessions (XRC and CC) you can have on a control unit, so when this counter is maxed out, the client is unable to execute more Concurrent Copy sessions. This issue is regarding the 5695DF117 System Data Mover (SDM) component.

Symptoms

Error messages reported:
- ANTPC2 messages flood the LOGREC, one for every CC request, which can be hundreds.

How to investigate

- This issue requires the user to take a hardware statesave on the box. A statesave is like taking a dump, but it also performs "checks and balances".
- The user should open a Hardware PMR to request the statesave.

Recovery actions

Engage DASD Hardware Support for recovery assistance.

Actions to avoid recurrence

Chapter 20. VSAM component operational problem determination

VSAM is an access method that arranges records in data sets by an index key, relative record number or relative byte addressing. VSAM is used for direct or sequential access to either fixed-length or variable-length records on DASD. See the manual *DFSMS Using Data Sets* for more information on VSAM data sets.

Basic VSAM problem determination functions

For problem determination with VSAM data sets, the following items are used:
- IDCAMS LISTCAT ALL COMMAND
- IDCAMS EXAMINE DATATEST INDEXTEST for VSAM data sets with an index component
- SMF type 60 through 66 records contain information about VSAM data sets, including definition, deletion, altering of VSAM data sets
- System dumps produced by VSAM in certain conditions with no customer request
- System dumps produced as a result of issuing the command F CATALOG,VDUMPON to generate a dump when certain conditions occur
- EREP reports in some cases where a call to media manager produced an unexpected result.

VSAM specific problems

- "VSAM Index Trap"
- "Hang in VSAM record management code" on page 273
- "Loop in VSAM record management code" on page 273
- "Unexpected return codes from VSAM record management" on page 274
- "Issues opening, closing, extending VSAM data sets" on page 275

VSAM Index Trap

When writing a record to a data set with an index or an ICF catalog, VSAM record management will check to see if the adding or updating of the record will damage the index component. If the code detects that it will damage the index, the request is **not** done and any changes made during the request will be backed out.

Symptoms

- Message IDAI1001E for VSAM data sets
- Message IDAI1002E for ICF Catalogs
- RPL Feedback code of X'xx08006D' for the request that detected this
- RPL Feedback code of X'xx08006E' for subsequent accesses
- System dump produced VSAM DYNAMIC RPL DUMP - IDAM19R3 +0xxx FEEDBACK CODE: 0108006D

How to investigate

1. Close VSAM data set if possible.

2. Run IDCAMS LISTCAT ALL of the data set and any associated data sets, such as AIX's

```
//STEP1    EXEC  PGM=IDCAMS,REGION=0M
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD *
  LISTCAT ENT(VSAM.DATA.SET) ALL
```

3. If the data set is closed, run the following IDCAMS commands:
   EXAMINE INDEXTEST NODATATEST ... followed by
   EXAMINE NOINDEXTEST DATATEST

```
//STEP1    EXEC  PGM=IDCAMS,REGION=0M
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD *
 EXAMINE VSAM.DATA.SET INDEXTEST NODATATEST
 EXAMINE VSAM.DATA.SET NOINDEXTEST DATATEST
```

4. If the data set cannot be closed or is an ICF Catalog, run the following IDCAMS commands:
   VERIFY
   EXAMINE IND EXTEST NODATATEST
   EXAMINE INDEXTEST NODATATEST,
   EXAMINE DATATEST NOINDEXTEST
   EXAMINE DATATEST NOINDEXTEST

```
//STEP1    EXEC  PGM=IDCAMS,REGION=0M
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD *
  VERIFY DATASET(VSAM.DATA.SET)
  EXAMINE VSAM.DATA.SET INDEXTEST NODATATEST
  EXAMINE VSAM.DATA.SET INDEXTEST NODATATEST
  EXAMINE VSAM.DATA.SET.NOINDEXTEST DATATEST
  EXAMINE VSAM.DATA.SET NOINDEXTEST DATATEST
```

5. Preserve the data set by renaming the cluster, data component and index component in case further information is needed from the data set. If the VSAM data set has associated components such as AIX's and PATHs these should also be renamed

```
//STEP1    EXEC  PGM=IDCAMS,REGION=0M
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD *
   ALTER VSAM.DATA.SET –
     NEWNAME(VSAM.DATA.SET.SAVE)
   ALTER VSAM.DATA.SET.DATA –
     NEWNAME(VSAM.DATA.SET.DATA.SAVE)
   ALTER VSAM.DATA.SET.INDEX -
     NEWNAME(VSAM.DATA.SET.INDEX.SAVE)
```

6. For ICF Catalogs, perform a DFDSS Physical Dump of the catalog, in case further information is needed.

```
//STEP001 EXEC PGM=ADRDSSU
     //SYSPRINT DD SYSOUT=*
     //DSYS004  DD DISP=(,CATLG),DSN=SYS1.DSSPHY.DUMP,
     //            UNIT=SYSDA,VOL=SER=XXXXXX,
     //            SPACE=(CYL,(1110,200),RLSE)
     //ISYS004  DD DISP=SHR,UNIT=SYSDA,VOL=SER=SYS004
     //SYSIN    DD *             DUMP -
       DS(INCL(CATALOG.NAME)) -
           PHYSINDDNAME( -
               (ISYS004) -
               ) -
           OUTDDNAME(DSYS004)
```

7. Gather SMF type 60 through 66 records from all systems that may have accessed the data set or catalog from a time when the data set or catalog was known to be good.

8. Submit the output from the above to the Support Center along with syslog and joblog (if available).

Recovery actions

- If the IDCAMS EXAMINE commands run with a return code of 4 or less, the data set or catalog should not need any further processing before being used.
- If the IDCAMS EXAMINE commands return a return code of 8 or higher, then the data set or catalog will need to be recovered using your normal recovery processing.

Actions to avoid recurrence

S

Hang in VSAM record management code

When processing a VSAM data set, a hang may occur in VSAM record management code

Symptoms

- Job that is processing VSAM data sets stops doing I/O to the VSAM data sets and is not using CPU
- RMF or other Monitors may indicate job is waiting in a VSAM module named IDA019xx

How to investigate

1. Obtain system dump of address space with the following SDATA: SDATA=(ALLNUC,CSA,GRSQ,LPA,LSQA,PSA,RGN,SQA,SUM,SWA,TRT)
2. If the monitor includes detail on the data set that the last request was for, obtain an IDCAMS LISTCAT ALL of the data set and any associated data sets such as AIX's

```
//STEP1    EXEC  PGM=IDCAMS,REGION=0M
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD *
  LISTCAT ENT(VSAM.DATA.SET) ALL
```

3. Submit the dump, LISTCAT output (if available), joblog from the job that is hung, and syslog to the Support Center.

Recovery actions

Cancel the hung job and attempt to restart.

Actions to avoid recurrence

Loop in VSAM record management code

When processing a VSAM data set, a loop may occur in VSAM record management code.

Symptoms

- A job that is processing VSAM data sets stops doing I/O to the VSAM data sets and is continuing to use CPU

- RMF or other Monitors may indicate that a job is using CPU in a VSAM module named IDA019xx
- Some applications may indicate that a loop is occurring. For example, CICS will issue message DFHFC0004 indicating a loop at offset X'FFFF' in module DFHFCVR.

How to investigate

1. Obtain system dump of address space with the following SDATA: SDATA=(ALLNUC,CSA,GRSQ,LPA,LSQA,PSA,RGN,SQA,SUM,SWA,TRT)
2. If the monitor includes detail on the data set that the last request was for, obtain an IDCAMS LISTCAT ALL of the data set and any associated data sets such as AIX's
```
//STEP1    EXEC  PGM=IDCAMS,REGION=0M
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD *
  LISTCAT ENT(VSAM.DATA.SET) ALL
```
3. Submit the dump, LISTCAT output (if available), joblog from the job that is hung, and syslog to the Support Center.

Recovery actions

Cancel or shutdown the hung job and attempt to restart.

Actions to avoid recurrence

Unexpected return codes from VSAM record management

Code to process VSAM data sets will normally handle expected return codes such as No Record Found, Duplicate Key, and Successful request. However at times, other return codes may be returned that are not handled by the code or indicate an error in VSAM record management processing.

Symptoms

The application or program issues a message or otherwise indicates it received an unusual return code from a request to VSAM record management.

How to investigate

1. Review the RPL Feedback code returned with the descriptions in the manual *DFSMS: Macro Instructions for Data Sets*. If the RPL feedback code is one that the job/application should handle, change the application to handle that feedback code.
2. If the RPL Feedback code indicates the problem may be in VSAM code a dump may be needed to determine the cause. Review the use the of the F CATALOG,VDUMPON command and issue the command to capture a dump or contact the Support Center who can assist in providing the values for the F CATALOG,VDUMPON command.
3. Submit the dump, joblog from the job that is receiving the return code, and syslog to the Support Center.

Recovery actions

- If the job/application needs to be changed, change the application.
- The Support Center will provide guidance for any other cases.

Actions to avoid recurrence

Issues opening, closing, extending VSAM data sets

At times there may be issues with opening, closing or extending VSAM data sets.

Symptoms

- Message IEC161I is issued for open problems (or in some cases information)
- Message IEC251I is issued for close problems (or in some cases information)
- Message IEC070I is issued for extend problems

How to investigate

1. Review the messages and codes manual for the specific return and reason codes. If they are informational decide if further action is necessary from your applications perspective. For example, some common messages such as

   ```
   IEC161I 001(DW)-255,P0331406,STEP1,AURM,,,VSAM.DATA.SET
   ```

   are informational as are some other messages such as the following:

   ```
   IEC161I 056-084,P0331406,STEP1,AURM,,,VSAM.DATA.SET
   IEC161I 062-086,P0331406,STEP1,AURM,,,VSAM.DATA.SET
   ```

   The return and reason codes for messages IEC251I and IEC070 are documented under message IEC161I
2. In some cases VSAM may take a DUMP during this processing. These dumps will have a title similar to the following:

   ```
   DUMP TITLE=VSAM O/C/EOV FFDC DUMP - IDA0192Y + 00000298 RC=020 CCC=054
   ```

   The return and reason codes are the same as those documented for messages IEC161I. The reason for the dump being taken was to gather doc at the time of the issue.
3. If the reason for problem in open, close or extend processing is not clear from the message description, submit the joblog, syslog and any dumps produced to the support center .

Recovery actions

- If the job/application needs to be changed according the messages received, change the job/application.
- The Support Center will provide guidance for any other cases.

Actions to avoid recurrence

Chapter 21. VSAM record-level sharing (RLS) operational problem determination

VSAM record-level sharing (RLS) is a data set access mode that allows multiple address spaces, CICS application owning regions (AORs) on multiple MVS systems, and jobs to access data at the same time. With VSAM RLS, multiple CICS systems can directly access a shared VSAM data set, eliminating the need for function shipping between application owning regions (AORs) and file owning regions (FORs). CICS provides the logging, commit, and rollback functions for VSAM recoverable files; VSAM provides record-level serialization and cross-system caching. CICS, not VSAM, provides the recoverable files function.

For additional info reference chapter 16 of *z/OS DFSMSdfp Storage Administration*.

Basic VSAM record-level sharing (RLS) problem determination functions

The following is a list of the commands that may be useful in diagnosing VSAM RLS problems. This is a referential list, and will be repeated in the subsequent sections about various possible problems.

Display commands:

- D GRS,C
  - Shows any outstanding GRS conflicts. Watch for SYSVSAM ENQs.
- D SMS,SMSVSAM,DIAG(C)
  - Lists any outstanding latch contention and the associated delays and the TCBs involved.
  - SCOPE=SYSTEM
- D SMS,CFLS
  - • Shows current lock structure information, such as false contention and lock rate.
- D XCF,STR,STRNM=strname (usually IGWLOCK00 or cache structure)
  - Lists XCF information related to the SMSVSAM coupling facility structures.
  - Can be used for lock structures or cache structures.
- D SMS,SMSVSAM,QUIESCE
  - Outlines any active quiesce activity within the SMSVSAM region on that system. This can often reveal potential hangs if a registered region is not responding.
  - SCOPE=SYSTEM
- D SMS,SMSVSAM,ALL
  - Shows the status of SMSVSAM around the sysplex.
  - SCOPE=SYSPLEX
- D SMS,TRANVSAM,ALL
  - Provides the status of Transactional VSAM (TVS) around the PLEX.
  - SCOPE=SYSPLEX

Bringing up/Taking down SMSVSAM:

- V SMS,SMSVSAM,ACTIVE

  - Starts SMSVSAM after a TERMINATESERVER
- V SMS,SMSVSAM,TERMINATESERVER
  - Brings down the SMSVSAM ASID

If TERMINATESERVER fails:
- • FORCE SMSVSAM,ARM
  - Force down SMSVSAM while still allowing recovery routines to run
- • FORCE SMSVSAM
  - Force SMSVSAM into end-of-memory

Dump commands:
- Console dump

```
DUMP COMM=(some meaningful dump title)
   R xx,JOBNAME=(SMSVSAM,XCFAS),CONT
   R yy,DSPNAME=('SMSVSAM'.*,'XCFAS'.*),CONT
   R nn,SDATA=(PSA,NUC,SQA,LSQA,SUM,RGN,GRSQ,LPA,
                 TRT,CSA,XESDATA),CONT
  R zz,REMOTE=(SYSLIST=(*('SMSVSAM')),DSPNAME,SDATA),END
```

- Dump using IEADMCxx PARMLIB member:

```
JOBNAME=(*MASTER*,SMSVSAM),DSPNAME=('SMSVSAM'.*),
SDATA=(COUPLE,PSA,NUC,SQA,LSQA,SUM,RGN,GRSQ,LPA,TRT,CSA,XESDATA),
REMOTE=(SYSLIST=(*('SMSVSAM')),DSPNAME,SDATA),END
```

VSAM record-level sharing (RLS) specific problems

- "HANG/WAIT in RLS/TVS" on page 279
- "ABEND0F4 failures"
- "SMSVSAM will not start up" on page 280
- "Share Control Datasets not specified" on page 280

ABEND0F4 failures

Symptoms

In the event of a logic error internal to the RLS code, an ABEND0F4 dump will be generated. In many cases, DUMPSRV will also generate dumps on the SMSVSAM images across the plex. The symptom string will include the 5695DF122 (COMPID=DF122) component ID. No additional console commands are typically necessary to diagnose this type of error.

How to investigate

Search IBM support portal for any APARs that match the symptoms of the ABEND

Recovery actions

For most ABEND issues, there is no immediate solution. However, recovery may involve restarting SMSVSAM, if it has not automatically done so. For many issues, no actions are required. To identify any pertinent recovery actions, check the IBM support portal for any APARs that match the symptoms of the ABEND.

Actions to avoid recurrence

Listed in APAR if available.

HANG/WAIT in RLS/TVS

Symptoms

If transactions are not processing, quiesces not quiescing, or data sets failing to open or close, you may have a form of hang within the SMSVSAM address space.

How to investigate

In order to troubleshoot the vast majority of RLS hang / wait / slowdown / loop situations, the following diagnostic commands will need to be issued:

```
- D GRS,C                     - ENQ contention (system level)
- D SMS,SMSVSAM,DIAG(C)       - RLS latch contention (system level)
- D SMS,SMSVSAM,QUIESCE       - Quiesce event status (system level)
- IDCAMS SHCDS LISTALL        - lists registered subsystems & lock info
- D SMS,CFLS(lock_structure)  - displays lock structure information
- D XCF,STR,STRNM=[IGWLOCK00 | secondary_lock_structure]
                               - another display of the lock str
```

For all "system level" commands, ensure that they are issued on every system in the plex. The best way to accomplish this is to use the route command. For example, RO *ALL,D GRS,C

Once the commands have been issued, dump SMSVSAM around the plex. Be sure to including the DATASPACEs for RLS as well as a minimum of SDATA parms GRSQ & XESDATA. If any CICS regions are affected, ensure that they are added to the dump specification as well.

Here is an example command to dump RLS, XCF and a CICS region on one system:

```
DUMP COMM=(some meaningful dump title)
 R xx,JOBNAME=(SMSVSAM,XCFAS,CICS1),CONT
 R yy,DSPNAME=('SMSVSAM'.*,'XCFAS'.*),CONT
 R nn,SD=(COUPLE,PSA,NUC,SQA,LSQA,SUM,RGN,GRSQ,LPA,TRT,CSA,XESDATA),END
```

Adding the REMOTE keyword will issue the same dump command on each member in the plex:

```
DUMP COMM=(some meaningful dump title)
 R xx,JOBNAME=(SMSVSAM,XCFAS,CICS1),CONT
 R yy,DSPNAME=('SMSVSAM'.*,'XCFAS'.*),CONT
 R nn,SD=(COUPLE,PSA,NUC,SQA,LSQA,SUM,RGN,GRSQ,LPA,TRT,CSA,XESDATA),CONT
 R zz,REMOTE=(SYSLIST=(*('SMSVSAM')),DSPNAME,SDATA),END
```

The dumping process can be simplified by including an entry similar to the following example in the IEADMCxx PAMRLIB member:

```
JOBNAME=(*MASTER*,SMSVSAM,CICS1),DSPNAME=('SMSVSAM'.*),
SDATA=(COUPLE,PSA,NUC,SQA,LSQA,SUM,RGN,GRSQ,LPA,TRT,CSA,XESDATA),
REMOTE=(SYSLIST=(*('SMSVSAM')),DSPNAME,SDATA)
```

Once the member is created, issuing DUMP COMM=(title),PARMLIB=xx will dump RLS all around the plex.

Recovery actions

As with all RLS problems, please be sure to collect appropriate documentation before attempting to clear the issue. Without documentation, support will be unable to verify the cause of the problem.

FTP the following documentation to the IBM support center for further diagnostics:

- DUMPs
- OPERLOG (or SYSLOG from all plex systems)
- LOGREC (from all systems)
- JOBLOGS (any which may be pertinent)

To clear the issue, start by investigating the task/lock/latch that the diagnostic command indicate is holding up the system and then attempt to clear the specific resource by cancelling the affected job / transaction / region / or system. Info APAR II14597 provides a detailed step by step set of instructions for this and other common scenarios.

Actions to avoid recurrence

SMSVSAM will not start up

Symptoms

If the SMSVSAM address space is active, but no IGW414I message is displayed.

How to investigate

Issue a D SMS,SMSVSAM,ALL command. The output will show the current status of the SMSVSAM instances around the plex. The STEP should indicate SmsVsamInitComplete if there were no problems. For any other issue, contact the support center and provide the OPERLOG showing the output of D SMS,SMSVSAM command

```
IGW420I DISPLAY SMS,SMSVSAM,ALL
DISPLAY SMS,SMSVSAM - SERVER STATUS
  SYSNAME:  SYSTEM1  UNAVAILABLE ASID: 00C5 STEP: SHC_Ph2_Init
....
```

Recovery actions

Actions to avoid recurrence

Share Control Datasets not specified

Symptoms

Receiving IGW611A and IGW609A

```
*08.41.23 SYSTEM1        *IGW611A SHARE CONTROL DATA SET NEVER ASSIGNED
*08.41.23 SYSTEM1        *IGW609A NO SPARE SHARE CONTROL DATA SETS
* EXIST. IMMEDIATE ACTION REQUIRED
```

How to investigate

Issue:

```
D SMS,SMSVSAM,ALL
```

If it displays a STEP of SHC_PH2_Init, this means that SMSVSAM is waiting for the user to enter the name of a Share Control Dataset (SHCDS) for SMSVSAM.

Recovery actions

In a normal setup we require 2 active SHCDS datasets, and 1 spare. If the SHCDS have already been defined, then simply issue the following commands to add them to SMSVSAM:

```
V SMS,SHCDS(shcds.name),NEW
V SMS,SHCDS(shcds.sparename),NEWSPARE
```

If they are not yet defined, refer to the *DFSMSdfp Storage Administration* manual, section “Defining sharing control data sets”.

Actions to avoid recurrence

Ensure SHCDs are properly defined prior to starting SMSVSAM.

Part 6. Diagnosis reference material

Before calling IBM, it is important to gather the correct information. The following topics can help you find specific diagnosis information for z/OS base elements and features and help you have the correct information available to discuss with the IBM support specialist.

Chapter 22. Diagnosis information for z/OS base elements and features

In addition to the reading topic about Part 5, “Diagnosing component-specific problems,” on page 237, find specific diagnosis information for z/OS base elements, features, and products that run on z/OS in the IBM Knowledge Center (www.ibm.com/support/knowledgecenter/SSLTBW/welcome).

Chapter 23. Reporting problems to IBM

Before you begin: Be familiar with the information in this document; know how to collect the data that your software specialist needs to solve your problem.

This chapter covers the following topics:
- "Software support service checklist" including severity levels and examples.
- "Automatic problem reporting" on page 289
- "Invoking IPCS as a background job" on page 289 which includes "Step for invoking IPCS as a background job" on page 289

Software support service checklist

In order to understand and resolve your software support service request in the most expedient way, it is important that you gather information about the problem and have it on hand when discussing the situation with the software specialist. The following information is required:
- Definition of the problem

  Note:

  It is very important that you are as specific as possible in explaining a problem or question to our software specialists. Our specialists want to be sure that they provide you with exactly the right solution so, the better they understand your specific problem scenario, the better they are able to resolve it. To assist you with problem identification, see the Chapter 24, "Problem diagnostic worksheet," on page 291 (in Appendix A).
- Background information

  Note:

  If possible, obtain all data about a problem soon after the problem occurs. Otherwise, updates to the system can cause discrepancies in the data. Ask yourself the following questions:
  - What levels of software were you running when the problem occurred? Please include all relevant products, for example: operating system as well as related products.
  - Has the problem happened before, or is this an isolated problem?
  - What steps led to the failure?
  - Can the problem be recreated? If so, what steps are required?
  - Have any changes been made to the system? (workload, hardware, netware or software)
  - Were any messages or other diagnostic information produced? If yes, what were they?
  - It is helpful to have the message number(s) of any messages received when you place the call for support or to document in the ETR.
  - Define your technical problem statements in specific terms and provide the version and release level of the product(s) in question.

- Relevant diagnosis information

  Note:

  It is often necessary that our software support specialists analyze specific diagnostic information, such as storage dumps and traces, in order to resolve your problem. Gathering this information is often the most critical step in resolving your problem. Product specific diagnostic documentation can be very helpful in identifying what information is typically required to resolve problems. You should keep all problem data until the problem is resolved or until the data is successfully transmitted to IBM. The following are examples of problem data that the support center might ask you to provide:
  - Any changes made to the system recently, preceding when the problem began occurring (for example, PTFs or new products installed or new hardware).
  - Problem type (for example: abend, hang, loop)
  - Search arguments
  - Dump data, see “Invoking IPCS as a background job” on page 289
  - Failing input request: macro, command, or statement
  - SDWAVRA keys, lengths, and contents
  - Offset of the failing instruction into the module or CSECT
  - Accompanying messages: identifiers and texts
  - Logrec report, if used
  - All printed output and output data sets related to the problem
  - Data on any related problems
  - Module name and level
  - Name and level of the operating system(s) with a list of program temporary fixes (PTF) applied at the time of the problem and all installation modifications, exits, and products with other than Class A service
  - Other problem data developed while using the diagnosis book for the component, subsystem, or program
- Severity level

  You need to assign a severity level to the problem when you report it, so you need to understand the business impact of the problem you are reporting. A description of the severity levels is in the following table.

Table 23. Severity levels and examples

Severity	Definition
1	Critical Impact/System Down: Business critical software component is inoperable or critical interface has failed. This indicates you are unable to use the program resulting in a critical impact on operations. This condition requires an immediate solution.
2	Significant impact: A software component is severely restricted in its use, causing significant business impact. This indicates the program is usable but is severely limited.
3	Moderate impact; A noncritical software component is malfunctioning, causing moderate business impact. This indicates the program is usable with less significant features.
4	Minimal impact; A noncritical software component is malfunctioning, causing minimal impact, or a nontechnical request is made.

- Mention the following items that apply to your situation:
  - If you are under business deadline pressure

- When you are available (for example, when you will be able to work with IBM Software Support)
- Where you can be reached
- A knowledgeable alternate contact with whom IBM can speak
- Other open problems (PMRs/Incidents) with IBM regarding this service request
- If you are participating in an early support program (ESP)
- If you have researched this situation prior to calling IBM and have detailed information or documentation to provide for the problem.

Automatic problem reporting

Parts of the system automatically report the need for service to IBM; for example, the central processor complex (CPC) reports problems directly to IBM. If the system contains a Hardware Management Console (HMC), you should be aware that problems in the Sysplex Timer and in direct access storage devices (DASD) might be automatically reported, even though the problems are recorded by MVS:

- MVS captures information about the problems and creates the following logrec records:
  - ETR record: For problems in the Sysplex Timer
  - DASD-SIM record: For problems in DASD
- For a unique Sysplex Timer or DASD error, HMC creates a problem record (PMR) in RETAIN to notify IBM that service is needed.

Invoking IPCS as a background job

Before calling IBM, format the dump using the IPCS subcommand recommended in the appropriate procedure. Some IPCS subcommands take time to run for a large dump and transferring the dump to IBM is often time consuming. You can start an IPCS session before calling IBM. Then, during the call, the output can be browsed as needed. See “Step for invoking IPCS as a background job.”

Sometimes you might discover it is easier to create a second dump with a subset of the original dump. Send the second dump to IBM for initial diagnosis. For example, sometimes only the address spaces of the suspected problem job are necessary to diagnose system hangs. Of course, this is not always the case, so if you do send a subset of the dump to IBM, do not delete the original dump.

Step for invoking IPCS as a background job

Use the IPCS **COPYDUMP** subcommand to reduce the size of a very large dump, such as a stand-alone dump. An initial review of ASIDs 1-10 and others that are known to be involved in the problem, greatly reduces the size and transfer time of the dataset to be sent to the support center for initial diagnosis.

Use **COPYDUMP** to extract the problem address spaces that you want to analyze. **COPYDUMP** always includes address spaces 1 through 4 in the new dump data set, as well as any data spaces associated with the address spaces. Use the **LISTDUMP** subcommand to see the address spaces available in a dump data set.

The IPCS **COPYDUMP** subcommand can copy a single unformatted dump from one data set to another. Use the following example JCL to guide your creation of a batch job that invokes IPCS as a background job. This job opens the dump and extracts the desired ASIDs, using the IPCS **COPYDUMP** command, and saves the result in another data set.

```
//RLWG JOB '796634,?,S=I','RL WRIGHT',MSGLEVEL=(2,1),
// CLASS=2,NOTIFY=RLW,MSGCLASS=H
//IKJEFT01 EXEC PGM=IKJEFT01,REGION=4096K,DYNAMNBR=50
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
IPCS DEFER NOPARM
COPYDUMP INDSNAME('IPCS1.SYSOS18.SV03235') +
  OUTDSNAME('IPCS1.SYSOS18.SV03235.SUBSET') +
  ASIDLIST(1:10,15) SPACE(15000 10000) +
  NOCONFIRM
END
```

Figure 49. JCL for invoking IPCS as a background job

The options on the example IPCS command do the following:

DEFER
Defers the use of a dump directory. In this example **COPYDUMP** had no need for the dump directory.

NOPARM
Indicates not to use an IPCSPRnn parmlib member. This eliminates the allocation of the problem and data set directories named by IPCSPR00 on the production system.

For a complete list of IPCS options, see *z/OS MVS IPCS Commands*.

Chapter 24. Problem diagnostic worksheet

Use this worksheet when calling IBM Technical Support to help you resolve your problem.

Table 24. What is the impact of your problem?

Impact
1. Critical business impact 2. Significant business impact 3. Some business impact 4. Minimal business impact
Is there a system outage? If yes, how many systems are affected:
Is the problem repetitive? Can you recreate the problem? Number of occurrences:
Details:

Table 25. How is your system configured?

System environment and level
CP model and serial number:
z/OS level:
How many systems are involved with the problem? • LPAR • VM • NATIVE • Sysplex
What other hardware devices are involved?

Table 26. What are the external symptoms

External symptoms
• Coded system wait state • System hung or partitioned from sysplex • Loop or high system overhead • Loop or high CP usage by job • Job/subsystem/application/function failure • Job/subsystem/application/function hang • Output incorrect or missing • Performance or slowdown • Error message issued

Table 26. What are the external symptoms (continued)

External symptoms
Details:

Table 27. What symptom information did you collect?

Symptoms extracted from diagnostic information
Dump title:
ABEND code(s):
Wait state code:
Message ID(s):
Module name(s) and rmid:
Component ID(s):
Other:

Table 28. Which type of documentation did you obtain?

Documentation obtained
Dump produced: • SYSABEND, SYSUDUMP, CEEDump (formatted dump) • SLIP, Console, SVC, SYSMDUMP, TDUMP (unformatted dump) • SADUMP
Joblog, SYSLOG, OPERLOG or other:
EREP report of SYS1.LOGREC:
GTF data set:
CTRACE data set:
Other:

Table 28. Which type of documentation did you obtain? (continued)

Documentation obtained

Table 29. What recovery actions did you attempt?

Recovery actions
• Program terminated • Job canceled • Job restarted • Job forced • Device taken offline • Restart key on HMC selected • System partitioned or re-ipled • Sysplex restarted
Other:

Part 7. Appendixes

Appendix. Accessibility

Accessible publications for this product are offered through the z/OS Information Center.

If you experience difficulty with the accessibility of any z/OS information, please send a detailed message to mhvrcfs@us.ibm.com or to the following mailing address:

IBM Corporation
Attention: MHVRCFS Reader Comments
Department H6MA, Building 707
2455 South Road
Poughkeepsie, NY 12601-5400
USA

Accessibility features

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size.

Using assistive technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using such products to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. Refer to *z/OS TSO/E Primer*, *z/OS TSO/E User's Guide*, and *z/OS ISPF User's Guide Vol I* for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

Dotted decimal syntax diagrams

Syntax diagrams are provided in dotted decimal format for users accessing the z/OS Information Center using a screen reader. In dotted decimal format, each syntax element is written on a separate line. If two or more syntax elements are always present together (or always absent together), they can appear on the same line, because they can be considered as a single compound syntax element.

Each line starts with a dotted decimal number; for example, 3 or 3.1 or 3.1.1. To hear these numbers correctly, make sure that your screen reader is set to read out punctuation. All the syntax elements that have the same dotted decimal number (for example, all the syntax elements that have the number 3.1) are mutually exclusive alternatives. If you hear the lines 3.1 USERID and 3.1 SYSTEMID, you know that your syntax can include either USERID or SYSTEMID, but not both.

The dotted decimal numbering level denotes the level of nesting. For example, if a syntax element with dotted decimal number 3 is followed by a series of syntax elements with dotted decimal number 3.1, all the syntax elements numbered 3.1 are subordinate to the syntax element numbered 3.

Certain words and symbols are used next to the dotted decimal numbers to add information about the syntax elements. Occasionally, these words and symbols might occur at the beginning of the element itself. For ease of identification, if the word or symbol is a part of the syntax element, it is preceded by the backslash (\) character. The * symbol can be used next to a dotted decimal number to indicate that the syntax element repeats. For example, syntax element *FILE with dotted decimal number 3 is given the format 3 * FILE. Format 3* FILE indicates that syntax element FILE repeats. Format 3* * FILE indicates that syntax element * FILE repeats.

Characters such as commas, which are used to separate a string of syntax elements, are shown in the syntax just before the items they separate. These characters can appear on the same line as each item, or on a separate line with the same dotted decimal number as the relevant items. The line can also show another symbol giving information about the syntax elements. For example, the lines 5.1*, 5.1 LASTRUN, and 5.1 DELETE mean that if you use more than one of the LASTRUN and DELETE syntax elements, the elements must be separated by a comma. If no separator is given, assume that you use a blank to separate each syntax element.

If a syntax element is preceded by the % symbol, this indicates a reference that is defined elsewhere. The string following the % symbol is the name of a syntax fragment rather than a literal. For example, the line 2.1 %OP1 means that you should refer to separate syntax fragment OP1.

The following words and symbols are used next to the dotted decimal numbers:

- ? means an optional syntax element. A dotted decimal number followed by the ? symbol indicates that all the syntax elements with a corresponding dotted decimal number, and any subordinate syntax elements, are optional. If there is only one syntax element with a dotted decimal number, the ? symbol is displayed on the same line as the syntax element, (for example 5? NOTIFY). If there is more than one syntax element with a dotted decimal number, the ? symbol is displayed on a line by itself, followed by the syntax elements that are optional. For example, if you hear the lines 5 ?, 5 NOTIFY, and 5 UPDATE, you know that syntax elements NOTIFY and UPDATE are optional; that is, you can choose one or none of them. The ? symbol is equivalent to a bypass line in a railroad diagram.
- ! means a default syntax element. A dotted decimal number followed by the ! symbol and a syntax element indicates that the syntax element is the default option for all syntax elements that share the same dotted decimal number. Only one of the syntax elements that share the same dotted decimal number can specify a ! symbol. For example, if you hear the lines 2? FILE, 2.1! (KEEP), and 2.1 (DELETE), you know that (KEEP) is the default option for the FILE keyword. In this example, if you include the FILE keyword but do not specify an option, default option KEEP will be applied. A default option also applies to the next higher dotted decimal number. In this example, if the FILE keyword is omitted, default FILE(KEEP) is used. However, if you hear the lines 2? FILE, 2.1, 2.1.1!

(KEEP), and 2.1.1 (DELETE), the default option KEEP only applies to the next higher dotted decimal number, 2.1 (which does not have an associated keyword), and does not apply to 2? FILE. Nothing is used if the keyword FILE is omitted.

- * means a syntax element that can be repeated 0 or more times. A dotted decimal number followed by the * symbol indicates that this syntax element can be used zero or more times; that is, it is optional and can be repeated. For example, if you hear the line 5.1* data area, you know that you can include one data area, more than one data area, or no data area. If you hear the lines 3*, 3 HOST, and 3 STATE, you know that you can include HOST, STATE, both together, or nothing.

  Note:

  1. If a dotted decimal number has an asterisk (*) next to it and there is only one item with that dotted decimal number, you can repeat that same item more than once.
  2. If a dotted decimal number has an asterisk next to it and several items have that dotted decimal number, you can use more than one item from the list, but you cannot use the items more than once each. In the previous example, you could write HOST STATE, but you could not write HOST HOST.
  3. The * symbol is equivalent to a loop-back line in a railroad syntax diagram.
- + means a syntax element that must be included one or more times. A dotted decimal number followed by the + symbol indicates that this syntax element must be included one or more times; that is, it must be included at least once and can be repeated. For example, if you hear the line 6.1+ data area, you must include at least one data area. If you hear the lines 2+, 2 HOST, and 2 STATE, you know that you must include HOST, STATE, or both. Similar to the * symbol, the + symbol can only repeat a particular item if it is the only item with that dotted decimal number. The + symbol, like the * symbol, is equivalent to a loop-back line in a railroad syntax diagram.

Notices

This information was developed for products and services offered in the U.S.A. or elsewhere.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

Policy for unsupported hardware

Various z/OS elements, such as DFSMS, HCD, JES2, JES3, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

Minimum supported hardware

The minimum supported hardware for z/OS releases identified in z/OS announcements can subsequently change when service for particular servers or devices is withdrawn. Likewise, the levels of other software products supported on a particular release of z/OS are subject to the service support lifecycle of those products. Therefore, z/OS and its product publications (for example, panels, samples, messages, and product documentation) can include references to hardware and software that is no longer supported.

- For information about software support lifecycle, see: http://www.ibm.com/software/support/systemsz/lifecycle/
- For information about currently-supported IBM hardware, contact your IBM representative.

Trademarks

Index

A

abend
- finding the module 163
- flowchart 158
- hardware-detected 155
- message text 156
- obtaining the reason code 159
- overview 155
- searching databases 166
- software-detected 155
- symptoms 156

abend analysis
- process 158

abend code
- obtaining 159
- steps for obtaining 159
- VERBEXIT MTRACE 162
- where to find 160

abend example
- determining the dump type 171
- dump title 163
- finding the abend code 160, 172
- finding the job name 170
- finding the program name 170
- free-format search argument 167
- identifying the module prefix 166
- obtaining the dump title 163
- RSMDATA SUMMARY report 173
- searching for multiple problems 170
- SELECT output 170
- STATUS CPU 165
- STATUS FAILDATA 160
- STATUS SYSTEM 171
- STATUS WORKSHEET 163
- SUMMARY TCBERROR 170
- SYSTRACE output 168
- variable recording area 174
- VERBEXIT MTRACE output 172
- VERBX LOGDATA 161
- VERBX MTRACE 162
- VERBX SYMPTOM 162

abend task
- abend code, obtaining
  - steps 159
- component, identifying
  - steps 164
- data, collecting
  - steps 171
- data, gathering
  - steps 168
- databases, searching
  - steps 167
- logrec, collecting
  - steps 169
- messages, collecting
  - steps 169, 171
- module, identifying
  - steps 164
- trace data, gathering
  - steps 168

ABEND0F4
- RLS 278

abnormal wait 200

accessibility 297
- contact IBM 297
- features 297

address
- where to find 164

address space
- status 196

AIRH187E
- tracked jobs exception report 110, 140

AIRSHREP.sh
- install script 73

AMATERSE 28

Assist On-Site 29

assistive technologies 297

ATR248E
- RRS hang symptom 253

ATR249E
- RRS hang symptom 253

authorization
- UPDATE 84

automatic problem reporting 289

automatic spin loop 213

automation
- COMMNDxx for HZR 37
- Runtime Diagnostics 37

B

best practices
- stand-alone dump 25

BLOCKER
- Runtime Diagnostics 42

bpxmtext 16
- example 16

C

catalog
- commands
  - problem determination 239

catalog component
- problem determination 239

CBFORMAT CVT
- example 14

check
- common storage 89
- enqueue request rate 97
- JES spool usage 107
- Logrec dump rate 113
- message arrival rate 120
- SMF arrival rate 136

checks
- RRS 251

coded wait state
- guidance for diagnosis 157

collecting data
- performance problem 228

collecting data *(continued)*
  using JES2 commands 230
collecting documentation
  RRS 251
command
  no system response 203
commands
  differences 79
  for PFA 79
  IBM Health Checker for z/OS 79
  pfa,display 81
  pfa,modify 84
  Predictive Failure Analysis (PFA) 81, 84
  Runtime Diagnostics dump 46
  update 84
common storage
  output report 92
communication
  console
    locked out 203
component
  diagnosis 239
component diagnosis
  RRS 239
component owner
  identifying 5
compressing
  dump data sets 28
console
  disabled loop
    locked out 203
  locked out 203
contention
  Runtime Diagnostics 39, 40, 177
control interval (CI)
  splits 27
COPYDUMP
  syntax 28
  using 27
CPU
  analysis
    Runtime Diagnostics 35
CPU analysis
  Runtime Diagnostics 40
creating PFA directories
  example 76
creep
  definition 90
critical messages
  analysis
    Runtime Diagnostics 35
CSA + SQA
  output 92
CSECT 208
CTIRRSxx
  creating 252
CTRACE
  RRS 252

D

DAE 10
data set allocation 219
data set output
  diagnosing 219
database
  for problem reporting 12
databases
  abend symptoms 166
  problem reporting 8
debug options 47
  HZR 47
  SVC dump 47
  target system home 47
  target system not equal to home 47
description
  RRS 251
detection Loop 41
determining a loop 41
Determining hardcopy settings 59
diagnose
  component problem 239
diagnosing
  job or subsystem hang
    steps 193
diagnosis
  best practices 25
  diagnosis 177
  hang 177
  information for elements 285
  loop 177, 203
  output problem 217
  overview xi
  performance problem 227
  Runtime Diagnostics 177
  starting xi
  steps 4
  synopsis xi
  system hang 177
  wait 177
diagnosis data
  gathering 5
diagnostic worksheet 186
Diagnostics report 41
DISABLED FOR PER 207
disabled loop
  diagnosing 208
disabled wait 178
disabled wait state 186
DISPLAY
  GRS,CONTENTION 239
  RRS 251
  STATUS 81
DISPLAY command
  examples 81
documentation
  where to find 23
dummy wait 178
dump
  for loop 207
  RRS 251
  RRS and system logger 251
  Runtime Diagnostics 46
dump directory
  defining 27
dump title
  obtain the title 163
dumps
  compressing 28
dynamic exits 221

E
enabled loop
    diagnosing 209
enabled wait 178
enabling
    Runtime Diagnostics 36
encrypting dump data 28
ENQ
    analysis
        Runtime Diagnostics 35
    contention
        Runtime Diagnostics 39
    determining contention 40
    IBM-supplied address spaces 40
ENQ BLOCKER Runtime Diagnostics 41
ENQ WAITER Runtime Diagnostics 41
enqueue request rate
    high 101
ENQUEUE request rate
    prediction report 101
event HIGHCPU 59
events
    Runtime Diagnostics 38
example 6
    ANALYZE RESOURCE 196
    ASCB control block 197
    bpxmtext 16
    CBFORMAT srb 197
    creating PFA directories 76
    finding the abend address 165
    finding the abend code 162
    free-format search argument 167
    hang
        abend during NIP 183
        CBFORMAT CVT 179
        IEE844W 187
        STATUS CPU 188
    identifying control 199
    identifying the module 166
    IEA794I 5
    IEA911E 5
    IEA995I 156
    IPCS STATUS WORKSHEET 14
    ISMF ABEND PANEL 156
    ISPF abend panel 157
    loop
        VERBEXIT LOGDATA output 214
        VERBEXIT LOGDATA Output 212
        VRA 213
    LSE 200
    message indicating an abend 156
    mount point 76
    mount points 76
    MVS Diagnostic Worksheet 214
    program check interruptions 165
    program name 170
    RB 199
    RSMDATA SUMMARY report 173
    Runtime Diagnostics
        debugging 59
    SELECT output 170
    spin loop messages 212
    STATUS CPU REGISTERS 165
    subsystem hang 193
    SUMMARY FORMAT 198
    SUMMARY FORMAT JOBNAME 196
    SUMMARY TCBERROR 170
example *(continued)*
    SVC dump request 156
    SVC dumps 5
    symbolic link (symlink) 76
    SYSTEM status 171
    SYSTEM TRACE TABLE 209
    TSO/E abend message 157
    VERBEXIT MTRACE output 172
    VERBEXIT SYMPTOM 162
    VERBX SRMDATA report 200
    VRA 174
    zFS attributes 76
example high CPU 48
examples
    DISPLAY 81
    UPDATE 84
exception handling
    Predictive Failure Analysis (PFA) 67
excessive spin
    diagnosing 212
excessive spin loop
    steps for diagnosing 212
    time out 204
EXSPATxx 205
external symptoms 5
extracting
    search argument 9

F
failure message for Runtime Diagnostics
    log stream 39
fast tape 26
Flash 12
    URL 23
flowchart
    abend analysis 158
    job or subsystem hang 195
    loop 206
FMID (function management identifier)
    for operating system 14
format
    for symptoms 9
free-format search
    building 9

H
hang
    dispatchable TCB or SRB 200
    during IPL
        steps 182
    job not running 200
    logrec, gathering
        steps 190
    messages, gathering
        steps 190
    overview 177
    problem description, collecting
        steps 181
    problem reporting, searching
        steps 188
    symptom 177
    system trace activity 200
    wait states during IPL 182

hang example
  ABEND issued 184
  ANALYZE EXCEPTION 184
  CBFORMAT 185
  COPYCAPD 190
  COUPLE SYSPLEX EXCEPTION 188
  CP disabled coded wait 181
  CP no work wait 181
  IPL statistics 182
  IPLDATA STATUS 182
  MVS diagnostic worksheet 186
  NIP statistics 183
  SELECT ALL 183
  STATUS CPU 179, 184
  wait 183, 184
  wait states during IPL 182
hardware-detected
  abend 155
HIGHCPU event 59
HIGHCPU report 41
Hot Topics 23
how PFA works 65
HZR
  Runtime Diagnostics 45
HZR integration with PFA 67
HZROUT 37

I

IBM Health Checker for z/OS
  for PFA commands 79
  RRS 251
IBM Service
  instructions Runtime Diagnostics 46
IBMLink 12
ICH408I
  insufficient access authority
    Runtime Diagnostics 39
identifying
  system problems 5
IEA611I 6
IEADMCxx
  RRS 251
IEE844W 187
incomplete output 220
Index trap
  VSAM 271
information
  where to find more 23
input
  analyzing 220
install script
  AIRSHREP.sh 73
installation exits
  analyzing 221
installing
  PFA 71
  shared file system 75
installing PFA 73
introduction
  problem determination 3
IPCS
  COPYDUMP 27
  performance 27
IPCS STATUS WORKSHEET
  example 14
IPCS subcommand
  by problem 172
IPL
  hang 182
  information 13
ISGGWAIT 199
ISMF
  ABEND PANEL 156
ISPF
  abend panel 157
IXC246E 45
IXC431I 45
IXC467I 45
IXC585E 45
IXG231I
  log stream connectivity 39
IXL013I 45

J

Java
  level 71, 76
Java 6.0
  example 79
Java path
  example 77
  updating 76
JCL
  for dumps 30
JES spool usage
  prediction report 110
JES2
  collecting data 230
job
  dispatchability 196
job hang
  overview 193
  symptom 193
job name
  finding 196
  Runtime Diagnostics 37
job or subsystem hang
  gathering data 195
    step 195
job querying
  analysis
    Runtime Diagnostics 35
job status
  determine 195
  waiting for resources 196

K

keyboard
  navigation 297
  PF keys 297
  shortcut keys 297
keyword list 10
known problem 10

L

latch contention
  Runtime Diagnostics 42
leak
  definition 90

local lock suspension
  Runtime Diagnostics 41
Local lock suspension 41
lock
  Runtime Diagnostics 41
lock suspension, Local 41
log stream
  failure for Runtime Diagnostics 39
  failure message 39
logrec
  analyzing for loop 214
Logrec
  abend example 157
LOGREC
  prediction report 116
loop
  analyzing interactively 210
  analyzing logrec 214
    steps 214
  analyzing printed output 210
  determining an enabled loop 208
  determining type 207
  diagnosis 203
  disabled
    definition 203
    diagnosing 208
    symptom 203
  enabled
    definition 203
    diagnosing 209
    symptom 203
  flowchart 206
  gathering data 206
  Runtime Diagnostics 41
  spin 212
    definition 203
    symptom 204
  steps for diagnosis 205
  TCB
    Runtime Diagnostics 40
Loop detection 41
loop example
  determining a loop 213
  DUMP command 207
  finding an excessive spin loop 213
  finding CSECT 208
  finding the dump title 214
  finding the module name 211
  reading system trace 209
  STATUS CPU 207, 208, 211
  SYSTRACE ALL 209
  viewing the FRR stack 214
  WHERE 210
LOOP report 41

M

managing PFA
  command 79
message analysis
  Runtime Diagnostics 39
message arrival rate
  prediction report 125
messages
  HZR 45
  Runtime Diagnostics 38

messages *(continued)*
  Runtime Diagnostics analysis
    BPX prefix 51
    IEA prefix 52
    IEE prefix 53
    IOS prefix 54
    IRA prefix 54
    IXC prefix 55
    IXL prefix 57
missing
  data set output 219
MODIFY
  CATALOG 239
module
  identify 164
  identifying 5
multiple release levels
  Runtime Diagnostics 38
multiprocessor 14
MVS diagnostic worksheet 186
MVS Diagnostic Worksheet 163
MVS symptom 10

N

navigation
  keyboard 297
no work wait 178
Notices 301

O

OnTop 29
operating system
  level
    determining 13
  name
    determining 13
OPERLOG
  assigning hardcopy 59
  overview 59
  Runtime Diagnostics 36, 39
  setting up
    steps 59
output
  analyzing 220
  using the message prefix 221
output problem
  data set allocation 219
  determine path 224
  diagnosis 217
  installation exit 221
  teleprocessing 224
output problem example
  identifying the component 221
  message prefix for identification 221
  obtaining the program name 221
  viewing an abend 222
output problems
  logrec 223
  messages 223
  reporting 225
output report
  PFA_COMMON_STORAGE_USAGE 92

output task
  additional data, gathering
    steps 223
  data set allocation, analyzing
    steps 219
  data, collecting
    overview 219
    step 219
  inputs and outputs, analyzing
    steps 220
  installation exits, analyzing
    steps 221
  problem reporting, searching
    steps 223
  problems, analyzing
    overview 217
  problems, reporting
    information 225
  program or component, identifying
    steps 221
overview
  enabled wait state 177
  hang 177
  job hang 193
  job or subsystem hang 193
  loop 203
  performance problem 227
  problem resolution 3

P

PDSE
  specific problems 244
PDSE component
  problem determination 243
performance
  cause of decline 230
performance problem
  diagnosis 227
  overview 227
performance problem example
  determining problem source 229
  displaying the contention report 231
  obtaining the program name 232
  requesting an SVC dump 234
  viewing a SLIP summary 229
  viewing held jobs 230
  viewing printer status 231
  viewing task status 230
performance task
  additional data, collecting
    steps 228
  additional data, gathering
    steps 233
  additional JES2 data, gathering
    steps 230
  dump, analyzing
    steps 234
  problem databases, reporting
    steps 232
  problems, reporting
    steps 235
PFA
  how it works 65
  Java level 71
  managing 79
PFA DISPLAY command 81
PFA MODIFY command 84
PFA_COMMON_STORAGE_USAGE
  check 89
  creep 90
  creep definition 90
  leak 90
  leak definition 90
  output report 92
  spike 90
  spike definition 90
PFA_ENQUEUE_REQUEST_RATE
  check 97
  parameters 104
PFA_JES_SPOOL_USAGE
  check 107
  parameters 111
PFA_LOGREC_ARRIVAL_RATE
  check 113
PFA_MESSAGE_ARRIVAL_RATE
  check 120
  parameters 132
PFA_SMF_ARRIVAL_RATE
  check 136
  parameters 148
PR/SM (Processor Resource/Systems Manager)
  determine use 14
prediction file
  PFA_COMMON_STORAGE_USAGE
    .csv format 95
    .csv values 95
  values 118
prediction report
  ENQUEUE request rate 101
  JES spool usage 110
  LOGREC 116
  message arrival rate 125
  SMF arrival rate 140
predictive failure
  PFA_COMMON_STORAGE_USAGE 89
  PFA_ENQUEUE_REQUEST_RATE 97
  PFA_JES_SPOOL_USAGE 107
  PFA_LOGREC_ARRIVAL_RATE 113
  PFA_MESSAGE_ARRIVAL_RATE 120
  PFA_SMF_ARRIVAL_RATE 136
Predictive Failure Analysis (PFA)
  components 66
  example
    typical check 66
  exception handling 66, 67
  how to update 84
  installing 71
  overview 65
  pfa,display 81
  pfa,modify 84
  prerequisites 71
  Runtime Diagnostics reports 67
problem
  categories 6
problem data
  checklist 289
  collecting 3
  report to IBM 289
problem determination
  ASIDs consuming excessive storage below 2GB 268
  catalog commands 239
  damaged or broken catalogs 240
  hang in the Catalog address space 239

problem determination *(continued)*
  hang in VSAM record management code 273
  HANG/WAIT in RLS/TVS 279
  IGW038A message 245
  introduction to 3
  Invalid Track Format error on a volume 266
  issues opening, closing or extending VSAM data sets 275
  loop in VSAM record management code 273
  message ANTX5104E 267
  microcode issue impacting concurrent copy 269
  mismatch of storage control sessions 269
  No Record Found error on a volume 266
  PDSE ABEND 0F4 failures 244
  PDSE data set corruption 247
  RRS 251
  RRS commands 251
  Share Control Datasets not specified 280
  slow performance for catalog address space 241
  SMSPDSE address space failure 249
  SMSPDSE1 address space failure 249
  SMSVSAM will not start up 280
  StorageControlTimeout 267
  unable to determine PPRC paths 265
  unexpected return codes from VSAM record management 274
problem determination, catalog component 239
problem determination, PDSE component 243
problem determination, System Data Mover (SDM) component 265
problem determination, VSAM component 271
problem determination, VSAM record-level sharing (RLS) component 277
problem diagnostic worksheet 291
problem indicators
  overview 6
  table 7
problem reporting
  additional data for output 223
  automatic 289
  databases 8
  output problems 223
  to IBM 289
problem reporting database
  search 12
problem resolution
  overview 3
problem source identification
  definition 3
  introduction to 3
problem type
  indicators 7
processing
  UPDATE 84
processor
  example 14
processors
  How many are active? 14
profile
  Runtime Diagnostics 37
program check 183
program check interruptions
  example 165
program name 170
PSI
  definition 3

Q

qualified success
  Runtime Diagnostics 39

R

real storage manager
  usage 173
reason codes
  determining meaning 16
  z/OS UNIX 16
remote access 29
remote checks
  PFA_COMMON_STORAGE_USAGE 89
  PFA_ENQUEUE_REQUEST_RATE 97
  PFA_JES_SPOOL_USAGE 107
  PFA_LOGREC_ARRIVAL_RATE 113
  PFA_MESSAGE_ARRIVAL_RATE 120
  PFA_SMF_ARRIVAL_RATE 136
report HIGHCPU 41
report LOOP 41
reporting example
  finding the system level 14
  number of active processors 14
reports
  Runtime Diagnostics 38
requirements Runtime Diagnostics 36
resource contention
  performance 231
Resource Recovery Services (RRS) 251
restrictions
  UPDATE 84
RETAIN symptom 9
return codes
  Runtime Diagnostics 45
RMF
  analyze output 231
RRS
  collecting documentation 251
  commands
    problem determination 251
  CTRACE options 252
  description 251
  dump 251
  problem determination 251
RSMDATA SUMMARY report
  example 173
RSVSF 29
Runtime Diagnostics
  contention 39
  CPU analysis 40
  critical message analysis
    example 44
  ENQ contention report 40
  event
    example 38
    explanation 38
  F, HZR 46
  guidelines 36
  high CPU and lock message 41
  HZR
    user ID 37
  HZRINIT 37
  JOBNAME paramater 37
  local lock suspension 41
  log stream 39

Runtime Diagnostics *(continued)*
  loop 41
  message
    log stream failure 39
  message analysis 39
  mixed release 38
  OPERLOG 39
  output 38
  overview 35
  partial analysis
    system 45
  Predictive Failure Analysis (PFA) 67
  PROC statement 38
  profile 37
  requirements 36
  return codes 45
  security 37
  STARTED class 37
  status message 38
  SYS1.PROCLIB data set 36
  target system 45
  target system analysis
    example 46
  tasks 35
  understanding messages 45
  understanding report messages 38
  using 36
  z/OS UNIX
    latch contention 42
Runtime Diagnostics LOOP and HIGHCPU report 41
Runtime Diagnostics proc
  SYS1.PROCLIB 36

S

SDM
  specific problems 265
search
  problem reporting database 12
search argument 8
  developing 9
  stand-alone dump 9
  SVC dump 9
  symptoms 10
searching
  loop problems 215
  problem reporting databases 8
sending comments to IBM xiii
serialization summary 40
shared file system
  installing 75
shortcut keys 297
SLIP
  setting for abend 171
SMF
  analyze output 231
SMF arrival rate
  prediction report 140
soft failure
  definition 35
soft failures
  description 65
software-detected
  abend 155
spike
  definition 90
spin loop
  diagnosing 212
SRB status
  examining 197
stand-alone dump
  automation 30
  creating 27
  dump directory 27
  example 31
  fast tape 26
  guidelines 26
  merging 27
  performance 26
  planning 26
  sample jobs 30
  using IPCS 27
standardized symptom keyword list 10
STARTED class Runtime Diagnostics 37
STATUS SYSTEM 171
status update 188
STATUS WORKSHEET
  example 163
subsystem hang
  symptom 193
subsystem Runtime Diagnostics 37
suspension, Local lock 41
SVC
  dump message 156
SVC dump
  job or subsystem hang 195
symbolic link (symlink)
  example 76
symlink 76
symptom
  classifications 3
  disabled loop 203
  enabled loop 203
  free format 9
  hang 177
  in search argument 10
  MVS format 9
  RETAIN format 9
  spin loop 204
  wait 177
symptoms
  abend 156
  search argument 8
  selecting 9
syntax
  HZR DEBUG options 46
sysplex
  system partitioned 188
System Data Mover (SDM) component
  problem determination 265
system hang
  diagnosis 177
system level
  determining 14
system partitioned 188
system problem
  identification 5
  what to collect 5
system trace
  analyzing for abend 168
SYSTEM TRACE TABLE 209
SYSTRACE
  wait entries 200

T

task
  abend
    overview 155
tasks
  hang, diagnosing
    steps 177, 179
  loop, diagnosing
    steps 41
  resource contention, checking
    steps 231
TB status
  examining 198
TCB loop
  Runtime Diagnostics 35
tech notes
  URL 23
techdocs
  URL 23
Technical help database 12
technotes
  URL 23
Technotes 12
testing
  dump operations 29
tip
  query trace 5
  routing information 4
tools
  IBM Health Checker for z/OS 3
trace
  query 5
trace data
  gathering for abends 168
trademarks 303
TRSMAIN 28
TSO/E abend message
  example 157

U

Understanding PFA commands 79
uniprocessor 14
UPDATE
  examples 84
  EXCLUDED_JOBS
    example 84
usage
  UPDATE 84
user ID
  Runtime Diagnostics 37
user interface
  ISPF 297
  TSO/E 297

V

VERBEXIT MTRACE output 172
VERBEXIT SYMPTOM
  example 162
  subcommands 172
VSAM
  Index trap 271
  problems 271
VSAM component
  problem determination 271
VSAM record-level sharing (RLS)
  problems 278
VSAM record-level sharing (RLS) component
  problem determination 277

W

wait
  coded 178
  disabled 178
    non-restartable 178
    restartable 178
  enabled 178
    causes 178
    definition 178
    dummy 178
    no work wait 178
  missing interrupt 178
  non-restartable 178
  restartable 178
  symptom 177
wait state
  coded
    diagnosing 186
WAITER
  Runtime Diagnostics 42
waiters
  list of job names 40
worksheet
  reporting problems 291

Z

z/OS UNIX
  reason codes 16

Product Number: 5650-ZOS

Printed in USA

SC23-6844-01