Linux on IBM Z and IBM LinuxONE

Using the Dump Tools on SUSE Linux Enterprise Server 15 SP5

SC34-2785-01

Note

Before using this information and the product it supports, read the information in “Notices” on page 85.

This edition applies to SUSE Linux® Enterprise Server 15 SP5 on IBM Z®, and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

What's new for SUSE Linux Enterprise Server 15 SP5

This edition (SC34-2785-01) contains the following changes compared to SUSE Linux Enterprise Server 15.

New information

- This edition includes dump information for KVM guests.
- You can now use NVMe disks as dump devices, see Chapter 7, “Using an NVMe disk dump device,” on page 39.
- For SCSI dump devices, this edition describes DPM partition dump and using the HMC API through the **zhmc** command. See “DPM partition example” on page 35 and “zhmc - Triggering dumps remotely through the HMC Web Services API” on page 79.
- With a new `--key` option, the **zgetdump** command can now handle the encrypted dumps of KVM guests in IBM® Secure Execution mode, see “zgetdump - Copy and convert kernel dumps” on page 66.

Changed Information

- None.

This revision also includes 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.

Deleted Information

- The **snipl** command is obsolete. Information about the command has been removed.

About this publication

This publication describes tools for obtaining dumps of SUSE Linux Enterprise Server 15 SP5 instances running on IBM Z or LinuxONE hardware.

Unless stated otherwise, all IBM z/VM® related information in this document assumes a current z/VM version, see www.vm.ibm.com/techinfo/lpmigr/vmleos.html.

For SUSE Linux Enterprise Server product documentation, see www.suse.com/documentation.

You can find the latest version of this document at ibm.com/docs/en/linux-on-systems?topic=overview-using-dump-tools

Authority

Most of the tasks described in this document require a user with root authority. In particular, writing to most of the described sysfs attributes requires root authority.

Throughout this document, it is assumed that you have root authority.

Note on device nodes

In this publication, the examples for DASD, tape, and NVMe devices use standard device nodes. For such nodes, the mapping of node name to physical storage space does not persist across reboots or reloading of the applicable device driver. In production environments, use persistent udev-created device nodes.

The SCSI examples in this publication use multipath device nodes.

Other relevant Linux on IBM Z publications

Go to IBM Documentation for Linux on IBM Z publications about SUSE Linux Enterprise Server 15 SP5.

For the latest version of *Device Drivers, Features, and Commands on SUSE Linux Enterprise Server 15 SP5*, SC34-2784, go to IBM Documentation at ibm.com/docs/en/linux-on-systems?topic=distributions-suse-linux-enterprise-server

These publications are available on IBM Documentation at ibm.com/docs/en/linux-on-systems?topic=linuxone-library-overview

- *Device Drivers, Features, and Commands*
- *Using the Dump Tools*
- *How to use FC-attached SCSI devices with Linux on z Systems®*, SC33-8413
- *Networking with RoCE Express*, SC34-7745
- *KVM Virtual Server Management*, SC34-2752
- *Configuring Crypto Express Adapters for KVM Guests*, SC34-7717
- *Introducing IBM Secure Execution for Linux*, SC34-7721
- *openCryptoki - An Open Source Implementation of PKCS #11*, SC34-7730
- *OpenSSL support for Linux on IBM Z and LinuxONE*, SC34-7732
- *libica Programmer's Reference*, SC34-2602
- *libzpc - A Protected-Key Cryptographic Library*, SC34-7731
- *Exploiting Enterprise PKCS #11 using openCryptoki*, SC34-2713
- *Secure Key Solution with the Common Cryptographic Architecture Application Programmer's Guide*, SC33-8294
- *Pervasive Encryption for Data Volumes*, SC34-2782

- *Enterprise Key Management for Pervasive Encryption of Data Volumes*, SC34-7740
- *How to set an AES master key*, SC34-7712
- *Troubleshooting*, SC34-2612
- *Kernel Messages*, SC34-2599
- *How to Improve Performance with PAV*, SC33-8414
- *How to Set up a Terminal Server Environment on z/VM*, SC34-2596

Chapter 1. Planning for dumps

Be prepared before disaster strikes! Consider what dump method you want to use, what size dumps you need to handle, and what possibilities exist to limit the size or spread the dump over several devices.

Tools overview

Different tools can be used for obtaining dumps for instances of SUSE Linux Enterprise Server 15 SP5 running on IBM Z or IBM LinuxONE (LinuxONE) servers.

As of IBM z13®, simultaneous multithreading is available for Linux in LPAR mode. SUSE Linux Enterprise Server 15 SP5 includes dump tools that can create dumps for both Linux instances with and without SMT enablement.

Dump tools from earlier versions of SUSE Linux Enterprise Server are restricted to Linux instances without SMT enablement. Do not use dump disks that were prepared with stand-alone dump tools of earlier versions to create dumps of SMT-enabled Linux instances.

You can use the dump analysis tool **`crash`** to analyze a dump. Depending on your support contract, you might also want to send a dump to IBM support to be analyzed.

Table 1 on page 1 summarizes the available dump tools:

Table 1. Dump tools summary

Dump aspect	kdump	DASD	Multi-volume DASD	SCSI	NVMe	Tape	virsh dump	VMDUMP	Live-system dump with zgetdump
Environment	KVM, z/VM, and LPAR	z/VM and LPAR	z/VM and LPAR	z/VM and LPAR	LPAR	z/VM and LPAR	KVM only	z/VM only	z/VM and LPAR
System size (see note "2" on page 2)	Large	Small	Large	Large	Large	Large	Large	Small	Large
Speed	Fast	Fast	Fast	Fast	Fast	Slow	Fast	Slow	Fast
Medium	Any available medium	ECKD or FBA (see note "3" on page 2) DASD	ECKD DASD	SCSI partition	NVMe disk partition	Tape cartridges	Any available medium	z/VM reader	Any available medium
Compression possible	While writing	No	No	No	No	Yes (see note "2" on page 2)	No	No	No

Table 1. Dump tools summary (continued)

Dump aspect	kdump	DASD	Multi-volume DASD	SCSI	NVMe	Tape	virsh dump	VMDUMP	Live-system dump with zgetdump
Dump filtering possible	While writing	When copying	When copying	When copying	While writing and when copying	When copying	When copying	When copying	No
Disruptive (see note"4" on page 2)	Yes	Yes	Yes	Yes	Yes	Yes	Optional	No	No
Stand-alone	No	Yes	Yes	Yes	Yes	Yes	No	No	No

Note:

1. Dump tools that are available for an LPAR environment are also available for supported storage types in DPM partitions. For information about IPL-initiated dump types supported in DPM partitions, see the DPM documentation.
2. For dump system sizes, see also "Maximum dump size by tool" on page 4.
3. SCSI disks can be emulated as FBA DASDs. This dump method can, therefore, be used for SCSI-only z/VM installations.
4. In this context, disruptive means that the dump process kills a running operating system.

kdump

The kdump tool is made available through a Linux kernel and initial RAM disk that are preloaded in memory, along with a production system.

You do not have to install kdump on a dedicated dump device. The kdump system can access the memory that contains the dump of the production system through a procfs file.

Filtering out extraneous memory pages and compression can take place while the dump is written to persistent storage or transferred over a network. The smaller dump size can significantly reduce the write or transfer time, especially for large production systems.

Because kdump can write dumps through a network, existing file system facilities can be used to prevent multiple dumps from being written to the same storage space. Sharing space for dumps across an enterprise is possible.

Stand-alone tools

Stand-alone tools are installed on a device on which you perform an IPL. Different tools are available depending on the device type.

The following stand-alone dump tools are shipped in the s390-tools package as part of the **`zipl`** package:

- DASD dump tool for dumps on a single DASD device
- Multi-volume DASD dump tool for dumps on a set of ECKD DASD devices
- Tape dump tool for dumps on (channel-attached) tape devices
- SCSI disk dump tool for dumps on SCSI disks

- NVMe disk dump tool for dumps on NVMe disks

You need to install these tools on the *dump device*. A dump device is used to initiate a stand-alone dump by IPL-ing the device. It must have a stand-alone dump tool installed and should provide enough space for the dump. For Linux on z/VM, the dump device must be on subchannel set 0. For Linux in LPAR mode, the device can be on any subchannel set. Linux on KVM does not support these stand-alone dump tools.

Typically, the system operator initiates a dump after a system crash, but you can initiate a dump at any time. To initiate a dump, you must IPL the dump device. This process is destructive, that is, the running Linux operating system is killed. The IPL process writes the system memory to the IPL device (DASD device, channel-attached tape, SCSI disk, or NVMe disk).

You can configure a dump device that is automatically used when a kernel panic occurs. For more information, see "dumpconf - Configure panic or PSW restart action" on page 73.

GRUB 2 usage: You cannot use GRUB 2 to install the stand-alone dump tools. You must use the **zipl** command as described in this document for the dump tools.

For more information about **zipl**, see "zipl - Prepare devices for stand-alone dump" on page 64.

The virsh dump command

The **virsh dump** command is a part of the libvirt package on the KVM host and does not need to be installed separately.

You can use **virsh dump** to create dumps of both running or crashed KVM guests. By default, **virsh dump** pauses a running guest during the dump process, but the guest continues running when the dump is complete. You can change this behavior through command parameters.

Always specify the `--memory-only` option to obtain a dump in valid `elf` format as required by the crash tool.

For information about more options for **virsh dump**, see the **virsh** man page at https://libvirt.org/manpages/virsh.html#dump. See also the **virsh dump** information in *KVM Virtual Server Management*, SC34-2752.

The VMDUMP tool

The **VMDUMP** tool is a part of z/VM and does not need to be installed separately.

Dumping with VMDUMP is not destructive. If you dump an operating Linux instance, the instance continues running after the dump is completed.

VMDUMP can also create dumps for z/VM guests that use z/VM named saved systems (NSS).

VMDUMP has an option for including DCSS content in the dump. Depending on the address range specifications that you use with this option, including DCSS content can considerably increase the dump size. In particular, the dump size can exceed the size of the guest's main memory.

Do not use VMDUMP to dump large z/VM guests; the dump process is very slow. Dumping 1 GB of storage can take up to 15 minutes depending on the used storage server and z/VM version.

For more information about VMDUMP, see *z/VM: CP Commands and Utilities Reference*, SC24-6268.

Live-system dump

You can create a kernel dump from a live system without disruption.

Use the **zgetdump** tool that is shipped with the s390-tools package to create a kernel dump while the Linux system continues running. No dump device must be prepared, because the `/dev/crash` device node is used to create the dump.

Maximum dump size by tool

The dump size depends on the size of the system for which the dump is to be created.

All dump methods require persistent storage space to hold the kernel and user space of this system.

kdump
Initially uses the memory of the Linux instance for which a dump is to be created, and so supports any size. A persistent copy can be written to any medium of sufficient size. While writing, the dump size can be reduced through page filtering and compression.

DASD
Depends on the disk size that is configured on the storage server. For example, you can use:

- 3390 DASD with 45 GB
- 3390 DASD with 1 TB

Multi-volume DASD
Can be up to the combined size of 32 ECKD DASD partitions. With the example DASD sizes of 45 GB and 1 TB:

- 32 x 45 GB ≅ 1.4 TB
- 32 x 1 TB = 32 TB

SCSI
Depends on the capacity of the SCSI disk and which other data it contains. To write a large memory dump to SCSI disk, you can use:

z/VM emulated FBA device that represents a real SCSI disk
FBA disks can be defined with the CP command SET EDEVICE. These disks can be used as single-volume DASD dump disks. The SCSI disk size depends on your storage server setup.

SCSI disk
The SCSI disk size depends on your storage server setup. For SCSI dump partitions greater than 2 TB, you must use the GPT disk layout.

NVMe disk
For NVMe dump partitions greater than 2 TB, you must use the GPT disk layout.

Channel-attached tape
Depends on the tape drive. For example, IBM TotalStorage™ Enterprise Tape System 3592 supports large dumps and also offers hardware compression. For large memory dumps, cartridges with up to 10 TB capacity are available.

virsh dump
Depends on the KVM host setup. The dump is written to a directory in the host file system. Size limits depend on the device that backs up this directory.

VMDUMP
Depends on the available spool space. The slow dump speed can lead to very long dump times for large dumps. Although technically possible, the slow dump speed makes VMDUMP unsuitable for large dumps.

zgetdump live-system dump
The dump can be written to any medium of sufficient size.

See “Reducing dump size” on page 55 for information specific to large dumps.

Dump methods compared

The process for preparing a dump device and obtaining a dump differs for the available dump methods.

Table 2. Comparing the dump methods

Dump aspect	kdump	Stand-alone tools	virsh dump	VMDUMP	Live-system dump with zgetdump
Preparation	Reserve memory with the `crashkernel=` kernel parameter Load the kdump kernel and the initial RAM disk into the memory of the production system. Use **kexec** or `systemctl start kdump`	Write the stand-alone dump tool to the dump device (**zipl**) Define the panic shutdown action (**dumpconf**)	None	Define the panic shutdown action (**dumpconf**)	None
Dump trigger	**Automatic:** Kernel panic **Initiated by operator:** PSW restart	**Automatic:** Kernel panic **Initiated by operator:** IPL of the dump device	**Automatic:** Kernel panic **Initiated by operator:** **virsh dump** invocation	**Automatic:** Kernel panic **Initiated by operator:** z/VM CP **VMDUMP** command	**Initiated by operator:** **zgetdump** invocation
Initial dump space	Memory	Dump device	File system of the KVM host	Spool device	Memory
Accessing the initial dump	Through `/proc/vmcore` from the kdump instance (automatically done by kdump initrd)	Using **zgetdump** from a new Linux instance	Available as a dump file on the KVM host	Using **vmur -c** from a new Linux instance	Through `/dev/crash`
Copying the initial dump to the final dump store (and releasing the initial dump space)	Copied from the kdump instance to any available storage (automatically done by kdump initrd)	Copied from the new Linux instance to any available storage	Copied from the KVM host to any available storage	Copied from the new Linux instance to any available storage	Copied from the current Linux instance to any available storage

Table 2. Comparing the dump methods (continued)

Dump aspect	kdump	Stand-alone tools	virsh dump	VMDUMP	Live-system dump with zgetdump
Optional: Filtering the initial dump	Using `/proc/vmcore` and **makedumpfile** on the kdump instance (automatically done by kdump initrd)	Using **zgetdump** and **makedumpfile** on the new Linux instance	**makedumpfile** on the KVM host	Using **zgetdump** and **makedumpfile** on the new Linux instance	Not recommended

Automatic dumping

You can configure a dump device that is automatically used when a kernel panic occurs.

If you set up kdump, a kernel panic or PSW restart automatically triggers a dump.

For Linux on KVM, you can also use the on_crash element in the virtual server configuration to set up automatic dumping.

For Linux in LPAR mode and Linux on z/VM, you can also use dumpconf to set up automatic dumping for stand-alone tools, or for your backup dump solution. See "dumpconf - Configure panic or PSW restart action" on page 73.

Once you have set up the automation, you can cause a kernel panic to test the configuration.

Testing automatic dump-on-panic

Cause a kernel panic to confirm that your dump configuration is set up to automatically create a dump if a kernel panic occurs.

Before you begin

- You need a Linux instance with active magic sysrequest functions.
- For KVM guests, you can use the **virsh inject-nmi** on the KVM host instead of the magic sysrequest function (see "KVM guest example" on page 12).

Procedure

Crash the kernel with a forced kernel panic.

If your method for triggering the magic sysrequest function is:	Enter:
A command on the 3270 terminal or line-mode terminal on the HMC	`^-c`
A command on the hvc0 terminal device	Ctrl+o c
Writing to procfs	`echo c > /proc/sysrq-trigger`

Note: Ctrl+o means pressing o while holding down the control key.

For more details about the magic sysrequest functions, see the documentation in the Linux source tree at `/usr/src/linux/Documentation/sysrq.txt` (requires installation of the kernel-source package).

Results

The production system crashes and if your automatic dump is set up correctly, the dump process is triggered. In particular, if kdump is set up correctly, the kdump kernel is booted, the dump is created (the default directory is `/var/crash/`), and your production system is rebooted.

Sharing dump devices between different versions of Linux

Do not share dump devices between Linux installations with different major releases.

For example, do not share dump devices between SUSE Linux Enterprise Server 12 and SUSE Linux Enterprise Server 15.

Always use the **zgetdump** command, **makedumpfile** command, and the **crash** utility that are delivered with your latest version of SUSE Linux Enterprise Server.

The latest **crash** utility and **makedumpfile** command can process dumps that are created from Linux instances of the same or in earlier versions of SUSE Linux Enterprise Server.

The latest **zgetdump** command can process dumps that are created with the **zipl** (s390-tools package) version in the same or in earlier versions of SUSE Linux Enterprise Server.

As of SUSE Linux Enterprise Server 15 SP5, **zipl** can prepare disks for dumps of SMT-enabled Linux instances. You can use a dump disk that is prepared with an SMT-enabled **zipl** version for both Linux instances with and without SMT enablement.

Dumps for KVM guests in IBM Secure Execution mode

Dumps for KVM guests in IBM Secure Execution mode require protection from unauthorized inspection.

This protection is given for kdump, because the kdump kernel that has access to the dump also runs in IBM Secure Execution mode on the same KVM virtual server.

For secure guests that are enabled for encrypted dumps, you can also use **virsh dump** to create an encrypted dump file. With the required key, you can then access the file with **zgetdump**, see “Working with dumps of KVM guests in IBM Secure Execution mode” on page 71.

Chapter 2. Using kdump

You can use kdump to create system dumps for instances of SUSE Linux Enterprise Server.

Advantages of kdump

kdump offers these advantages over other dump methods:

- While writing the dump, you can filter out extraneous pages and compress the dump, and so handle large dumps in a short time.
- When writing dumps over a network, you can use existing file system facilities to share dump space among multiple Linux instances without special preparations.

Shortcomings of kdump

kdump has these drawbacks:

- kdump cannot be used for issues that occur before kdump is initialized, for example, for early boot problems. For such problems, use a stand-alone dump tool for Linux on z/VM and for Linux in LPAR mode. For Linux on KVM, use **`virsh dump`** instead.
- kdump is not as reliable as the stand-alone dump tools. For critical systems, you can set up stand-alone dump tools as a backup, in addition to the kdump configuration (see "Failure recovery and backup tools" on page 10). For Linux on KVM, **`virsh dump`** can serve as the backup.
- kdump cannot dump a z/VM named saved system (NSS).
- For production systems that run in LPAR mode, kdump consumes memory (see "Memory consumption" on page 10).

How kdump works on IBM Z

You can set up kdump according to your needs.

With kdump, you do not need to install a dump tool on the storage device that is to hold a future dump. Instead, you use a kdump kernel, a Linux instance that controls the dump process.

The kdump kernel occupies a reserved memory area within the memory of the production system for which it is set up. The reserved memory area is defined with the `crashkernel=` kernel parameter. After the production system is started, the kdump init service loads the kdump kernel and its initial RAM disk (initrd) into the reserved memory area with the **`kexec`** tool.

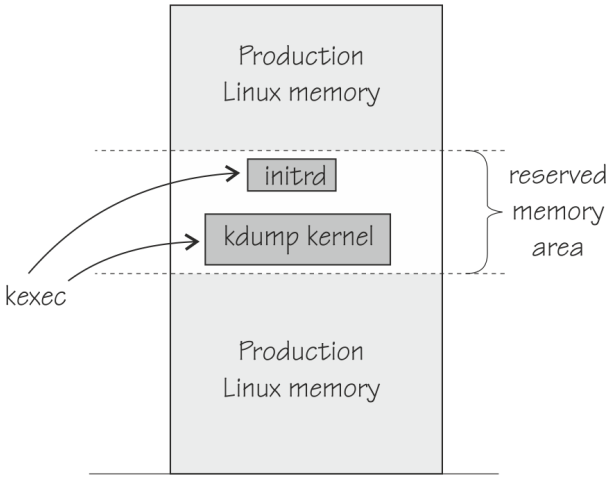


Figure 1. Running production system with preloaded kdump kernel and initial RAM disk

At the beginning of the dump process, the reserved memory area is exchanged with the lower memory regions of the crashed production system. The kdump system is then started and runs entirely in the memory that was exchanged with the reserved area. From the running kdump kernel, the memory of the crashed production system can be accessed as a virtual file, `/proc/vmcore`.

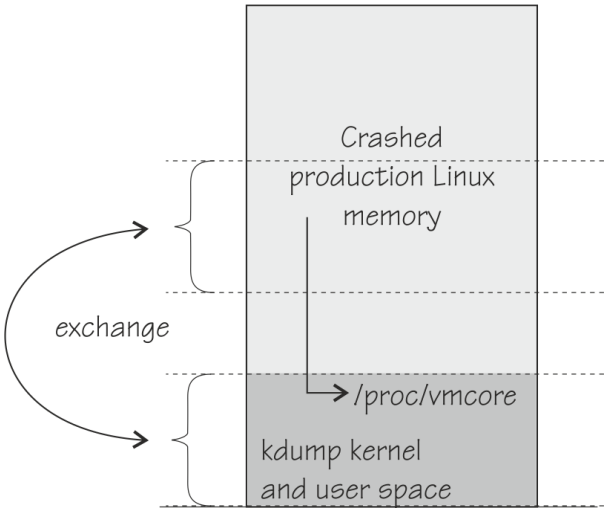


Figure 2. Running kdump kernel

This process is fast because the kdump kernel is started from memory, and no dump data needs to be copied up to this stage. For SUSE Linux Enterprise Server, the **`makedumpfile`** tool in the kdump initrd writes a filtered and compressed version of the dump to a file on persistent storage, locally or over a network. Again, this method saves time because the dump is reduced in size while it is written or transferred.

By default, kdump initrd automatically IPLs the production system after the dump is written.

Memory consumption

Although each Linux instance must be defined with additional memory for kdump, the total memory consumption for your KVM or z/VM installation does not increase considerably.

On most architectures, the inactive kdump system consumes the entire memory that is reserved with the `crashkernel=` kernel parameter.

For Linux on z/VM
: Only the kdump image and its initial RAM disk consume actual memory. The remaining reserved memory is withheld by the z/VM hypervisor until it is required in exchange for the lower memory region of the crashed production system.

For both Linux on z/VM and Linux on KVM
: Because the kdump image and the initial RAM disk are not used during regular operations, the hypervisor swaps them out of memory some time after IPL. Thereafter, no real memory is occupied for kdump until it is booted to handle a dump.

For Linux in LPAR mode, the reserved memory area consumes real memory.

Failure recovery and backup tools

If kdump fails, stand-alone dump tools, virsh dump, or VMDUMP can be used as backup tools. Backup tools are, typically, set up only for vital production systems.

Because of being preloaded into memory, there is a small chance that parts of kdump are overwritten by malfunctioning kernel functions. The kdump kernel is, therefore, booted only if a checksum assures the integrity of the kdump kernel and initial RAM disk. This failure can be recovered automatically by setting up a backup dump tool with the **`dumpconf`** service or through a backup dump that is initiated by a user. See “dumpconf - Configure panic or PSW restart action” on page 73.

A second possible failure is the kdump system itself crashing during the dump process. This failure occurs, for example, if the reserved memory area is too small for the kdump kernel and user space. For this failure, initiate a backup dump, which captures data for both the crashed production system and the crashed kdump kernel. You can separate this data with the **zgetdump --select** option. See "zgetdump - Copy and convert kernel dumps" on page 66.

Setting up kdump

SUSE Linux Enterprise Server 15 SP5 provides two ways of setting up kdump.

About this task

You can choose between the following methods of setting up kdump:

- The kdump configuration utility in YaST: Graphical tool with kdump configuration options.
- Manually, using the configuration file `/etc/sysconfig/kdump`. For a configuration example, see the chapter about kexec and kdump in the *System Analysis and Tuning Guide*, available at

  ```
  www.suse.com/documentation
  ```

What to do next

You can check whether kdump is set up with the lsshut command. If the command lists kdump as a shutdown action, kdump is configured. For example:

```
# lsshut
Trigger          Action
=======================
Halt             stop
Power off        stop
Reboot           reipl
Restart          kdump,stop
Panic            kdump,stop
```

As a backup, you can set up a stand-alone dump tool in addition to kdump. See "dumpconf - Configure panic or PSW restart action" on page 73 about how to run a backup tool automatically, if kdump fails.

Initiating a dump

A kernel panic automatically triggers the dump process with kdump. If this automation fails, there are other methods you can use to trigger the dump process.

About this task

With kdump installed, a kernel panic or PSW restart trigger kdump rather than the shutdown actions defined in `/sys/firmware`. The definitions in `/sys/firmware` are used only if an integrity check for kdump fails (see also "Failure recovery and backup tools" on page 10 and "dumpconf - Configure panic or PSW restart action" on page 73).

To trigger kdump, use one of the methods according to your environment:

- For Linux on KVM: Issue the **virsh inject-nmi** command on the KVM host. See "KVM guest example" on page 12 for an example.
- For Linux on z/VM: Run the z/VM CP **system restart** command. See "z/VM guest example" on page 12 for an example.
- For Linux in LPAR mode: Run the **PSW restart** task on the HMC. See "LPAR example" on page 13 for an example.

In all three environments, you can use the following methods to trigger a dump.

The diag288 watchdog

You can use the diag288 watchdog to trigger kdump. The default setup triggers the correct actions.

The magic sysrequest feature

Crash the kernel through the magic sysrequest feature, for example by writing c to `/proc/sysrq-trigger`.

```
# echo c > /proc/sysrq-trigger
```

Hint: You might have to write 1 to `/proc/sys/kernel/sysrq` first to enable the sysrequest feature.

For more information about the diag288 watchdog and about the magic sysrequest feature, see *Device Drivers, Features, and Commands on SUSE Linux Enterprise Server 15 SP5*, SC34-2784.

Results

After kdump is triggered, first kernel messages from the booting kdump kernel and later dump progress messages are issued. The messages are written to the Operating System Messages applet of the HMC for LPAR, to the 3270 terminal for z/VM, or to the console that has been configured for your KVM virtual server. The kdump scripts copy the dump and reboot automatically.

What to do next

Verify that your production system is up and running again. Send the created dump to your support organization.

KVM guest example

With kdump, you do not need a dump device to initiate the dump.

Before you begin

Your Linux instance must be set up for kdump as described in "Setting up kdump" on page 11.

Procedure

From the KVM host, issue a **`virsh inject-nmi`** command for the virtual server.

For example, to initiate the dump process for a virtual server `vserv3`, issue:

```
# virsh inject-nmi vserv3
```

Depending on the boot configuration, a console is available where boot messages for the kdump kernel indicate that the dump process started.

Results

kdump automatically collects the dump and reboots Linux.

z/VM guest example

With kdump you do not need a dump device to initiate the dump.

Before you begin

Your Linux instance must have been set up for kdump as described in "Setting up kdump" on page 11.

Procedure

Issue the **`system restart`** z/VM CP command, for example from a 3270 terminal emulation for the Linux instance to be dumped:

```
#cp system restart
```

Boot messages for the kdump kernel indicate that the dump process has started.

Results

kdump automatically collects the dump and reboots Linux.

LPAR example

You can initiate a kdump process on an LPAR from an HMC (Hardware Management Console) or SE (Support Element).

About this task

The following description refers to an HMC, but the steps also apply to an SE.

Procedure

1. In the navigation pane of the HMC, expand Systems Management and select the hardware system you want to work with.

   A table of LPARs is displayed in the content area.
2. Select the LPAR for which you want to initiate the dump.
3. In the **Tasks** area, click **PSW restart**, which is located in the **Recovery** section. This initiates the dump process.
   Figure 3 on page 13 shows an example of an HMC with a selected hardware system and LPAR. The **PSW restart** task can be seen in the **Tasks** area.

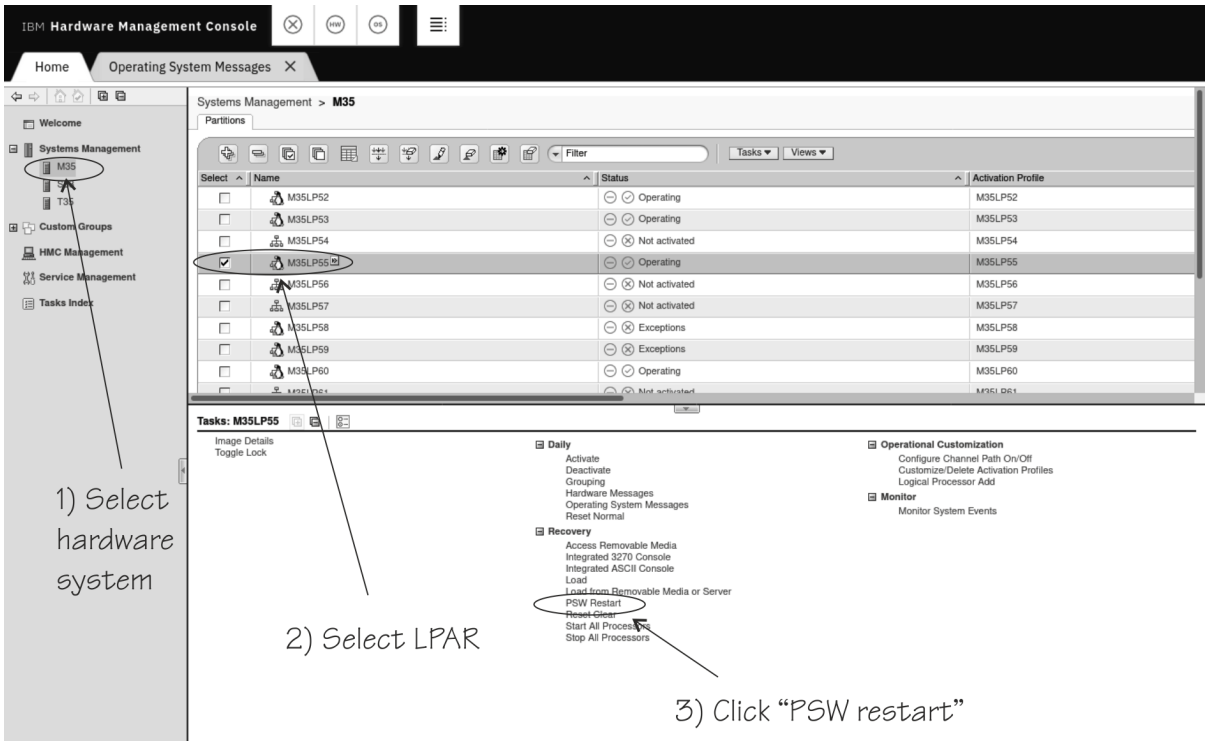


*Figure 3. HMC with the **PSW restart** task*

4. Wait until the dump process completes. Click the **Operating System Messages** icon for progress and error information.

Results

kdump automatically collects the dump and reboots Linux.

Chapter 3. Using a DASD dump device

To use a DASD dump device, you need to install the stand-alone DASD dump tool and perform the dump process. Then, copy the dump to a file in a Linux file system.

DASD dumps are written directly to a DASD partition that is not formatted with a file system. Model 3390 DASDs are supported for both ECKD and FBA.

Installing the DASD dump tool

Install the DASD dump tool on an unused DASD partition. Dumps are written to this partition.

Before you begin

You need an unused DASD partition with enough space (memory size + 10 MB) to hold the system memory. If the system memory exceeds the capacity of a single DASD partition, use the multi-volume dump tool, see Chapter 4, “Using DASD devices for multi-volume dump,” on page 21.

GRUB 2 usage: You cannot use GRUB 2 to install the stand-alone dump tools. You must use the **zipl** command as described in this document for the dump tools.

About this task

The examples in assume that `/dev/dasdc` is the dump device and that you want to dump to the first partition `/dev/dasdc1`.

The steps that you need to perform for installing the DASD dump tool depend on your type of DASD, ECKD or FBA:

- If you are using an ECKD-type DASD, perform all three of the following steps.
- If you are using an FBA-type DASD, skip steps 1 and 2 and perform step 3 only.

Procedure

1. ECKD only: Format your DASD with **dasdfmt**.
   Use a block size of 4 KB.
   For example:

   ```
   # dasdfmt -f /dev/dasdc -b 4096
   ```

2. ECKD only: Create a partition with **fdasd**.
   The partition must be sufficiently large (the memory size + 10 MB).
   For example:

   ```
   # fdasd /dev/dasdc
   ```

3. Install the dump tool with the **zipl** command.
   Specify the dump device on the command line.
   For example:

   ```
   # zipl -d /dev/dasdc1
   ```

   Note: When you use a DASD of type ECKD that is formatted with the traditional Linux disk layout ldl, the dump tool must be reinstalled with **zipl** after each dump.

Initiating a DASD dump

You can initiate a dump from a DASD device.

Procedure

To obtain a dump with the DASD dump tool, perform the following main steps:

1. Stop all CPUs.
2. Store status on the IPL CPU.
3. IPL the dump tool on the IPL CPU.

   Note: Do not clear the main memory.

   The dump process can take several minutes depending on the device type you are using and the amount of system memory. After the dump completes, the IPL CPU should go into disabled wait.

   The following PSW indicates that the dump process completed successfully:

   ```
   (64-bit) PSW: 00020000 80000000 00000000 00000000
   ```

   Any other disabled wait PSW indicates an error.

   After the dump tool is IPLed, messages that indicate the progress of the dump are written to the console:

   ```
   Dumping 64 bit OS
   00000032 / 00000256 MB
   00000064 / 00000256 MB
   00000096 / 00000256 MB
   00000128 / 00000256 MB
   00000160 / 00000256 MB
   00000192 / 00000256 MB
   00000224 / 00000256 MB
   00000256 / 00000256 MB
   Dump successful
   ```

Results

You can IPL Linux again.

z/VM guest example

In this example, a dump to DASD device 193 is initiated from z/VM.

Example

If 193 is the dump device:

```
#cp cpu all stop
#cp store status
#cp i 193
```

On z/VM, a three-processor machine in this example, you will see messages about the disabled wait:

```
01: The virtual machine is placed in CP mode due to a SIGP stop from CPU 00.
02: The virtual machine is placed in CP mode due to a SIGP stop from CPU 00.
"CP entered; disabled wait PSW 00020000 80000000 00000000 00000000"
```

You can now IPL your Linux instance and resume operations.

LPAR example

In this example, a Linux instance running on an LPAR is dumped to a DASD device. The dump is initiated from an HMC or SE.

About this task

The following description refers to an HMC, but the steps also apply to an SE.

Procedure

1. In the navigation pane of the HMC, expand Systems Management and select the hardware system you want to work with.

   A table of LPARs is displayed in the content area.
2. Select the LPAR for which you want to initiate the dump.
3. In the **Tasks** area, click **Stop all processors** (which is located in the **Recovery** section) to stop all CPUs. Confirm when you are prompted to do so.
4. In the **Tasks** area, click **Load** to display the Load panel.
   Figure 4 on page 17 shows an example of an HMC with a selected hardware system and LPAR. The **Load** and **Stop all processors** tasks can be seen in the Tasks area.

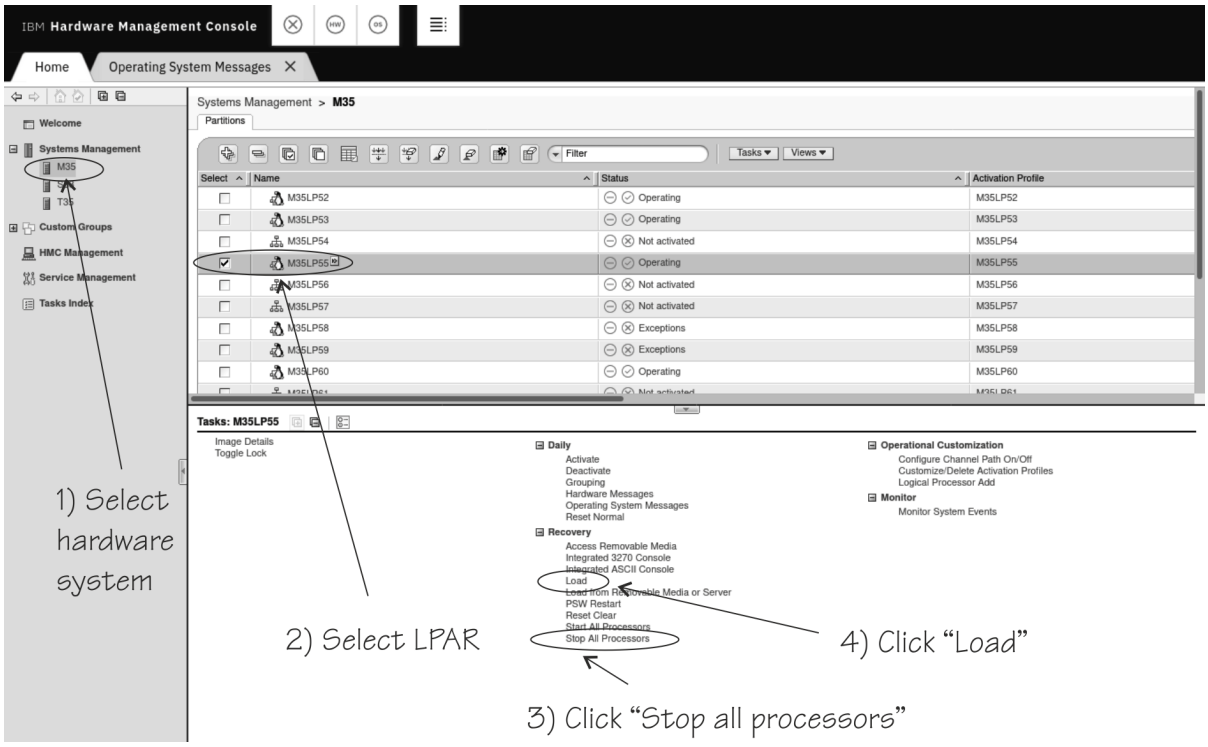


*Figure 4. HMC with the **Load** and **Stop all processors** tasks*

5. Select **Load type** "Standard load".

   Note: You must not select the **Clear the main memory on this partition before loading it** check box. Clearing the memory destroys the dump content.
6. Select the **Store status** check box.
7. Type the device number of the dump device into the **Load address** field.

   To IPL from a subchannel set other than 0, specify five digits: The subchannel set ID followed by the device number, for example 1E711. Figure 5 on page 18 shows a Load panel with all entries and selections required to start the dump process for a DASD or tape dump device.

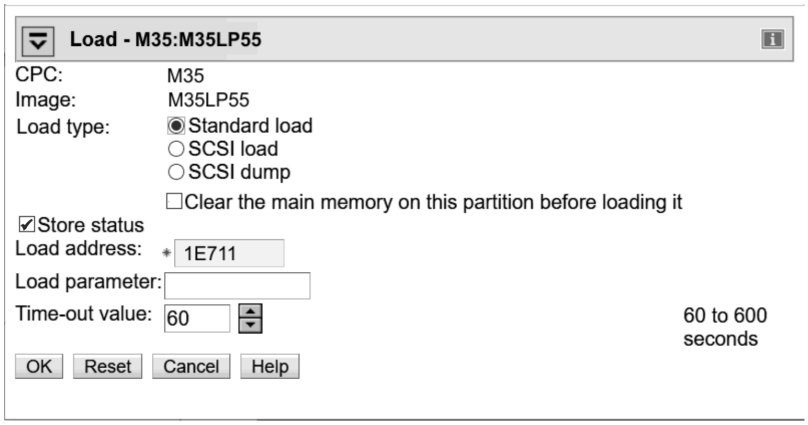


Figure 5. Load panel for dumping to DASD

8. Click **OK** to start the dump process.
9. Wait until the dump process completes. Click the **Operating System Messages** icon for progress and error information.

Results

When the dump has completed successfully for a stand-alone dump tool, you can IPL Linux again.

Automatic dump example

On both z/VM and LPAR, you can use the dumpconf service to set up automatic dumping. In this example, a dump is automatically triggered when a kernel panic occurs.

About this task

Use the dumpconf service to set up automatic dumping. To set up dumping, edit the configuration file `/etc/sysconfig/dumpconf`.

Example

Example configuration for a CCW dump device (DASD):

```
ON_PANIC=dump
DUMP_TYPE=ccw
DEVICE=0.0.4714
```

For details on how to set up dumpconf, see “dumpconf - Configure panic or PSW restart action” on page 73.

Copying a DASD dump to a file

Use the **zgetdump** command to copy a DASD dump to a file in the file system.

About this task

By default, the **zgetdump** tool takes the dump device as input and writes its contents to standard output. To write the dump to a file system, you must redirect the output to a file.

Procedure

Assuming that the dump is on DASD device `/dev/dasdc1` and you want to copy it to a file named `dump.elf`:

```
# zgetdump /dev/dasdc1 > dump.elf
```

What to do next

You can use **zgetdump** to display information about the dump. See “Checking whether a DASD dump is valid and printing the dump header” on page 69 for an example.

For general information about **zgetdump**, see “zgetdump - Copy and convert kernel dumps” on page 66 or the man page.

Chapter 4. Using DASD devices for multi-volume dump

You can handle large dumps, up to the combined size of 32 DASD partitions, by creating dumps across multiple volumes.

Before you begin

You need to prepare a set of ECKD DASD devices for a multi-volume dump, and install the stand-alone dump tool on each DASD device that is involved. Then, perform the dump process, and copy the dump to a file in a Linux file system.

GRUB 2 usage: You cannot use GRUB 2 to install the stand-alone dump tools. You must use the **zipl** command as described in this document for the dump tools.

About this task

You can specify up to 32 partitions on ECKD DASD volumes for a multi-volume dump. The dump tool is installed on each volume involved. The volumes must be:

- In subchannel set 0.
- Formatted with the compatible disk layout (cdl, the default option when using the **dasdfmt** command.)

You must specify block size 4096 for **dasdfmt**.

For example, Figure 6 on page 21 shows three DASD volumes, dasdb, dasdc, and dasdd, with four partitions selected to contain the dump. To earmark the partition for dump, a dump signature is written to each partition.

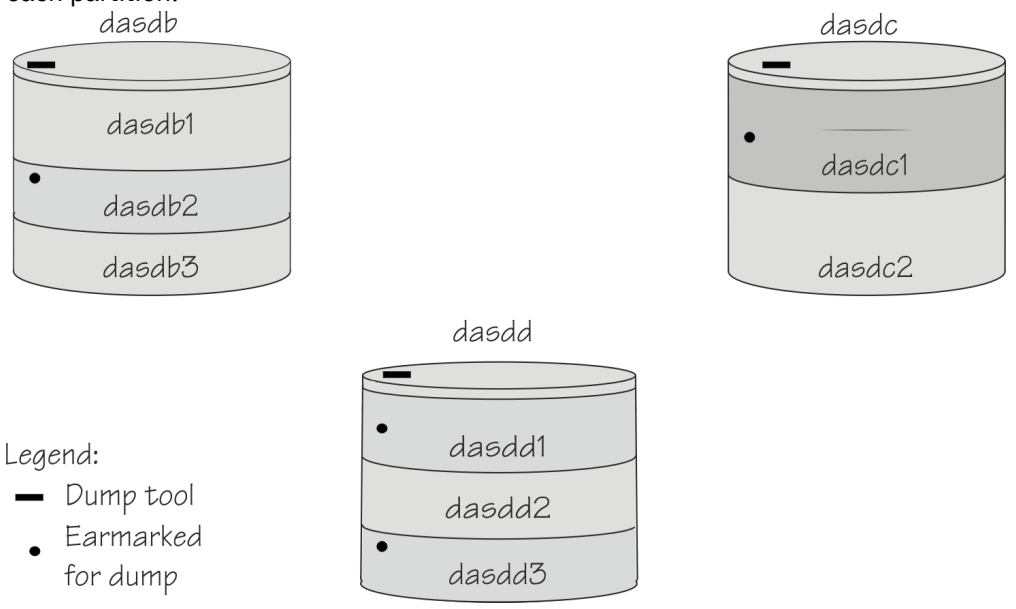


Figure 6. Three DASD volumes with four partitions for a multi-volume dump

The partitions need to be listed in a configuration file, for example:

```
/dev/dasdb2
/dev/dasdc1
/dev/dasdd1
/dev/dasdd3
```

See “Installing the multi-volume DASD dump tool” on page 22 for how to create the configuration file. All three DASD volumes are prepared for IPL; regardless of which you use the result is the same.

The following sections will take you through the entire process of creating a multi-volume dump.

Installing the multi-volume DASD dump tool

This example shows how to perform the dump process on two partitions, `/dev/dasdc1` and `/dev/dasdd1`, which reside on ECKD volumes `/dev/dasdc` and `/dev/dasdd`.

About this task

Assume that the corresponding device bus-IDs (as displayed by **lsdasd**) are 0.0.4711 and 0.1.4712.

Procedure

1. Format both dump volumes with **dasdfmt**.
   The command shown uses the default cdl (compatible disk layout) and specifies a block size of 4KB.

```
# dasdfmt -f /dev/dasdc -b 4096
# dasdfmt -f /dev/dasdd -b 4096
```

2. Create the partitions with **fdasd**.
   The sum of the partition sizes must be sufficiently large (the memory size + 10 MB):

```
# fdasd /dev/dasdc
# fdasd /dev/dasdd
```

3. Create a file named `mvdump.conf` containing the device nodes of the two partitions, separated by one or more line feed characters (0x0a).
   The file's contents are as follows:

```
/dev/dasdc1
/dev/dasdd1
```

4. Prepare the volumes using the **zipl** command.
   Specify the dump list on the command line.

   Command line example:

```
# zipl -M mvdump.conf
 Dump target: 2 partitions with a total size of 1234 MB.
 Warning: All information on the following partitions will be lost!
 /dev/dasdc1
 /dev/dasdd1
 Do you want to continue creating multi-volume dump partitions (y/n)?
```

Results

Now the two volumes `/dev/dasdc` and `/dev/dasdd` with device bus-IDs 0.0.4711 and 0.1.4712 are prepared for a multi-volume dump. Use the **--device** option of **zgetdump** to display information about these volumes:

```
# zgetdump -d /dev/dasdc
Dump device info:
  Dump tool.........: Multi-volume DASD dump tool
  Version...........: 2
  Architecture......: s390x (64 bit)
  Dump size limit...: none
  Force specified...: no

  Volume 0: 0.0.4711 (online/valid)
  Volume 1: 0.1.4712 (online/valid)
```

During **zipl** processing both partitions were earmarked for dump with a valid dump signature. The dump signature ceases to be valid when data other than dump data is written to the partition. For example, writing a file system to the partition overwrites the dump signature. Before writing memory to a partition, the dump tool checks the partition's signature and exits if the signature is invalid. Thus any data inadvertently written to the partition is protected.

You can circumvent this protection, for example, if you want to use a swap space partition for dumping, by using the **zipl** command with the **--force** option. This option inhibits the dump signature check, and any data on the device is overwritten. Exercise great caution when using the force option.

The **zipl** command also takes a size specification, see Appendix A, “Obtaining a dump with limited size,” on page 61. For more details about the **zipl** command, see “zipl - Prepare devices for stand-alone dump” on page 64.

Initiating a multi-volume DASD dump

After preparing the DASD volumes, you can initiate a multi-volume dump by performing an IPL from one of the prepared volumes.

Procedure

To obtain a dump with the multi-volume DASD dump tool, perform the following main steps:

1. Stop all CPUs.
2. Store status on the IPL CPU.
3. IPL the dump tool using one of the prepared volumes, either 0.0.4711 or 0.1.4712. The process for initiating the IPL is the same as for a single volume DASD.

   Note: Do not clear the main memory.

   For an example performed for a single DASD on the HMC, see “LPAR example” on page 17. For an example performed for a single DASD on z/VM, see “z/VM guest example” on page 16.

   The dump process can take several minutes depending on each volume's block size and the amount of system memory. After the dump has completed, the IPL CPU should go into disabled wait.

   The following PSW indicates that the dump process has completed successfully:

   ```
   (64-bit) PSW: 00020000 80000000 00000000 00000000
   ```

   Any other disabled wait PSW indicates an error.

   After the dump tool is IPLed, messages that indicate the progress of the dump are written to the console:

   ```
   Dumping 64 bit OS
   Dumping to: 0.0.4711
   00000128 / 00001024 MB
   00000256 / 00001024 MB
   00000384 / 00001024 MB
   00000512 / 00001024 MB
   Dumping to: 0.1.4712
   00000640 / 00001024 MB
   00000768 / 00001024 MB
   00000896 / 00001024 MB
   00001024 / 00001024 MB
   Dump successful
   ```

Results

You can IPL Linux again.

Copying a multi-volume dump to a file

Use the **zgetdump** command to copy a multi-volume dump to a file in the file system.

About this task

This example assumes that the two volumes /dev/dasdc and /dev/dasdd (with device bus-IDs 0.0.4711 and 0.1.4712) contain the dump. Dump data is spread along partitions /dev/dasdc1 and /dev/dasdd1.

Procedure

Use **zgetdump** without any options to copy the dump parts to a file:

```
# zgetdump /dev/dasdc > dump.elf
Format Info:
Source: s390mv
Target: elf

Copying dump:
00000000 / 00001024 MB
00000171 / 00001024 MB
00000341 / 00001024 MB
00000512 / 00001024 MB
00000683 / 00001024 MB
00000853 / 00001024 MB
00001024 / 00001024 MB

Success: Dump has been copied
```

If you want to only check the validity of the multi-volume dump rather than copying it to a file, use the **--info** option with **zgetdump**. See “Checking whether a DASD dump is valid and printing the dump header” on page 69 for an example.

Chapter 5. Using a tape dump device

You can use a channel-attached tape as a dump device. To use a tape, you need to install the stand-alone tape dump tool and perform the dump process. Then, copy the dump to a file in a Linux file system.

The following tape devices are supported:

- 3480
- 3490
- 3590
- 3592

The following sections take you through the entire process of creating a dump on a tape device.

Installing the tape dump tool

Install the tape dump tool on the tape that is to hold the dump.

Before you begin

Have enough empty tapes ready to hold the system memory (memory size + 10 MB).

GRUB 2 usage: You cannot use GRUB 2 to install the stand-alone dump tools. You must use the **zipl** command as described in this document for the dump tools.

About this task

The examples assume that `/dev/ntibm0` is the tape device that you want to dump to.

Procedure

1. Insert an empty dump cartridge into your tape device.
2. Ensure that the tape is rewound.
3. Install the dump tool by using the **zipl** command.
   Specify the dump device on the command line.
   For example:

```
# zipl -d /dev/ntibm0
```

Initiating a tape dump

Initiate a tape dump by performing an IPL on the IPL CPU.

Procedure

To obtain a dump with the tape dump tool, perform the following main steps:

1. Ensure that the tape is rewound.
2. Stop all CPUs.
3. Store status on the IPL CPU.
4. IPL the dump tool on the IPL CPU.

   Note: Do not clear the main memory.

   The dump tool writes the number of dumped MB to the tape drive message display.

The dump process can take several minutes, depending on the device type you are using and the amount of system memory available. When the dump is complete, the message dump*end is displayed and the IPL CPU goes into disabled wait.

The following PSW indicates that the dump process completed successfully:

```
(64-bit) PSW: 00020000 80000000 00000000 00000000
```

Any other disabled wait PSW indicates an error.

After the dump tool is IPLed, messages that indicate the progress of the dump are written to the console:

```
Dumping 64 bit OS
00000032 / 00000256 MB
00000064 / 00000256 MB
00000096 / 00000256 MB
00000128 / 00000256 MB
00000160 / 00000256 MB
00000192 / 00000256 MB
00000224 / 00000256 MB
00000256 / 00000256 MB
Dump successful
```

Messages might be shown on the tape display.

number
The number of MB dumped.

dump*end
The dump process ended successfully.

5. You can IPL Linux again.

z/VM guest example

You can initiate a dump to tape from a Linux instance that is running as a z/VM guest.

Procedure

If 193 is the tape device:

1. Rewind the tape:

```
#cp rewind 193
```

2. Stop all CPUs:

```
#cp cpu all stop
```

3. Store status:

```
#cp store status
```

4. IPL the tape device:

```
#cp i 193
```

Results

On z/VM, a three-processor machine in this example, you will see messages about the disabled wait:

```
01: The virtual machine is placed in CP mode due to a SIGP stop from CPU 00.
02: The virtual machine is placed in CP mode due to a SIGP stop from CPU 00.
"CP entered; disabled wait PSW 00020000 80000000 00000000 00000000"
```

You can now IPL your Linux instance and resume operations.

LPAR example

You can initiate a dump to tape on an LPAR from an HMC (Hardware Management Console) or SE (Support Element).

About this task

The following description refers to an HMC, but the steps also apply to an SE.

Procedure

1. In the navigation pane of the HMC, expand Systems Management and select the hardware system you want to work with.

   A table of LPARs is displayed in the content area.
2. Select the LPAR for which you want to initiate the dump.
3. In the **Tasks** area, click **Stop all processors** to stop all CPUs. Confirm when you are prompted to do so.
4. In the **Tasks** area, click **Load** to display the Load panel.
   Figure 7 on page 27 shows an example of an HMC with a selected hardware system and LPAR. The **Load** and **Stop all processors** tasks can be seen in the **Tasks** area.

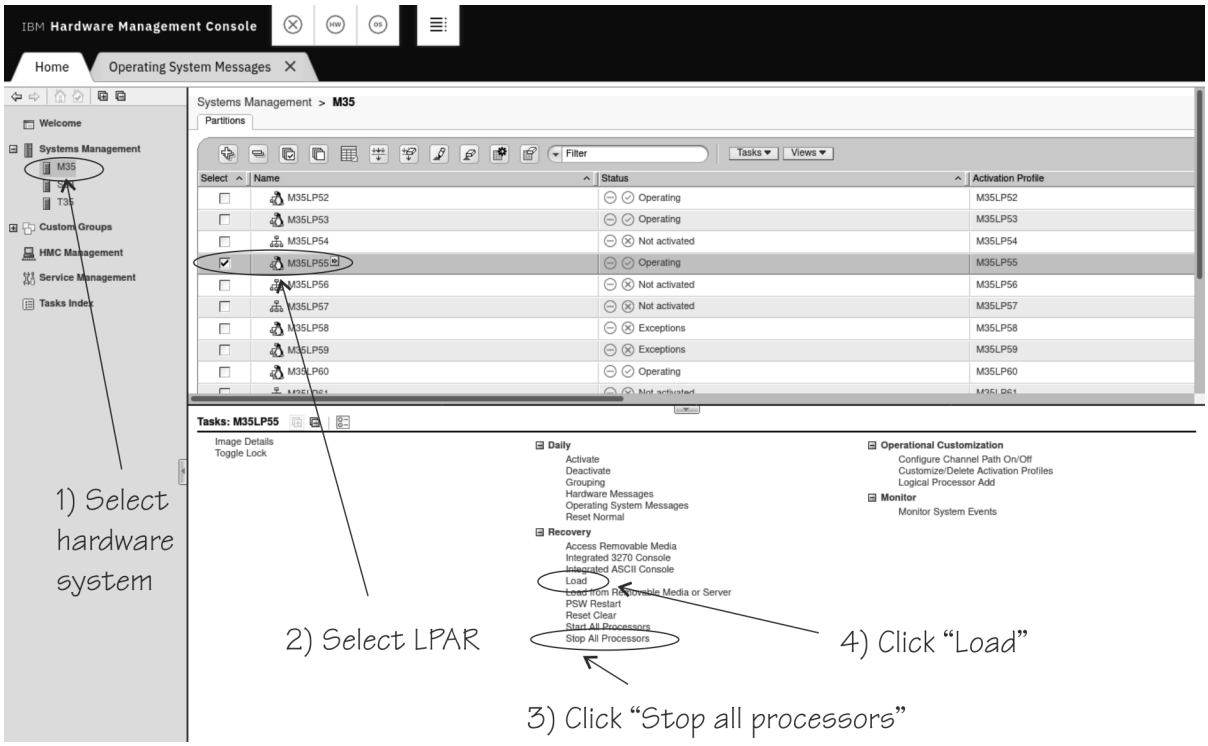


*Figure 7. HMC with the **Load** and **Stop all processors** tasks*

5. Select **Load type** "Standard load".
6. Select the **Store status** check box.
7. Type the device number of the dump device into the Load address field. To IPL from a subchannel set other than 0, specify five digits: The subchannel set ID followed by the device number, for example 1E711.

   Figure 8 on page 28 shows a Load panel with all entries and selections required to start the dump process for a channel-attached tape dump device.

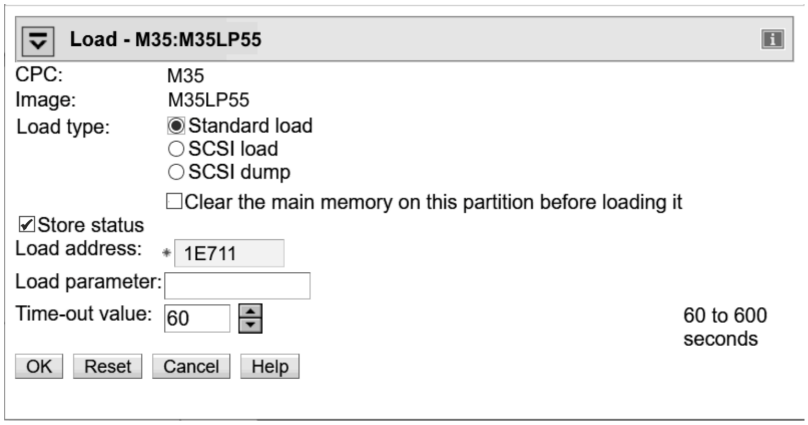


Figure 8. Load panel for dumping to tape

8. Click **OK** to start the dump process.
9. Wait until the dump process completes. Click the **Operating System Messages** icon for progress and error information.

Results

When the dump has completed successfully, you can IPL Linux again.

Copying the dump from tape

You can copy a tape dump to a file system by using the **zgetdump** tool.

Before you begin

The **mt** utility must be installed.

Preparing the dump tape

You need to rewind the tape, and find the correct position on the tape to start copying from.

About this task

Use the **mt** tool to manipulate the tape.

Procedure

1. Rewind the tape.

   For example:

```
# mt -f /dev/ntibm0 rewind
```

2. Skip the first file on the tape (this file is the dump tool itself).

   For example:

```
# mt -f /dev/ntibm0 fsf
```

Using the zgetdump tool to copy the dump

Use the **zgetdump** tool to copy the dump file from the tape to a file system.

Before you begin

The tape must be in the correct position (see “Preparing the dump tape” on page 28).

About this task

By default, the **zgetdump** tool takes the dump device as input and writes its contents to standard output. To write the dump to a file system, you must redirect the output to a file.

The example assumes that the dump is on tape device `/dev/ntibm0`.

Procedure

Copy the dump from tape to a file named `dump.elf` in the file system:

```
# zgetdump /dev/ntibm0 > dump.elf
```

For general information on **zgetdump**, see “zgetdump - Copy and convert kernel dumps” on page 66 or the man page.

Checking whether a dump is valid, and printing the dump header

To check whether a dump is valid, use the **zgetdump** command with the **-i** option.

Procedure

1. Ensure that the volume is loaded.
2. Skip the first file on the tape (this is the dump tool itself):

   ```
   # mt -f /dev/ntibm0 fsf
   ```

3. Issue the **zgetdump** command with the **-i** option:

   ```
   # zgetdump -i  /dev/ntibm0
   ```

   The **zgetdump** command goes through the dump until it reaches the end. See also “Using zgetdump to copy a tape dump” on page 68.

Chapter 6. Using a SCSI disk dump device

To use a SCSI dump device you need to install the stand-alone SCSI dump tool, perform the dump process, and copy the dump to a file in a Linux file system.

Installing the SCSI disk dump tool

You install the SCSI dump tool with the **zipl** command.

Before you begin

The dump partition needs enough free space (memory size + 10 MB) to hold the system memory.

GRUB 2 usage: You cannot use GRUB 2 to install the stand-alone dump tools. You must use the **zipl** command as described in this document for the dump tools.

The following examples assume that a SCSI dump disk is accessed through the partition device node on a device mapper multipath device `/dev/mapper/36005076303ffd40100000000000020c0-part1` and is IPLed with the following parameters:

- devno: 0.0.4711
- wwpn: 0x4712076300ce93a7
- lun: 0x4712000000000000

About this task

As an example, a partition on a SCSI disk is used as dump partition.

This example assumes that `/dev/mapper/36005076303ffd40100000000000020c0` is the dump device, and that you want to dump to the first partition, `/dev/mapper/36005076303ffd40100000000000020c0-part1`. Always use multipath devices instead of single path SCSI disk device nodes, if possible.

Procedure

1. Create a partition with **fdisk** or **parted**, or the expert partitioner of **yast disk**, using the PC-BIOS or GPT layout.
   For example:

```
# fdisk /dev/mapper/36005076303ffd40100000000000020c0
```

2. Install the dump tool by using the **zipl** command.
   Specify the dump partition on the command line:
   For example:

```
# zipl -d /dev/mapper/36005076303ffd40100000000000020c0-part1
```

Results

When you perform an IPL from any of the paths of `/dev/mapper/36005076303ffd40100000000000020c0` by using boot program selector 1 or 0 (default), the memory dump is written directly to partition 1 of `/dev/mapper/36005076303ffd40100000000000020c0`. For LPAR, the boot program selector is located on the load panel, see Figure 10 on page 34 for an example. For z/VM, the boot program selector is configured with 'CP SET DUMPDEV BOOTPROG', see “z/VM guest example” on page 32.

Initiating a SCSI dump

To initiate the dump, IPL the SCSI dump tool by using the **SCSI dump** load type. To IPL the dump tool, specify its WWPN, LUN, and device- bus ID.

About this task

The dump process can take several minutes depending on the device type you are using and the amount of system memory. The dump progress and any error messages are reported on the operating system messages console for LPAR, or on the 3270 console for a z/VM guest.

Procedure

IPL the SCSI dump tool.

After the dump tool is IPLed, messages that indicate the progress of the dump are written to the console:

```
Linux System Dumper starting

Version 3.0 (64 bit)
Linux version ...

Dump start at:.
 Fri, 9 Dec 2016 13:22:55 +0000

Dump parameters:.
 devno....: 0.0.4711
 wwpn.....: 0x4712076300ce93a7
 lun......: 0x4712000000000000
 conf.....: 0
 partition: /dev/sda1

Writing dump:
      0 of 6144 MB   0.0%    0 MB/s unknown ETA
   2489 of 6144 MB  40.5%  277 MB/s 0:00:13 ETA
   5114 of 6144 MB  83.2%  269 MB/s 0:00:03 ETA
   6144 of 6144 MB 100.0%  267 MB/s 0:00:00 ETA

Dump successful.
```

Results

The dump process copies the dump to the dump partition. When the dump completes successfully, you can IPL Linux again. You can then extract the dump from the dump partition into a file.

z/VM guest example

You can initiate a dump to a SCSI disk using z/VM.

About this task

Assume your SCSI dump disk has the following parameters:

- WWPN: 4712076300ce93a7
- LUN: 4712000000000000
- FCP adapter device number: 4711

Optionally, you can use the `scpdata` parameter to specify additional parameters for the SCSI dump tool, see “zipl - Prepare devices for stand-alone dump” on page 64 .

Results

Messages on the operating system console will show when the dump process is finished.

Example

Using the required parameters:

```
#cp set dumpdev portname 47120763 00ce93a7 lun 47120000 00000000
#cp ipl 4711 dump
```

Using **scpdata** to specify an optional parameter for the SCSI dump tool:

```
#cp set dumpdev portname 47120763 00ce93a7 lun 47120000 00000000 scpdata 'dump_debug=3'
#cp ipl 4711 dump
```

In the example, dump_debug=3 sets the level of debug messages. For more information, see the zipl information in *Device Drivers, Features, and Commands on SUSE Linux Enterprise Server 15 SP5*, SC34-2784.

What to do next

You can now IPL your Linux instance and resume operations.

LPAR example

You can initiate a dump to SCSI disk of a Linux instance running on an LPAR from an HMC or SE web interface.

About this task

The following description refers to an HMC, but the steps also apply to an SE.

As an alternative to the steps that follow, you can also use the HMC Web Services API to trigger the dump process. For more information and an example, see "Example for triggering an LPAR dump to a SCSI dump device" on page 79.

Procedure

1. In the navigation pane of the HMC, expand Systems Management and select the hardware system you want to work with.

   A table of LPARs is displayed in the content area.
2. Select the LPAR for which you want to initiate the dump.
3. In the **Tasks** area, click **Load** to display the Load panel.
   Figure 9 on page 34 shows an example of an HMC with a selected hardware system and LPAR. The **Load** task can be seen in the **Tasks** area.

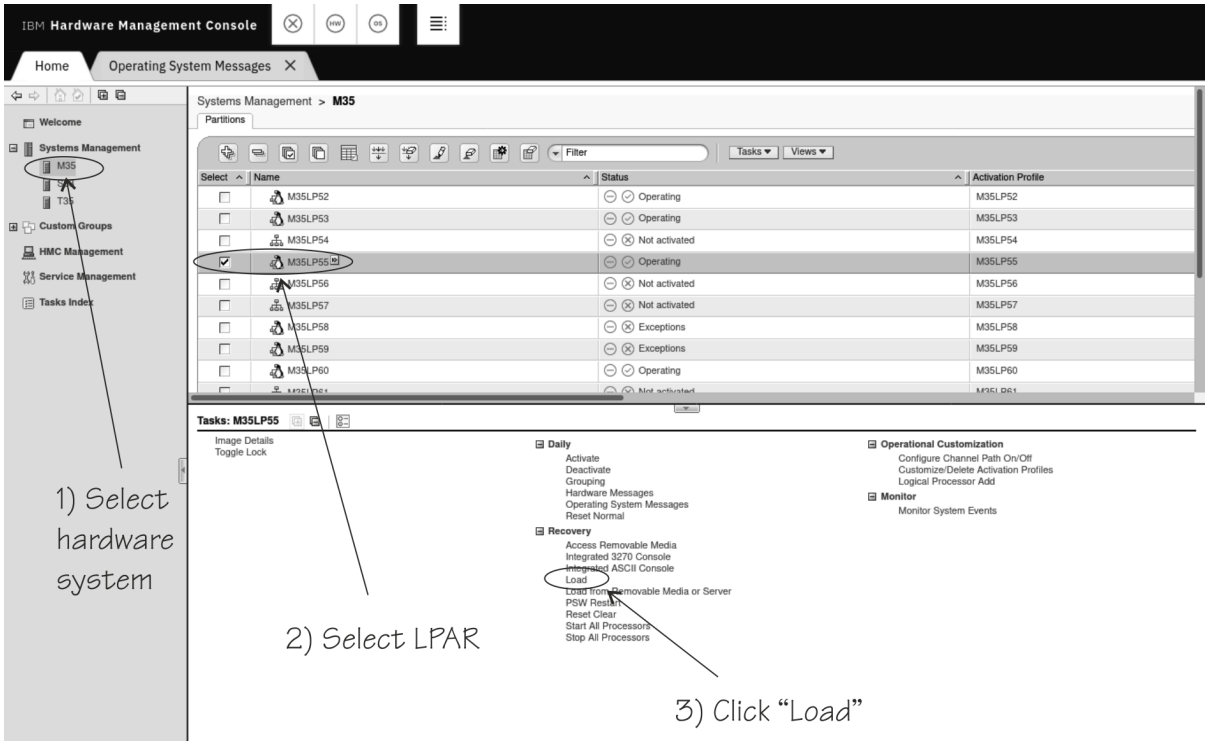


*Figure 9. HMC with the **Load** task*

4. Select **Load type** "SCSI dump".
5. Type the device number of the FCP adapter for the SCSI disk into the **Load address** field.
6. Type the World Wide Port name of the SCSI disk into the **World wide port name** field.
7. Type the Logical Unit Number of the SCSI disk into the **Logical unit number** field.
8. Type 0 in the Boot program selector field.
9. Accept the defaults for the remaining fields.

   Figure 10 on page 34 shows a Load panel with all entries and selections required to start the SCSI dump process.

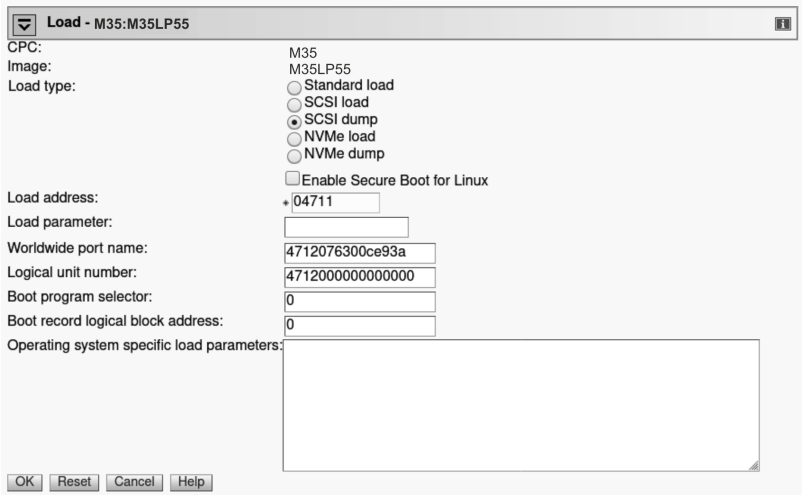


Figure 10. Load panel with enabled SCSI feature for dumping to SCSI disk

   Optionally, you can specify dump tool parameters in the **Operating system specific load parameters** field. For example, for the SCSI dump tool, you can specify `dump_debug=3` to set the level of debug messages.

10. Click **OK** to start the dump process.

11. Wait until the dump process completes. Click the **Operating System Messages** icon for progress and error information.

Results

When the dump has completed successfully, you can IPL Linux again.

DPM partition example

You can initiate a dump process on a DPM partition from a Hardware Management Console (HMC).

Before you begin

SCSI dump devices are FC-attached disk volumes. In DPM mode, the HMC interface presents such disk volumes as part of storage groups of type FCP. To select a SCSI volume as a dump device, you must know its storage group and the UUID that identifies it.

About this task

The following description refers to an HMC.

As an alternative to the steps that follow, you can also use the HMC Web Services API to trigger the dump process. For more information and an example, see "Example for triggering a DPM partition dump" on page 79.

Procedure

1. In the navigation pane of the HMC, expand **Systems Management** and select the hardware system that you want to work with.

   A table of DPM partitions is displayed on the **Partitions** tab in the content area.
2. Select the partition for which you want to initiate the dump.
3. In the **Tasks** area, expand **Recovery** and click **Dump**
   Figure 11 on page 35 shows an example of an HMC with a selected hardware system and partition. The **Dump** task can be seen in the Recovery list within the **Tasks** area.

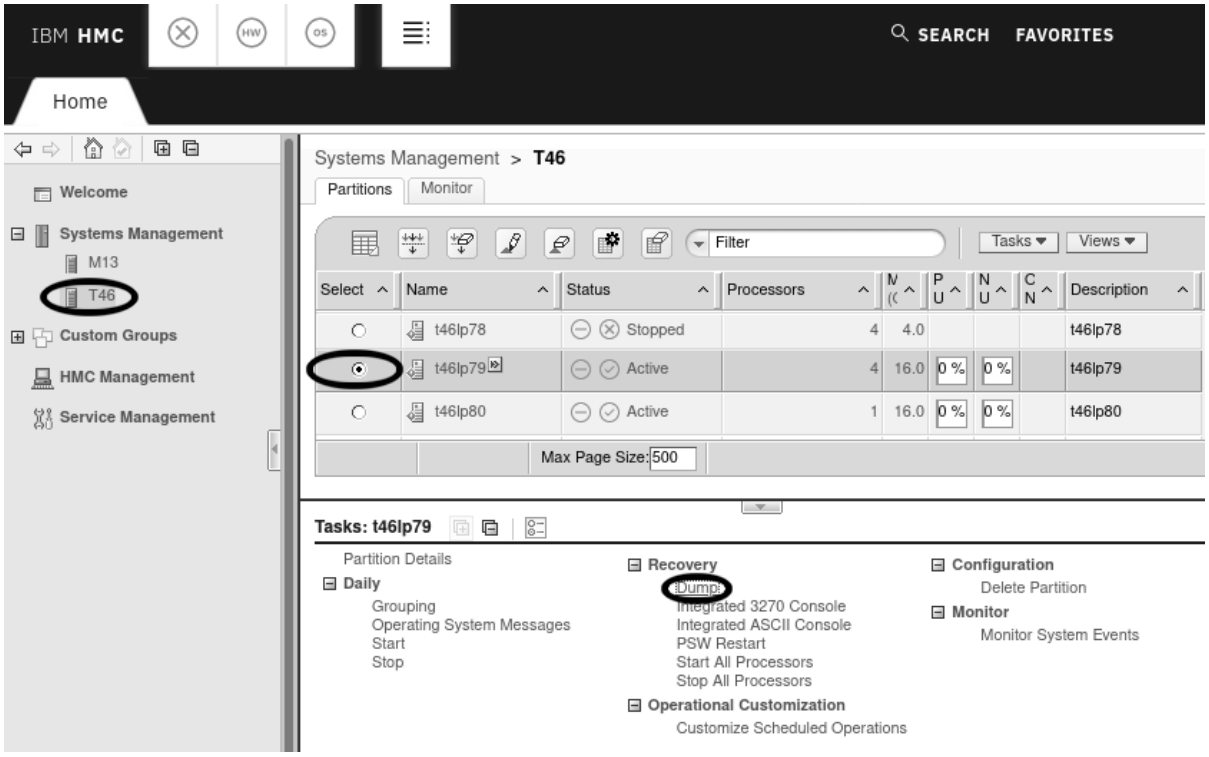


*Figure 11. HMC with the **Dump** task*

4. From the storage group table, select the storage group for your dump device. For SCSI dump the storage group is of type FCP.
5. In the volume list, select the dump device.
6. Type 0 in the Boot program selector field.
7. Accept the defaults for the remaining fields.

   Figure 12 on page 36 shows a **Dump** panel with all entries and selections that are required to start the SCSI dump process.

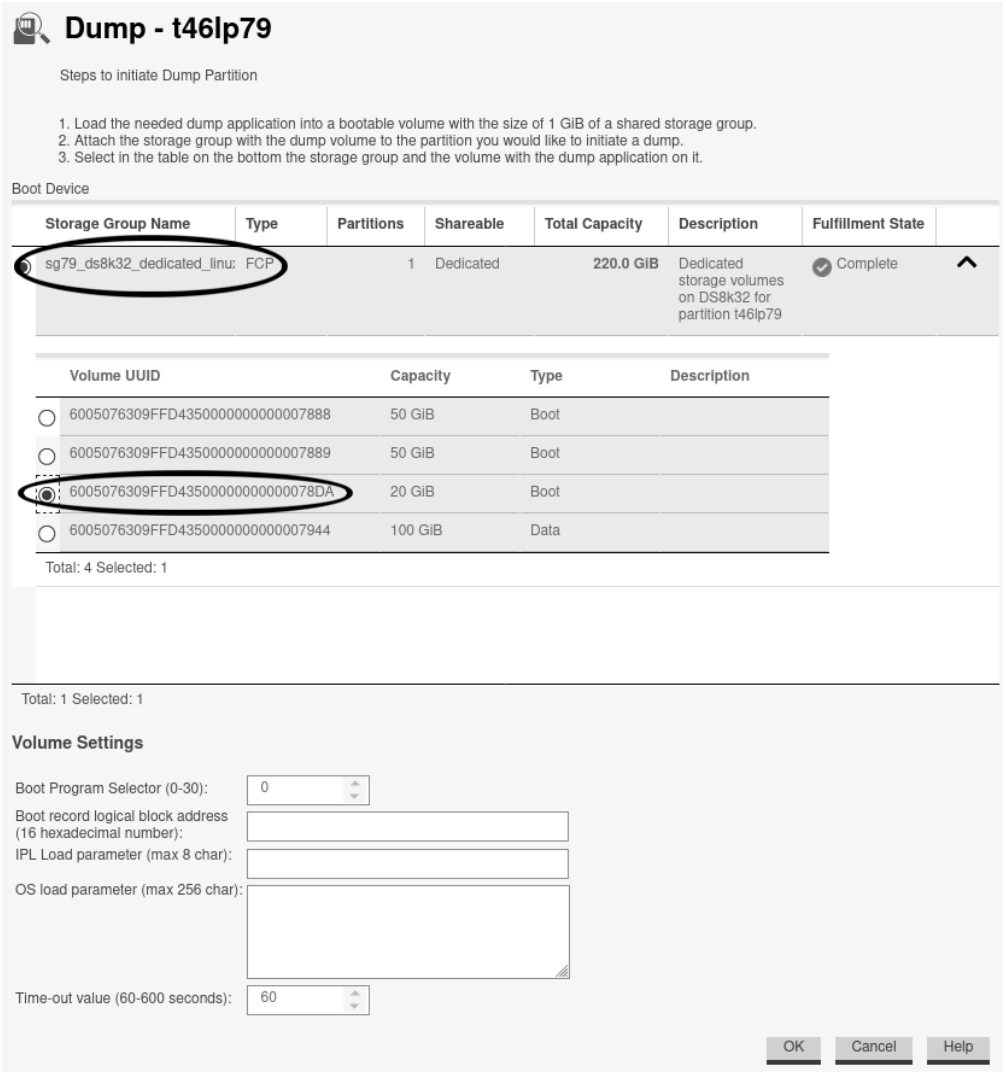


Figure 12. Dump panel with a storage group of type FCP selected

   Optionally, you can specify dump tool parameters in the **OS load parameter** field. For example, for the SCSI dump tool, you can specify `dump_debug=3` to set the level of debug messages. For more information, see the zipl information in *Device Drivers, Features, and Commands on SUSE Linux Enterprise Server 15 SP5*, SC34-2784.
8. Click **OK** to start the dump process.
9. Wait until the dump process completes. Open the **Operating System Messages** tab for progress and error information.

Results

When the dump has completed successfully, you can IPL Linux again.

Automatic dump example

On both z/VM and LPAR, you can use the dumpconf service to set up automatic dumping. In this example, a dump is automatically triggered when a kernel panic occurs.

About this task

To set up dumping using dumpconf, edit the configuration file `/etc/sysconfig/dumpconf`.

Example

Example configuration for an FCP dump device (SCSI disk), where the disk has device bus ID 0.0.4711, the WWPN is 0x4712076300ce93a7, and the LUN is 0x4712000000000000:

```
ON_PANIC=dump
DUMP_TYPE=fcp
DEVICE=0.0.4711
WWPN=0x4712076300ce93a7
LUN=0x4712000000000000
BOOTPROG=0
BR_LBA=0
```

For details on how to set up dumpconf, see "dumpconf - Configure panic or PSW restart action" on page 73.

Copying a SCSI dump to a file

Use the **zgetdump** command to copy a SCSI dump to a file in the file system.

About this task

By default, the **zgetdump** tool takes the dump device as input and writes its contents to standard output. To write the dump to a file system, you must redirect the output to a file.

Procedure

Assuming that the dump is on SCSI partition `/dev/mapper/36005076303ffd40100000000000020c0-part1` and you want to copy it to a file named `dump.elf`:

```
# zgetdump /dev/mapper/36005076303ffd40100000000000020c0-part1 > dump.elf
```

What to do next

You can use **zgetdump** to display information about the dump. See "Checking whether a SCSI dump is valid and printing the dump header" on page 69 for an example. For general information about **zgetdump**, see "zgetdump - Copy and convert kernel dumps" on page 66 or the man page.

Printing the SCSI dump header

To print the dump file header, use **zgetdump** with the **-i** option.

Procedure

Specify the **zgetdump** command with the `-i` option:

```
# zgetdump -i /dev/mapper/36005076303ffd40100000000000020c0-part1
General dump info:
  Dump format........: elf
  Version............: 1
  UTS node name......: mylnxsys
  UTS kernel release.: 4.12.14-18-default
  UTS kernel version.: #1 SMP Mon Jun 11 12:18:48 UTC 2018
  System arch........: s390x (64 bit)
  CPU count (online).: 3
  Dump memory range..: 768 MB

Memory map:
  0000000000000000 - 000000002fffffff (768 MB)
```

Chapter 7. Using an NVMe disk dump device

To use an NVMe device, you need to install the stand-alone NVMe disk dump tool, perform the dump process, and copy the dump from a partition to a file in a Linux file system.

You can use NVMe disks as dump devices for LPARs and DPM partitions as of LinuxONE III hardware.

Installing the NVMe disk dump tool

You install the NVMe disk dump tool with the **zipl** command.

Before you begin

- You need the s390-tools RPM with the ngdump dracut module.
- The dump partition needs enough free space (memory size + 10 MB) to hold the system memory.
- Depending on your HMC version, you might have to install the NVMe dump tool on a partition in name space 1 to be able to trigger an LPAR dump from the HMC GUI.

About this task

A partition on an NVMe disk is used as a dump partition.

The following example assumes that the device node for this partition is `/dev/nvme0n1p1`. In the node name, n1 denotes name space 1 and p1 partition 1 on that name space.

Procedure

1. Create the partition with **sfdisk**, using the GPT or MBR layout.
   For example issue these commands:

   ```
   # echo "label: gpt" | sfdisk /dev/nvme0n1
   # echo ";" | sfdisk /dev/nvme0n1
   ```

   You can confirm that the partition has been created by issuing **ls -l /dev/nvme0n1p1**.
2. Install the dump tool by using the **zipl** command.

   Specify the dump partition on the command line:

   For example:

   ```
   # zipl -d /dev/nvme0n1p1
   ```
3. Optional: Confirm that name space 1 of your NVMe disk is a valid dump device.
   For example:

   ```
   # zgetdump -d /dev/nvme0n1
   ```

   See also, “Checking whether an NVMe disk contains a valid dump record” on page 70.

Results

When you perform an IPL from `/dev/nvme0n1` by using boot program selector 1 or 0 (default), the memory dump is written directly to `/dev/nvme0n1p1`. The boot program selector is located on the load panel, see Figure 14 on page 42 for an example.

Initiating an NVMe disk dump

To initiate the dump, IPL the NVMe dump tool with the NVMe dump load type. To IPL the dump tool, specify the function ID and name space ID of the NVMe disk on which the dump tool is installed.

About this task

The dump process can take several minutes depending on the device type you are using and the amount of system memory. The dump progress and any error messages are reported on the operating system messages console.

Procedure

IPL the NVMe dump tool.

After the dump tool is IPLed, messages that indicate the progress of the dump are written to the console:

```
Starting NGDump...
[    8.932343] ngdump.sh[327]: NGDump started
[    8.934739] ngdump.sh[336]: Checking for memory holes                    : [  0.0 %] /
[    8.967536] ngdump.sh[336]: Checking for memory holes                    : [100.0 %] |
[    9.076976] ngdump.sh[336]: Excluding unnecessary pages                  : [100.0 %] \
[   11.721157] ngdump.sh[336]: Copying data                                 : [  0.0 %] -
[   12.317319] ngdump.sh[336]: Copying data                                 : [ 22.5 %] /        eta: 2s
[   13.273000] ngdump.sh[336]: Copying data                                 : [100.0 %] |        eta: 0s
[   13.273244] ngdump.sh[336]: The kernel version is not supported.
[   13.273263] ngdump.sh[336]: The makedumpfile operation may be incomplete.
[   13.273281] ngdump.sh[336]: The dumpfile is saved to /ngdump/dump.elf.
[   13.273299] ngdump.sh[336]: makedumpfile Completed.
```

Results

The dump process copies the dump to the dump partition. After the dump completes successfully, you can IPL Linux again. You can then extract the dump from the dump partition into a file.

LPAR example

You can initiate a dump process on an LPAR from a Hardware Management Console (HMC) or from the Support Element (SE).

About this task

The following description refers to an HMC, but the steps also apply to an SE.

Procedure

1. In the navigation pane of the HMC, expand Systems Management and select the hardware system you want to work with.

   A table of LPARs is displayed in the content area.
2. Select the LPAR for which you want to initiate the dump.
3. In the **Tasks** area, click **Load** to display the Load panel.
   Figure 13 on page 41 shows an example of an HMC with a selected hardware system and LPAR. The **Load** task can be seen in the **Tasks** area.

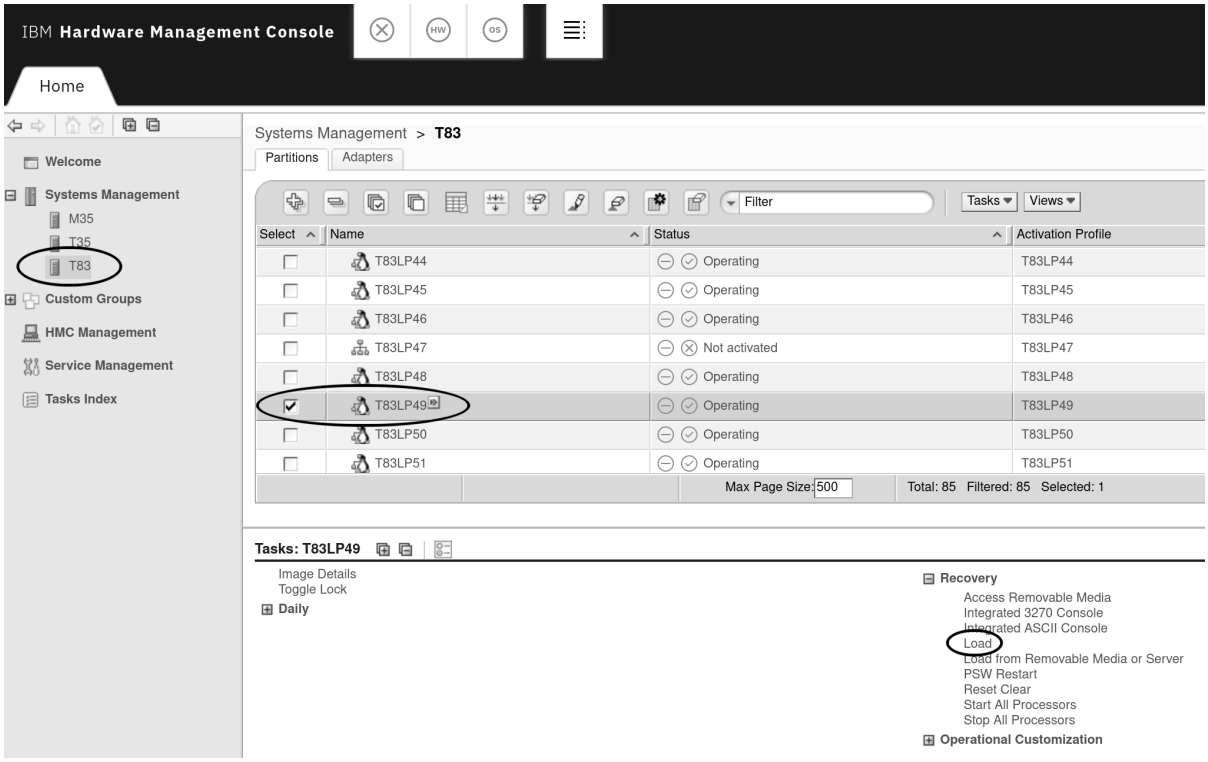


*Figure 13. HMC with the **Load** task*

4. Select **Load type** "NVMe dump".
5. Enter the function ID of the NVMe device in the **Load address** field. You can omit leading zeroes.
6. Enter the name space ID in the corresponding field. If your version of the Load panel does not provide this field, name space ID 1 is used.
7. Type 0 in the **Boot program selector** field.
8. Accept the defaults for the remaining fields.

   Figure 14 on page 42 shows a Load panel with all entries and selections required to start the NVMe dump process. The example panel does not provide a field for the name space ID, so name space 1 is used.

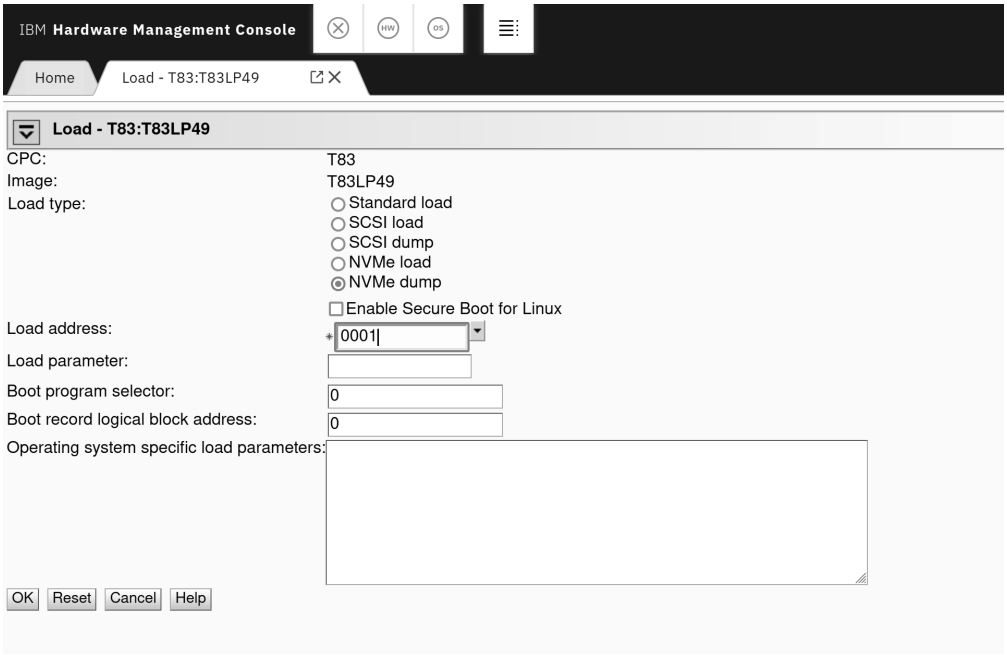


Figure 14. Load panel with specifications for dumping to an NVMe disk

9. Click **OK** to start the dump process.
10. Wait until the dump process completes. Open the **Operating System Messages** tab for progress and error information.

Results

When the dump has completed successfully, you can IPL Linux again.

DPM partition example

You can initiate a dump process on a DPM partition from a Hardware Management Console (HMC).

Before you begin

In DPM mode, the HMC interface presents NVMe disks as volumes within storage groups of type NVMe. To select an NVMe volume as a dump device, you must know its storage group and the serial number that identifies it.

About this task

The following description refers to an HMC.

As an alternative to the steps that follow, you can also use the HMC Web Services API to trigger the dump process. For more information and an example, see “Example for triggering a DPM partition dump” on page 79.

Procedure

1. In the navigation pane of the HMC, expand **Systems Management** and select the hardware system that you want to work with.

   A table of DPM partitions is displayed on the **Partitions** tab in the content area.
2. Select the partition for which you want to initiate the dump.
3. In the **Tasks** area, expand **Recovery** and click **Dump**
   Figure 15 on page 43 shows an example of an HMC with a selected hardware system and partition. The **Dump** task can be seen in the Recovery list within the **Tasks** area.

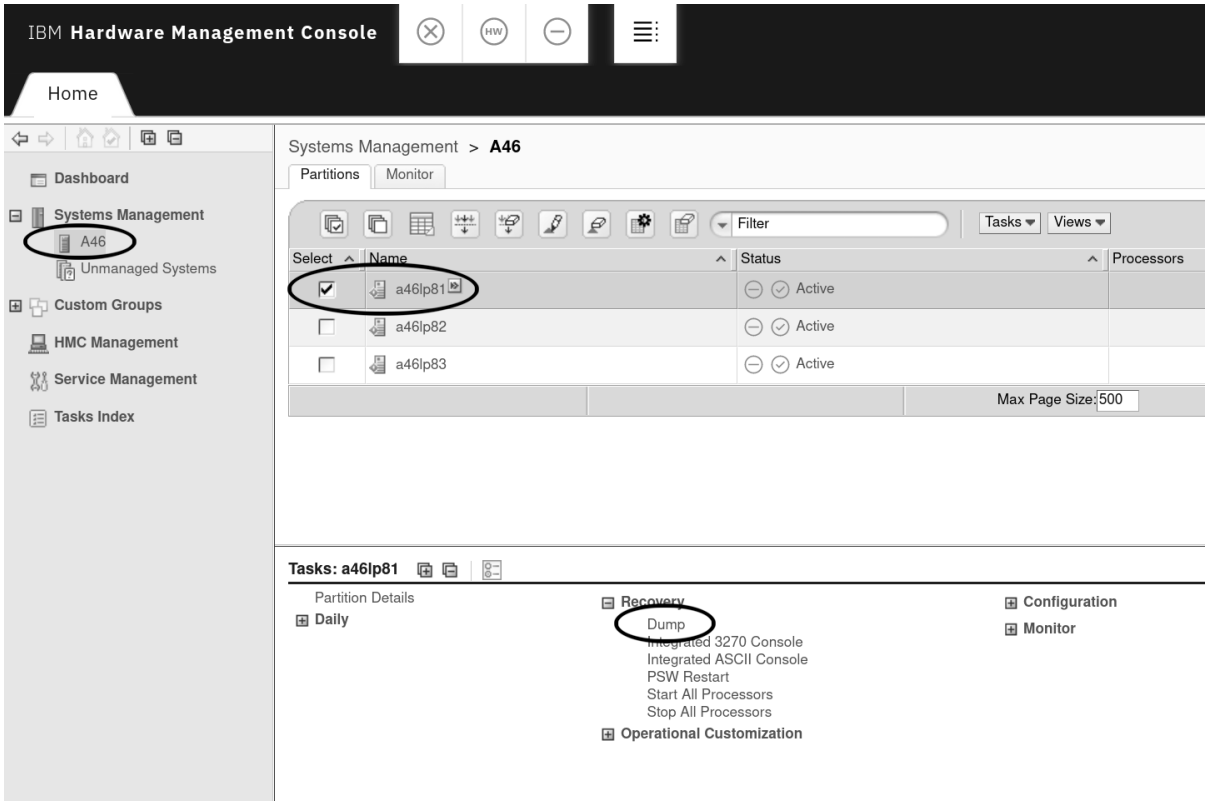


*Figure 15. HMC with the **Dump** task*

4. From the storage group table, select the storage group for your dump device. For NVMe dump, the storage group is of type NVMe.
5. In the volume list, select the dump device.
6. Type 0 in the Boot program selector field.
7. Accept the defaults for the remaining fields.

   Figure 16 on page 44 shows a **Dump** panel with all entries and selections that are required to start the NVMe dump process.

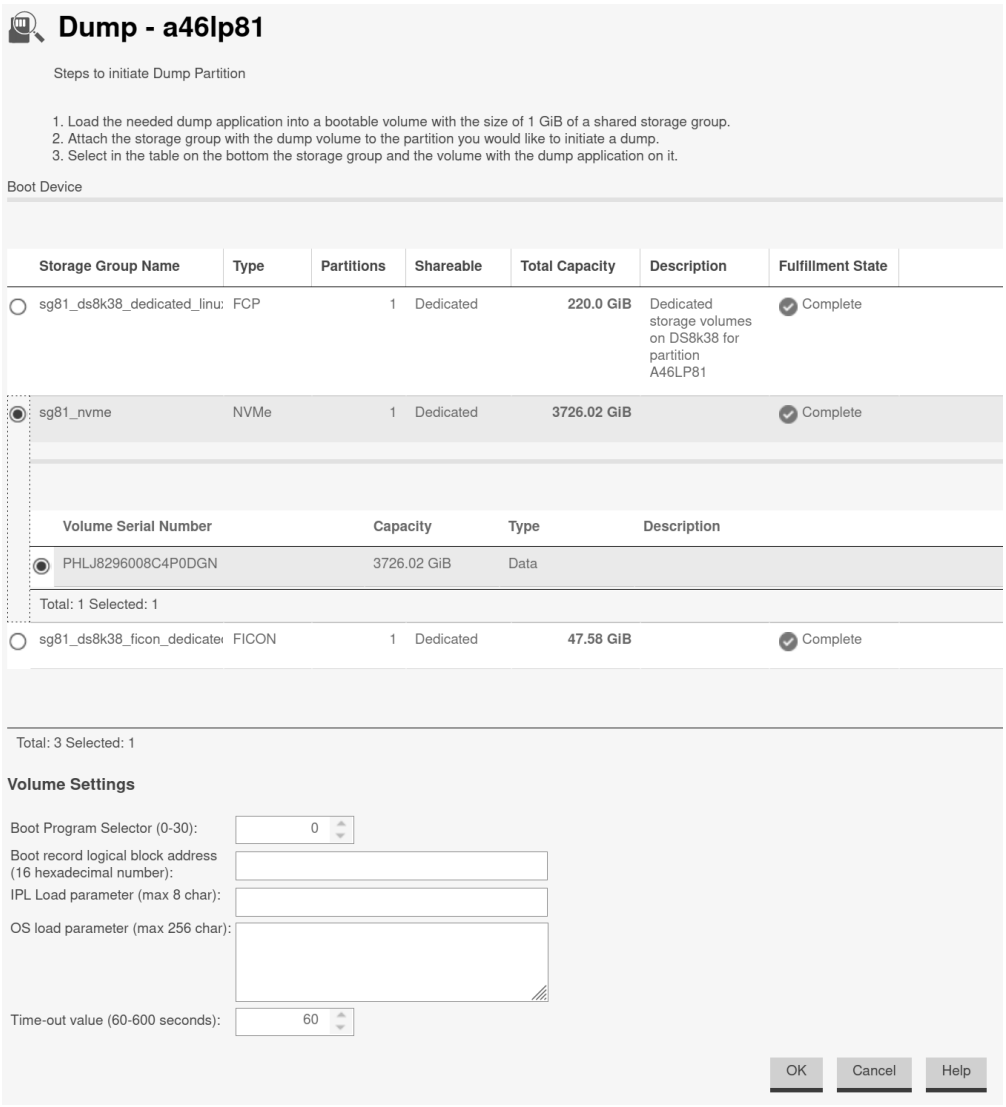


Figure 16. Dump panel with a storage group of type NVMe selected

8. Click **OK** to start the dump process.
9. Wait until the dump process completes. Open the **Operating System Messages** tab for progress and error information.

Results

When the dump has completed successfully, you can IPL Linux again.

Automatic dump example

You can use the dumpconf service to set up automatic dumping. In this example, a dump is automatically triggered when a kernel panic occurs.

About this task

To set up dumping using dumpconf, edit the `/etc/sysconfig/dumpconf` configuration file.

Example

Example configuration for an NVMe disk, where the disk corresponds to an NVMe device with function ID `0x00000001` and name space ID `0x00000001`:

```
ON_PANIC=dump
DUMP_TYPE=nvme
FID=0x00000001
NSID=0x00000001
BOOTPROG=0
BR_LBA=0
```

For details on how to set up dumpconf, see “dumpconf - Configure panic or PSW restart action” on page 73.

Copying an NVMe dump to a file

Use the **zgetdump** command to copy an NVMe dump to a file in the file system.

About this task

By default, the **zgetdump** tool takes the dump device as input and writes its contents to standard output. To write the dump to a file system, you must redirect the output to a file.

Procedure

Assuming that the dump is on an NVMe partition `/dev/nvme0n1p1` and you want to copy it to a file named `dump.elf`:

```
# zgetdump /dev/nvme0n1p1 > dump.elf
```

What to do next

You can use **zgetdump -i** to display information about the dump. See “Printing the NVMe dump header” on page 45 for an example. For general information about **zgetdump**, see “zgetdump - Copy and convert kernel dumps” on page 66 or the man page.

Printing the NVMe dump header

To print the dump file header, use **zgetdump** with the **-i** option.

Procedure

Specify the **zgetdump** command with the **-i** option:

```
# zgetdump -i /dev/nvme0n1p1
General dump info:
  Dump format........: elf
  Version............: 1
  UTS node name......: t83lp49
  UTS kernel release.: 5.14.21
  UTS kernel version.: #1 SMP Wed May 10 12:18:48 UTC 2023
  System arch........: s390x (64 bit)
  CPU count (online).: 64
  Dump memory range..: 8192 MB

Memory map:
  0000000000000000 - 00000001ffffffff (8192 MB)
```

Chapter 8. Creating dumps on z/VM with VMDUMP

Use VMDUMP to create dumps on z/VM systems, using the z/VM reader as the dump medium.

About this task

Do not use **VMDUMP** to dump large z/VM guests; the dump process is very slow. Dumping 1 GB of storage can take up to 15 minutes depending on the used storage server and z/VM version.

This section describes how to create a dump with **VMDUMP**, how to transfer the dump to Linux, and how to convert the z/VM dump to a convenient format. **VMDUMP** does not need to be installed separately.

The following sections take you through the process of creating a dump with **VMDUMP**.

Initiating a dump with VMDUMP

Start the VMDUMP process with the CP **VMDUMP** command.

Procedure

Issue the following command from the 3270 console of the z/VM guest virtual machine:

```
#CP VMDUMP
```

Results

z/VM CP temporarily stops the z/VM guest virtual machine and creates a dump file. The dump file is stored in the reader of the z/VM guest virtual machine. After the dump is complete, the Linux on z/VM instance continues operating.

You can use the **TO** option of the **VMDUMP** command to direct the dump to the reader of another guest virtual machine of the same z/VM system.

Example

To write the dump to the reader of z/VM guest virtual machine `linux02` issue:

```
#CP VMDUMP TO LINUX02
```

For more information about **VMDUMP** refer to *z/VM: CP Commands and Utilities Reference*, SC24-6268.

z/VM guest example

You can initiate a dump of Linux instances running under z/VM by using VMDUMP.

Procedure

To initialize a dump with **VMDUMP**, issue this command from the console of your z/VM guest virtual machine:

```
#cp vmdump
```

Results

Dumping does not force you to perform an IPL. If the Linux instance ran as required before dumping, it continues running after the dump is completed.

Copying the dump to Linux

Copy the dump from the z/VM reader using the **vmur** command.

Before you begin

Ensure that the device node for the reader device is available.

```
# ls /dev/vmrdr*
vmrdr-0.0.000c
```

The command output of the example shows the default device node for the reader device.

If the device is not available, you might have to load the `vmur` module and set the device online.

```
# modprobe vmur
# chzdev -e -a 0.0.000c
```

For more information about `vmur`, see *Device Drivers, Features, and Commands on SUSE Linux Enterprise Server 15 SP5*, SC34-2784.

Procedure

1. Find the spool ID of the **VMDUMP** spool file in the output of the **vmur li** command:

   ```
   # vmur li
   ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME   TYPE DIST
   T6360025 0463 V DMP 00020222 001 NONE 06/11 15:07:42 VMDUMP FILE T6360025
   ```

   In the example the required **VMDUMP** file spool ID is 463.
2. Copy the dump into your Linux file system using the **vmur receive** command.

   To convert the dump into a format that can be processed with the Linux dump analysis tool **crash**, convert the dump using the **--convert** option:

   ```
   # vmur rec 463 -c myvmdump
   vmdump information:
     architecture: 64 bit (big)
     storage.....: 256 MB
     date........: Thu Feb  5 08:39:48 2009
     cpus........: 1
      256 of    256 |################################################################| 100%
   ```

Results

The created file, named `myvmdump`, can now be used as input to **crash**.

Chapter 9. Using virsh dump to create dumps on KVM hosts

The **virsh dump** command writes KVM guest dumps to a file in the KVM host file system.

About this task

Use virsh dump as an alternative or as a backup method for creating dumps of KVM guests with kdump.

The **virsh dump** command is installed with the libvirt package on the KVM host. Apart from setting aside a host directory with sufficient storage space to hold the dumps, no special preparation is needed.

You can use **virsh dump** to create dumps of both running or crashed KVM guests. By default, **virsh dump** pauses a running guest during the dump process. If all is well, the guest continues running when the dump is complete, but lengthy pausing might result in a malfunctioning guest. For example, a large guest memory or the encryption associated with a guest in IBM Secure Execution mode can lead to excessive pausing.

For more information about KVM guests, KVM virtual servers, and **virsh** commands, see *KVM Virtual Server Management*, SC34-2752.

Procedure

On the KVM host, issue a virsh dump command with the `--memory-only` option to create a dump.

```
# virsh dump --memory-only [<mode>] <virtual_server> <dump_file>
```

where

--memory-only
writes the dump file in a valid `elf` format as required by the crash tool.

<mode>
Optional: specifies one of the following dump modes.

--crash
halts the KVM guest with a crashed state before starting the dump process.

--reset
resets the virtual server when the dump process is complete.

--live
does not pause the guest during the dump process. This option can result in an inconsistent dump.

<virtual_server>
specifies the name of the virtual server on which the KVM guest runs.

<dump_file>
specifies the file to which the dump is written. If a file name without a fully qualified path is specified, the dump file is written to the current working directory from which the **visrh dump** command is issued.

The following example creates a dump of a Linux instance on a virtual server `vserv3` to a host file `/dumps/vser3.dump1.elf`.

```
# virsh dump vserv3 /dumps/vser3.dump1.elf --memory-only
```

Results

You can use **makedumpfile** to compress and filter the dump, and can analyze the dump with the crash tool.

For dumps of KVM guests in IBM Secure Execution mode, you must decrypt the dump with the **zgetdump** tool before you can compress, filter, or analyze it.

You must enable encrypted dumps when you create the secure image with the **genprotimg** command. This enablement requires two command options: `--enable-dump` and `--comm-key`. With the `--comm-key` option, you specify the customer communication key, which is required to decrypt the dump.

This example shows how a secure image is created with dumps enabled:

```
# genprotimg -i /boot/vmlinuz -r /boot/initrd.img -p parmfile -k HKD-8651-000201C048.crt \
--cert ibm_signkey.crt --cert digicert_intermediate.crt -o /boot/secure-linux \
--enable-dump --comm-key customer.key
```

Encrypted dumps for secure guests require hardware support and were introduced with IBM z16 and IBM LinuxONE Emperor 4.

Chapter 10. Creating live-system dumps with zgetdump

If you require a kernel dump of a Linux instance, but no downtime is acceptable, you can create a kernel dump from a live system without disruption.

Because the Linux system continues running while the dump is written, and kernel data structures are changing during the dump process, the resulting dump contains inconsistencies. The faster the dump process completes, the fewer inconsistencies the resulting live-system dump will contain. Therefore, run the dump process with the highest acceptable priority.

You can change the scheduling priority with the nice command. For example, use **nice -n -20** to set the highest possible priority.

Creating a kernel dump on a live system

You can create non-disruptive kernel dumps on a running Linux system with the **zgetdump** tool.

Before you begin

- The dump directory needs enough free space (memory size + 10 MB) to hold the system memory.
- Ensure that during the dump process no memory hotplug or CPU hotplug is performed.
- If applicable, stop the cpuplugd service by issuing the command:

```
# service cpuplugd stop
```

- Load the crash kernel module by issuing the command:

```
# modprobe crash
```

Procedure

1. Optional: Use the **-i** option to print information for the currently running Linux image:

```
# zgetdump -i /dev/crash
General dump info:
  Dump format........: devmem
  Dump method........: live
  UTS node name......: mylnxsys
  UTS kernel release.: 5.14.21
  UTS kernel version.: #1 SMP Wed May 10 12:18:48 UTC 2023
  System arch........: s390x (64 bit)
  Dump memory range..: 896 MB

Memory map:
    0000000000000000 - 0000000037ffffff (896 MB)
```

2. Create a dump from a live system by specifying `/dev/crash` as input dump and redirecting the output to a dump file. Run the dump process with a high priority.

```
# nice -n -20 zgetdump /dev/crash > dump.elf
```

   Optionally, you can also specify a target dump format with the **-f** option:

```
# zgetdump /dev/crash -f elf > dump.elf
```

3. Optional: Print information for the live-system dump.

   Use the **-i** option to print information for live-system dumps that are generated by **zgetdump**:

```
# zgetdump -i dump.elf
General dump info:
  Dump format........: elf
  Version............: 1
  Dump method........: live
  UTS node name......: mylnxsys
  UTS kernel release.: 5.14.21
  UTS kernel version.: #1 SMP Wed May 10 12:18:48 UTC 2023
  System arch........: s390x (64 bit)
  Dump memory range..: 896 MB

Memory map:
    0000000000000000 - 0000000037ffffff (896 MB)
```

The value "live" in the **Dump method** field indicates that this is a dump from a live system.

Example

```
# nice -n -20 zgetdump /dev/crash -f elf > dump.elf
Format Info:
  Source: devmem
  Target: elf
Copying dump:
  00000000 / 00000896 MB
  00000149 / 00000896 MB
  ...
  00000747 / 00000896 MB
  00000896 / 00000896 MB
Success: Dump has been copied
```

What to do next

After you create a dump from a live system, you can work with crash, see “Opening a live-system dump with the crash tool” on page 52.

After the live dump has been copied to a file system, you can compress it with **makedumpfile**. Note that the dump level must not be greater than 1 because of the dump inconsistencies.

For example:

```
# makedumpfile dump.elf -c -d 1 dump.kdump
```

Opening a live-system dump with the crash tool

Inconsistencies in a kernel dump from a live system can cause some crash commands to fail.

Before you begin

You might need to install debuginfo to see information about a dump when you use the **crash** command. Use the **zypper** command to install debuginfo, for example:

```
# zypper in kernel-default-debuginfo
```

Should you need to install crash, you can use **zypper** again, for example:

```
# zypper install crash
```

Procedure

- Use the **crash** command to find information about whether a dump is from a live system. This information is displayed in the startup messages, or when you use the sys command:

```
# crash dump.elf /boot/vmlinux-5.14.<xx>-<y>-default.gz
...
    KERNEL: /boot/vmlinux-5.14.<xx>-<y>-default.gz
 DEBUGINFO: /usr/lib/debug/boot/vmlinux-5.14.<xx>-<y>-default.debug
  DUMPFILE: dump.elf [LIVE DUMP]
      CPUS: 6
...
crash> sys | grep DUMPFILE
...
 DUMPFILE: dump.elf  [LIVE DUMP]
...
```

The tag [LIVE DUMP] informs you that the dump contains inconsistencies.

- Detect whether a dump is from a live system by using the **`help -p`** command:

```
# crash> help -p | grep flags2
flags2: 40 (LIVE_DUMP)
```

- Use the `--minimal` option if the crash tool fails to start because of inconsistent data structures in the kernel dump.

  With this option, crash tolerates a degree of inconsistency. However, only a subset of crash commands is then available:

```
# crash --minimal dump.elf /boot/vmlinux-5.14.<xx>-<y>-default.gz
...
NOTE: minimal mode commands: log, dis, rd, sym, eval, set, extend and exit
```

Chapter 11. Processing dumps

You can copy and transfer the dump file to another system, reduce the dump size, and send the reduced dump to IBM Support.

Procedure

- For LPAR dumps on DASD, tape, or a SCSI or NVMe disk, use the **zgetdump** command to copy the dump and transfer it to another system, see "zgetdump - Copy and convert kernel dumps" on page 66.

  With kdump, you can transmit the dump through a network. Use existing mechanisms to prevent conflicts when concurrently writing multiple dumps to a shared persistent storage space.
- To analyze the dump, you can use **crash**, see "crash - Analyze kernel dumps" on page 76.
- To receive a VMDUMP file, use the **vmur** command, see "vmur - Receive dumps from the z/VM reader" on page 78

Reducing dump size

Methods exist for handling memory dumps that are especially large (greater than 10 GB in size).

Before you begin

The preferred method for handling dumps of large production systems is using kdump. With kdump, you do not need to set up a dedicated dump device with a dump tool for each individual system. Instead, you set aside storage space to receive any dumps from across your installation. When using kdump, the information in this section applies if you want to set up a backup dump method for a critical system with a large memory.

About this task

Large dumps present a challenge as they:

- Take up a large amount of disk space
- Take a long time dumping
- Use considerable network bandwidth when being sent to a support organization.

Note: Sometimes you can re-create the problem on a test system with less memory, which makes the dump handling much easier. Consider this option before creating a large dump.

Procedure

Complete these steps to prepare and process a large dump.

1. Choose a dump device.

   If you want to dump a system with a large memory footprint, you have to prepare a dump device that is large enough. You can use the following dump devices for large dumps:

   Single-volume DASD
   Depending on the disk size that is configured on the storage server you can use, for example:

   - 3390 DASD with 45 GB
   - 3390 DASD with 1 TB

   Multi-volume DASD
   Can be up to the combined size of 32 ECKD DASD partitions. With the example DASD sizes of 45 GB and 1 TB:

   - 32 x 45 GB ≅ 1.4 TB

- 32 x 1 TB = 32 TB

z/VM emulated FBA device that represents a real SCSI device
FBA disks can be defined with the CP command **SET EDEVICE**. These disks can be used as single-volume DASD dump disks. The SCSI disk size depends on your storage server setup.

SCSI disk
The SCSI disk size depends on your storage server setup. For SCSI dump partitions greater than 2 TB, you must use the GPT disk layout.

NVMe disk
For NVMe dump partitions greater than 2 TB, you must use the GPT disk layout.

Dump on 3592 channel-attached tape drive
Cartridges with up to 10 TB capacity.

Do not use **VMDUMP** for large systems, because this dump method is very slow.

2. Estimate the dump time.

   The dump speed depends on your environment, for example your SAN setup and your storage server. Assuming about 100 MB per second dump speed on DASDs or SCSI disks, and a system with 50 GB memory, the dump takes about eight minutes. Do a test dump on your system to determine the dump speed for it. Then you will have an indication of how long a dump will take in case of emergency.
3. Reduce the dump size.

   For transferring dumps in a short amount of time to a support organization, it is often useful to reduce the dump size or split the dump into several parts for easier and faster transmission. To reduce the dump, choose one of these methods:

   - “Compressing a dump using makedumpfile” on page 56
   - “Compressing a dump using gzip and split” on page 57
4. Send the dump.

Compressing a dump using makedumpfile

Use the **makedumpfile** tool to compress s390 dumps and exclude memory pages that are not needed for analysis. Alternatively, you can use the **gzip** and **split** commands.

About this task

Compressing the dump substantially reduces the size of dump files and the amount of time needed to transmit them from one location to another. In SUSE Linux Enterprise Server 15 SP5, the **makedumpfile** tool is part of the makedumpfile RPM that you can install, for example, with the command:

```
# zypper in makedumpfile
```

Because **makedumpfile** expects as input dump files in ELF format, you first have to transform your s390 format dump to ELF format. This is best done by mounting the dump using the **zgetdump** command. If you have read access to the dump file, you do not need root authority to mount the dump with **zgetdump**.

Procedure

1. Mount the dump in ELF format by performing one of these steps:
   - To mount a DASD dump from the partition `/dev/dasdb1` to `/mnt`, issue:

     ```
     # zgetdump -m -f elf /dev/dasdb1 /mnt
     ```
   - To mount a SCSI dump from the partition `/dev/mapper/36005076303ffd4010000000000000020c0-part1` to `/mnt`, issue:

     ```
     # zgetdump -m -f elf /dev/mapper/36005076303ffd4010000000000000020c0-part1 /mnt
     ```

- To mount an NVMe dump from the partition /dev/nvme0n1p1 to /mnt, issue:

```
# zgetdump -m -f elf /dev/nvme0n1p1 /mnt
```

After mounting the dump in ELF format with **zgetdump**, the dump is available in the file named /mnt/dump.elf.

2. Create a file with a filtered and compressed version of the dump.

   Use the **makedumpfile-d** (dump level) option to exclude pages that are typically not needed to analyze a kernel problem. For dump level 31, pages containing only zeroes, pages used to cache file contents (cache, cache private), pages belonging to user-space processes, and free pages are all excluded.

   See the man page for **makedumpfile** for a description of the dump level and other options of **makedumpfile**.

   The following command accesses a dump at /mnt/dump.elf filters it with dump level 31, compresses it, and writes it to a file /dumps/dump.kdump:

```
# makedumpfile -c -d 31 /mnt/dump.elf /dumps/dump.kdump
```

   You might want to retain a copy of the original dump file until the problem is resolved. This reserves the option to create further copies at different dump levels should any of the excluded pages be required for problem determination.

3. Optional: For initial problem analysis, you can also extract the kernel log with **makedumpfile**, and send it to your support organization:

```
# makedumpfile --dump-dmesg /mnt/dump.elf /dumps/kernel.log
```

What to do next

After you have used **makedumpfile**, you can unmount the dump:

```
# zgetdump -u /mnt
```

Compressing a dump using gzip and split

Use the **gzip** and **split** commands to compress the dump and split it into parts. Alternatively, you can use the **makedumpfile** command.

Procedure

1. Compress the dump and split it into parts of 1 GB by using the **gzip** and **split** commands.
   - For a DASD dump:

```
# zgetdump /dev/dasdd1 | gzip | split -b 1G
```

   - For a tape dump:

```
# mt -f /dev/ntibm0 rewind
# mt -f /dev/ntibm0 fsf
# zgetdump /dev/ntibm0 | gzip | split -b 1G
```

   - For a SCSI dump:

```
# zgetdump /dev/mapper/36005076303ffd40100000000000020c0-part1 | gzip | split -b 1G
```

   - For an NVMe dump:

```
# zgetdump /dev/nvme0n1p1 | gzip | split -b 1G
```

The split creates several compressed files in your current directory:

```
# ls
# xaa xab xac xad xae
```

2. Create md5 sums of parts:

```
# md5sum * > dump.md5
```

3. Upload the parts together with the MD5 information to the support organization.
4. The receiver (the support organization) must do the following:
   a) Verify md5 sums:

```
# cd dumpdir
# md5sum -c dump.md5
xaa: OK
xab: OK
...
```

   b) Merge parts and extract the dump:

```
# cat x* | gunzip -c > dump
```

Preparing for analyzing a dump

To analyze your dump with **crash**, additional files are required.

If you need to send your dump for analysis, it might be good to include these additional files with the dump file. SUSE Linux Enterprise Server provides the additional files in RPMs.

To begin analyzing the dump using **crash**, these two files (as a minimum) are required:

- vmlinux (text): Contains addresses of kernel symbols
- vmlinux (debug): Contains datatype debug information

If you need to send your dump for analysis, please include the RPMs shown in Table 4 on page 59.

SLES debug files

The SLES debug files are:

Table 3. SUSE Linux Enterprise Server debug file names

Debug file	Path
System.map	/boot/System.map-5.14.*<xx>*-*<y>*-default
vmlinux (text)	/boot/vmlinux-5.14.*<xx>*-*<y>*-default.gz
vmlinux (debug)	/usr/lib/debug/boot/vmlinux-5.14.*<xx>*-*<y>*-default.debug
modules (text)	/lib/modules/5.14.*<xx>*-*<y>*-default/kernel/
modules (debug)	/usr/lib/debug/lib/modules/5.14.*<xx>*-*<y>*-default/kernel/
debug source	/usr/src/debug/kernel-default-5.14.*<xx>*-*<y>*/linux-5.14/

Files that contain the addresses of kernel symbols are named (text).
Files that contain the corresponding datatype debug information are named (debug).

The RPMs that contain the debuginfo files are:

Table 4. SUSE Linux Enterprise Server debuginfo RPM names

SLES version	RPM
SLES 15	• kernel-default-5.14.*<xx>*-*<y>*.s390x.rpm • kernel-default-debuginfo-5.14.*<xx>*-*<y>*.s390x.rpm • kernel-default-debugsource-5.14.*<xx>*-*<y>*.s390x.rpm

For debugging problems that are related to kernel modules, additional RPMs might be required.

Sending a dump to IBM Support

After compressing the dump, you can transfer it using FTPS, SFTP, or an HTTPS upload server.

Before you begin

You must open a support case and obtain a support case number before you send the data.

Procedure

See Enhanced Customer Data Repository (ECuRep) at https://www.ibm.com/support/ecurep for detailed instructions.

Appendix A. Obtaining a dump with limited size

The `mem` kernel parameter can make Linux use less memory than is available to it. A dump of such a Linux system does not need to include the unused memory. You can use the **`zipl`** command with the **`size`** option to limit the amount of memory that is dumped.

About this task

This section does not apply to kdump.

The **`size`** option is available for all **`zipl`** based dumps except SCSI: DASD and tape in command-line mode or in configuration-file mode. The **`size`** option is appended to the dump device specification with a comma as separator.

The value is a decimal number that can optionally be suffixed with K for kilobytes, M for megabytes, or G for gigabytes. Values that are specified in byte or kilobyte are rounded to the next megabyte boundary.

Be sure not to make the dump size smaller than the amount of memory that is actually used by the system to be dumped. Limiting the dump size to less than the amount of used memory results in an incomplete dump.

Example

The following command, entered in a Linux shell, prepares a DASD dump device for a dump that is limited to 100 MB:

```
# zipl -d /dev/dasdc1,100M
Setting dump size limit to 100MB
Dump target: partition '/dev/dasdc1' with a size of 7043 MB.
Warning: All information on partition '/dev/dasdc1' will be lost!
Do you want to continue creating a dump partition (y/n)?y
Done.
```

After IPL of the dump device, you can see output from the dump tool on the z/VM console as the dump is created:

```
00:
00: CP I 63AD
01: HCPGSP2630I The virtual machine is placed in CP mode due to a SIGP stop
    and store status from CPU 00.
02: HCPGSP2630I The virtual machine is placed in CP mode due to a SIGP stop
    and store status from CPU 00.
00: zIPL v1.15.0-0.132.4 dump tool (64 bit)
00: Dumping 64 bit OS
00:
00: 00000012 / 00000100 MB
00: 00000025 / 00000100 MB
00: 00000037 / 00000100 MB
00: 00000050 / 00000100 MB
00: 00000062 / 00000100 MB
00: 00000075 / 00000100 MB
00: 00000087 / 00000100 MB
00: 00000100 / 00000100 MB
00: Dump successful
00: HCPGIR450W CP entered; disabled wait PSW 00020000 80000000 00000000 00000000
```

An equivalent section in a configuration file could look like this:

```
[dump1]
dumpto=/dev/dasdc1,100M
```

Appendix B. Command summary

The descriptions of the commands contain only the options and parameters that are relevant to dumping on SUSE Linux Enterprise Server. For a full description see the man pages.

zipl - Prepare devices for stand-alone dump

Use **zipl** to prepare a dump device with a stand-alone dump tool.Command options that do not apply to SUSE Linux Enterprise Server 15 SP5 have been omitted. For details, see the man page.

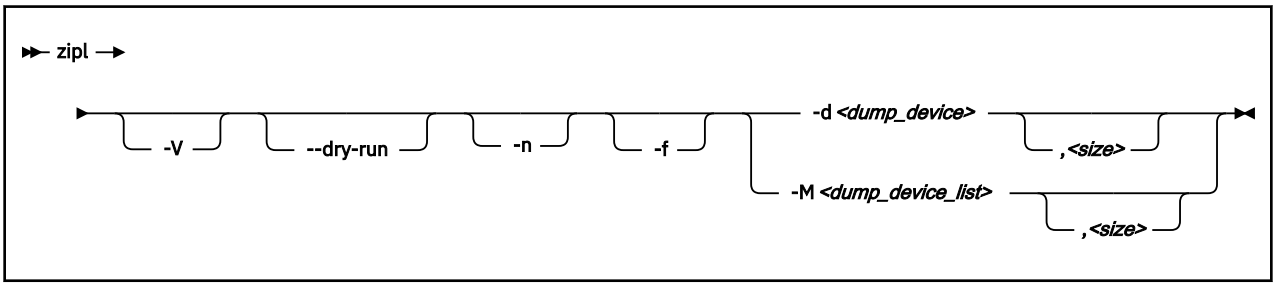
zipl supports the following dump devices:

- Enhanced Count Key Data (ECKD) DASDs with fixed block Linux disk layout (ldl)
- ECKD DASDs with z/OS-compliant compatible disk layout (cdl)
- Fixed Block Access (FBA) DASDs
- Magnetic tape subsystems compatible with IBM3480, IBM3490, IBM3590, or IBM3592
- SCSI with PC-BIOS or GPT disk layout

For multi-volume dumps, only ECKD DASDs with compatible disk layout are supported.

zipl syntax

Note: You can specify zipl parameters in a configuration file, but the preferred way of using zipl is the command line. For details about the configuration file, see the man page.



Parameters

-d *<dump_device>* or --dumpto=*<dump_device>*
is the device node of the tape device or of the DASD, SCSI, or NVMe partition to be prepared as a dump device. **zipl** deletes all data on the partition or tape and installs the boot loader code there.

Note:

- If the dump device is an ECKD disk with fixed-block layout (ldl), a dump overwrites the dump utility. You must reinstall the dump utility before you can use the device for another dump.
- If the dump device is a tape, SCSI disk, NVMe disk, FBA disk, or ECKD disk with the compatible disk layout (cdl), you do not need to reinstall the dump utility after every dump.

-M *<dump_device_list>*
contains the device nodes of the dump partitions, separated by one or more line feed characters. **zipl** writes a dump signature to each involved partition and installs the stand-alone multi-volume dump tool on each involved volume. Duplicate partitions are not allowed. A maximum of 32 partitions can be listed. The volumes must be formatted with cdl and use block size 4096.

<size>
(Optional) The amount of memory to be dumped. The value is a decimal number that can optionally be suffixed with K for kilobytes, M for megabytes, or G for gigabytes. The value is rounded to the next megabyte boundary.

If you limit the dump size below the amount of memory that is used by the system to be dumped, the resulting dump is incomplete. If no limit is provided, all of the available physical memory is dumped.

Note: For SCSI dump devices, the "size" option is not available.

-f or --force
ensures that no signature checking takes place when dumping. Any data on all involved partitions is overwritten without warning.

-n or --noninteractive
suppresses confirmation prompts that require operator responses to allow unattended processing (for example, for processing DASD or tape dump configuration sections). This option is available on the command line only.

-V or --verbose
provides more detailed command output.

--dry-run
simulates a **zipl** command. Use this option to test a configuration without overwriting data on your device.

During simulation, **zipl** performs all command processing and issues error messages where appropriate. Data is temporarily written to the target directory and is cleared up when the command simulation is completed.

-v or --version
displays version information.

-h or --help
displays help information.

Preparing a DASD dump device

The following command prepares a DASD partition `/dev/dasdc1` as a dump device and suppresses confirmation prompts that require an operator response:

```
# zipl -d /dev/dasdc1 -n
```

Preparing a SCSI dump device

The following command prepares a SCSI partition `/dev/mapper/36005076303ffd40100000000000020c0-part1` as a dump device:

```
# zipl -d /dev/mapper/36005076303ffd40100000000000020c0-part1
```

Preparing an NVMe dump device

The following command prepares an NVMe partition `/dev/nvme0n1p1` as a dump device:

```
# zipl -d /dev/nvme0n1p1
```

Preparing a multi-volume dump on ECKD DASD

The following command prepares two DASD partitions `/dev/dasdc1`, `/dev/dasdd1` for a multi-volume dump and suppresses confirmation prompts that require an operator response:

```
# zipl -M mvdump.conf -n
```

where the `mvdump.conf` file contains the two partitions, separated by line breaks:

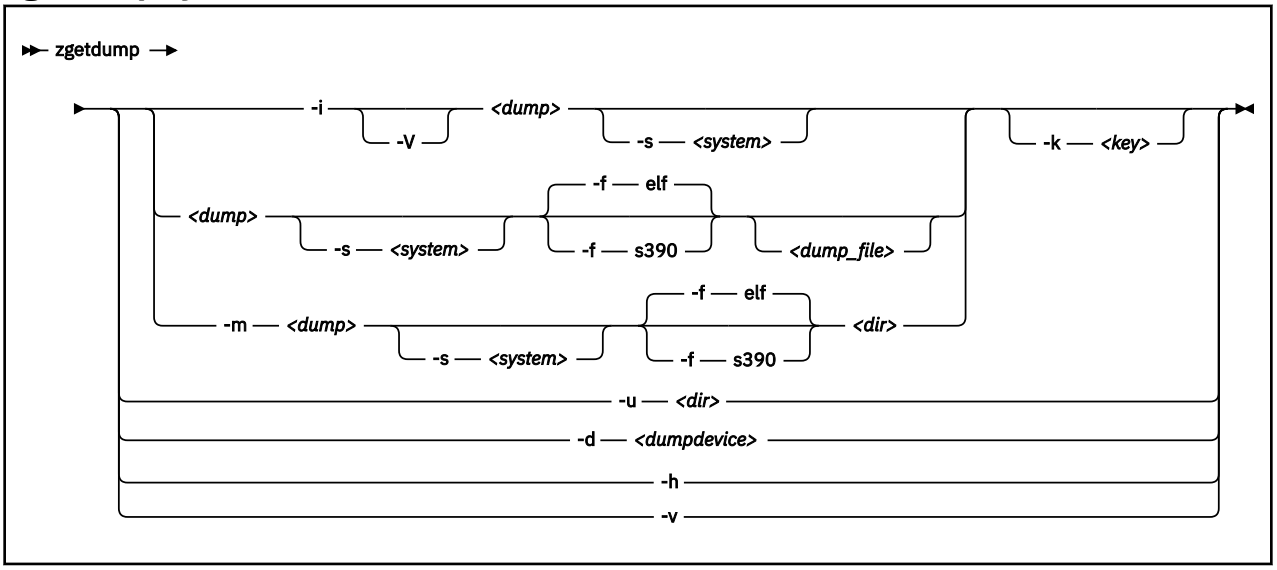
```
/dev/dasdc1
/dev/dasdd1
```

zgetdump - Copy and convert kernel dumps

The **zgetdump** tool copies a source dump into a target dump with a configurable dump format. The source dump can be located either on a dump device or on a file system. The source dump content is written to standard output, unless you redirect it to a specific file. You can also mount the dump content, print dump information, check whether a stand-alone dump device contains a valid dump tool, or create a non-disruptive kernel dump on a live system.

Before you begin: Mounting is implemented with fuse (file system in user space). Therefore, the fuse kernel module must be loaded before you can use the **-m** option. Should the fuse module not be loaded, you can load it, for example, with **modprobe fuse**. You do not need root authority to work with a dump using **zgetdump**.

zgetdump syntax



Parameters

<dump>

is the file, DASD device or partition, multipath partition of a SCSI disk, NVMe partition, channel-attached tape device, or live system device node where the source dump is located:

- Regular dump file (for example `/testdir/dump.0`)
- DASD partition device node (for example `/dev/dasdc1`)
- DASD device node for multi-volume dump (for example `/dev/dasdc`)
- Device mapper multipath partition device node of a SCSI disk (for example `/dev/mapper/36005076303ffd4010000000000020c0-part1`)
- Device node for an NVMe partition (for example, `/dev/nvme0n1p1`).
- Tape device node (for example `/dev/ntibm0`)
- Device node for live system (`/dev/mem`)

Note: For a DASD multi-volume dump, it is sufficient to specify only one of the multi-volume DASDs as *<dump>*.

-s *<system>* or --select *<system>*

for dumps that capture two systems, selects the system of interest. To check whether a dump contains two systems, use **zgetdump -i**. The `-s` option is mandatory when you access the dump of a crashed kdump instance, but returns an error if applied to a regular dump.

A dump can contain data for a crashed production system and for a crashed kdump system. A dump like this is created if a stand-alone dump tool is used to create a dump for a kdump instance that crashed while creating a dump for a previously crashed production system. *<system>* can be:

prod
to select the data for the crashed production system.

kdump
to select the data for the kdump instance that crashed while creating a dump for the previously crashed production system.

-f *<format>* or --fmt *<format>*
uses the specified target dump format *<format>* when writing or mounting the dump. The following target dump formats are supported:

elf
Executable and Linking Format core dump (default)

s390
S/390® dump

-k *<key>* or --key *<key>*
Applies only to the encrypted dumps of KVM guests in IBM Secure Execution mode, see "Working with dumps of KVM guests in IBM Secure Execution mode" on page 71. *<key>* specifies a file with the decryption key.

<dump_file>
is the file to which the output is redirected. The default is standard output.

-m *<dump>* *<dir>* or --mount *<dump>* *<dir>*
mounts the source dump *<dump>* to mount point *<dir>* and generates a virtual target dump file instead of writing the content to standard output. The virtual dump file is named *<dump>*.*<FMT>*, where *<FMT>* is the name of the specified dump format (see the **`--fmt`** option).

-u *<dir>* or --umount *<dir>*
unmounts the dump that is mounted at mount point `<dir>`. You can specify the dump itself instead of the directory, for example `/dev/dasdd1`. This option is a wrapper for **`fusermount -u`**.

-i *<dump>* or --info *<dump>*
displays the dump header information from the dump and performs a validity check.

-d *<dumpdevice>* or --device *<dumpdevice>*
checks whether the specified ECKD, FBA, SCSI disk, or NVMe disk device contains a valid dump tool and prints information about it.

<dumpdevice> can be the device node of a DASD device (for example `/dev/dasdb`), an NVMe disk device node (for example, `/dev/nvme0n1`), or a multipath device node of a SCSI disk (for example, `/dev/mapper/36005076303ffd4010000000000020c0-part1`).

-V or --verbose
Shows the detailed memory-map layout when printing the dump header information for s390_ext and ELF dump formats.

-h or --help
displays the help information for the command.

-v or --version
displays the version information for the command.

Using zgetdump to copy a dump

Assuming that the dump is on DASD partition `/dev/dasdb1` and that you want to redirect the command output from `stdout` to a file named `dump.elf`:

```
# zgetdump /dev/dasdb1 > dump.elf
```

The following command writes directly to `dump.elf`:

```
# zgetdump /dev/dasdb1 dump.elf
```

Using zgetdump to transfer a dump with ssh

Assuming that the dump is on DASD partition `/dev/dasdd1` and that you want to transfer it to a file on another system with ssh:

```
# zgetdump /dev/dasdd1  | ssh user@host "cat > dump.elf"
```

Using zgetdump to transfer a dump with FTP

Assuming that you want to use FTP to transfer a dump to a file, `dump.elf`, on another system:

1. Establish an FTP session with the target host and log in.
2. To transfer a file in binary mode, enter the FTP **`binary`** command:

   ```
   ftp> binary
   ```

3. To send the dump file to the FTP host issue:

   ```
   ftp> put |"zgetdump /dev/dasdb1" dump.elf
   ```

Using zgetdump to copy a multi-volume dump

Assuming that the dump is on DASD devices `/dev/dasdc` and `/dev/dasdd` spread along partitions `/dev/dasdc1` and `/dev/dasdd1`, and that you want to copy it to a file named `dump.elf`:

```
# zgetdump /dev/dasdc > dump.elf
```

For an example of the output from this command, see Chapter 4, “Using DASD devices for multi-volume dump,” on page 21.

Using zgetdump to copy a tape dump

Assuming that the tape device is `/dev/ntibm0`:

```
# zgetdump /dev/ntimb0 > dump.elf
Format Info:
  Source: s390tape
  Target: elf

Copying dump:
  00000000 / 00001024 MB
  00000171 / 00001024 MB
  00000341 / 00001024 MB
  00000512 / 00001024 MB
  00000683 / 00001024 MB
  00000853 / 00001024 MB
  00001024 / 00001024 MB

Success: Dump has been copied
```

Using zgetdump to create a dump from a live system

To store an ELF-format dump from a live system in a file called `dump.elf` issue:

```
# nice -n -20 zgetdump /dev/crash > dump.elf
```

For an example of the output from this command, see “Creating a kernel dump on a live system” on page 51.

Checking whether a tape dump is valid, and printing the dump header

Assuming that the tape device is /dev/ntibm0:

```
# zgetdump -i /dev/ntibm0
Checking tape, this can take a while...
General dump info:
  Dump format........: s390tape
  Version............: 5
  Dump created.......: Tue, 19 Jul 2022 17:26:46 +0200
  Dump ended.........: Tue, 19 Jul 2022 17:27:58 +0200
  Dump CPU ID........: ff00012320948000
  Build arch.........: s390x (64 bit)
  UTS node name......: mylnxsys
  UTS kernel release.: 5.14.21
  UTS kernel version.: #1 SMP #1 SMP Wed May 10 12:18:48 UTC 2023
  System arch........: s390x (64 bit)
  CPU count (online).: 2
  CPU count (real)...: 2
  Dump memory range..: 1024 MB
  Real memory range..: 1024 MB

Memory map:
  0000000000000000 - 000000003fffffff (1024 MB)
```

Checking whether a DASD dump is valid and printing the dump header

Assuming that the dump is on a partition, part1, of a DASD device /dev/dasdb1:

```
# zgetdump -i /dev/dasdb1
General dump info:
  Dump format........: s390
  Version............: 5
  Dump created.......: Wed, 20 Jul 2022 11:14:33 +0100
  Dump ended.........: Wed, 20 Jul 2022 11:14:46 +0100
  Dump CPU ID........: ff00012320978000
  UTS node name......: mylnxsys
  UTS kernel release.: 5.14.21
  UTS kernel version.: #1 SMP #1 SMP Wed May 10 12:18:48 UTC 2023
  Build arch.........: s390x (64 bit)
  System arch........: s390x (64 bit)
  CPU count (online).: 3
  CPU count (real)...: 3
  Dump memory range..: 1024 MB
  Real memory range..: 1024 MB

Memory map:
  0000000000000000 - 000000003fffffff (1024 MB)
```

Checking whether a SCSI dump is valid and printing the dump header

Assuming that the dump is on the first partition of a SCSI disk, for example /dev/mapper/36005076303ffd4010000000000000020c0-part1:

```
# zgetdump -i /dev/mapper/36005076303ffd4010000000000000020c0-part1
General dump info:
  Dump format........: elf
  Version............: 1
  UTS node name......: r3545010
  UTS kernel release.: 5.14.21
  UTS kernel version.: #1 SMP #1 SMP Wed May 10 12:18:48 UTC 2023
  System arch........: s390x (64 bit)
  CPU count (online).: 3
  Dump memory range..: 1024 MB

Memory map:
  0000000000000000 - 000000003fffffff (1024 MB)
```

Checking whether an NVMe dump is valid and printing the dump header

Assuming that the dump is on the first partition on name space 1 of an NVMe disk, for example /dev/nvme0n1p1:

```
# zgetdump -i /dev/nvme0n1p1
General dump info:
  Dump format........: elf
  Version............: 1
  UTS node name......: t83lp49
  UTS kernel release.: 5.14.21
  UTS kernel version.: #1 SMP Wed May 10 12:18:48 UTC 2023
  System arch........: s390x (64 bit)
  CPU count (online).: 64
  Dump memory range..: 8192 MB

Memory map:
  0000000000000000 - 00000001ffffffff (8192 MB)
```

Checking whether a DASD device contains a valid dump record

Checking DASD device /dev/dasdb, which is a valid dump device:

```
# zgetdump -d /dev/dasdb
Dump device info:
  Dump tool.........: Single-volume DASD dump tool
  Version...........: 2
  Architecture......: s390x (64 bit)
  DASD type.........: ECKD
  Dump size limit...: none
```

Checking DASD device /dev/dasdc, which is not a valid dump device:

```
# zgetdump -d /dev/dasdc
zgetdump: No dump tool found on "/dev/dasdc"
```

Checking whether a SCSI disk contains a valid dump record

Checking SCSI multipath device /dev/mapper/36005076303ffd40100000000000020c0, which is a valid dump device:

```
# zgetdump -d /dev/mapper/36005076303ffd40100000000000020c0
Dump device info:
Dump tool.........: Single-volume SCSI dump tool
Version...........: 1
Architecture......: s390x (64 bit)

Partition info:
Partition number..: 1
Maximum dump size.: 20473 MB
```

Checking SCSI multipath device /dev/mapper/36005076307ffc5e300000000000084cf, which is not a valid dump device:

```
# zgetdump -d /dev/mapper/36005076307ffc5e300000000000084cf
zgetdump: No dump tool found on "/dev/mapper/36005076307ffc5e300000000000084cf"
```

Checking whether an NVMe disk contains a valid dump record

Checking the first partition of name space 1 of an NVMe disk /dev/nvme0n1, which is a valid dump device:

```
# zgetdump -d /dev/nvme0n1
Dump device info:
  Dump tool.........: Next Generation (NGDump) dump tool
  Version...........: 1
  Architecture......: s390x (64 bit)

Partition info:
  Partition number..: 1
```

Checking name space two of the NVMe disk, which is not a valid dump device:

```
# zgetdump -d /dev/nvme0n2
zgetdump: No dump tool found on "/dev/nvme0n2"
```

Using the mount option

Mounting is useful for multi-volume DASD dumps. After a multi-volume dump has been mounted, it is shown as a single dump file that can be accessed directly with dump processing tools such as **crash**.

The following example mounts a multi-volume source DASD dump as an ELF dump, processes it with **crash**, and unmounts it with **zgetdump**:

```
# zgetdump -m /dev/dasdx /dumps
# crash  /dumps/dump.elf /boot/vmlinux-5.14.<xx>-<y>-default.gz
# zgetdump -u /dumps
```

Mounting can also be useful when you want to process the dump with a tool that cannot read the original dump format. Use the **--fmt** option to mount the dump with a format other then the default format.

Selecting data from a dump that includes a crashed kdump

The following example mounts dump data for a crashed production system from a DASD backup dump for a failed kdump (see “Failure recovery and backup tools” on page 10 for details).

```
# zgetdump -s prod -m /dev/dasdb1 /mnt
```

Checking whether a dump has captured two systems

A dump can contain data from two systems. To check for this use **zgetdump -i**, for example, assuming that the previous dump example contains both a dump from the production system and a kdump kernel dump:

```
# zgetdump -i /dev/mapper/36005076303ffd40100000000000020c0-part1
zgetdump: The dump contains "kdump" and "production system"
          Access "production system" with "-s prod"
          Access "kdump" with "-s kdump"
          Send both dumps to your service organization
```

Working with dumps of KVM guests in IBM Secure Execution mode

Secure images can be built to support encrypted dumps with **virsh dump**. A cryptographic key, the customer communication key, must then be specified with the command that builds the image (**genprotimg**). You need this key to read the dump of a secure guest. For more information, see Chapter 9, “Using virsh dump to create dumps on KVM hosts,” on page 49.

In the following examples, `se-dump.elf` is a file with an encrypted dump and `cck` is the file with the required key. Both files are in the current working directory.

- Use the **zgetdump** `-i` option to confirm that `se-dump.elf` is a valid dump.

  ```
  # zgetdump -i se-dump.elf
  General dump info:
    Dump format........: ELF (protected virtualization dump)
    System arch........: s390x (64 bit)

  zgetdump: No key for dump decryption provided.
  Try 'zgetdump --help' for more information.
  ```

  The (`protected virtualization dump`) in the output confirms that the dump is for a KVM guest in IBM Secure Execution mode.

- Specify the key to display more information.

```
# zgetdump -i se-dump.elf --key cck
General dump info:
  Dump format........: ELF (protected virtualization dump)
  Version............: 1
  UTS node name......: lnxlp05
  UTS kernel release.: 5.14.21
  UTS kernel version.: #1 SMP Wed May 10 12:18:48 UTC 2023
  System arch........: s390x (64 bit)
  CPU count (online).: 2
  Dump memory range..: 1024 MB

Memory map:
  0000000000000000 - 000000003fffffff (1024 MB)
```

- You can create a decrypted copy of the dump file, and then proceed working with the decrypted version.

```
# zgetdump se-dump.elf se-dump.decrypted.elf --key cck
```

  Protect the decrypted dump from unauthorized access.

- As an alternative to creating a decrypted dump file, you can mount the dump. A mounted dump is decrypted incrementally, as it is accessed. Especially for large dumps, this method can be advantageous over first creating a decrypted copy of the dump file. In the example, the mount point is a directory /dumps in the file system.

```
# zgetdump -m se-dump.elf /dumps --key cck
```

  Analyze the mounted dump with crash, as usual.

```
# crash vmlinux /dumps/dump.elf
```

  Unmount the dump when you have completed your work.

```
# zgetdump -u /dumps
```

dumpconf - Configure panic or PSW restart action

The **dumpconf** service configures the action to be taken if a kernel panic or PSW restart occurs.

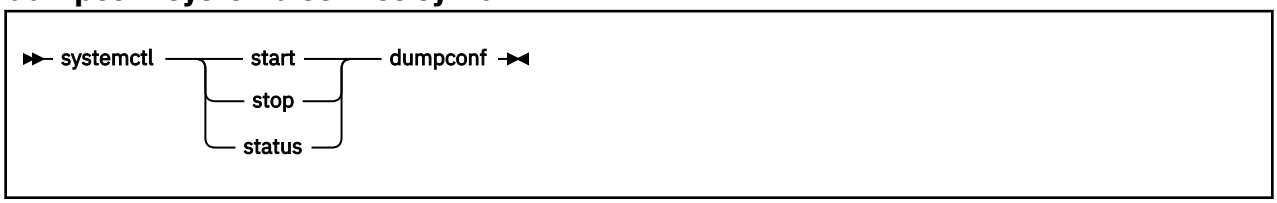
The service is installed as a systemd unit and reads the configuration file `/etc/sysconfig/dumpconf`.

Note: kdump does not depend on **dumpconf** and cannot be enabled or disabled with **dumpconf**. If kdump was set up for your production system, dump tools as configured with **dumpconf** are used only if the integrity check for kdump fails. With kdump set up, you can use **dumpconf** to enable or disable backup dump tools. See also "Failure recovery and backup tools" on page 10.

To enable the **dumpconf** service, issue:

```
# systemctl enable dumpconf
```

dumpconf systemd service syntax



Parameters

start
enables the configuration that is defined in `/etc/sysconfig/dumpconf`.

stop
stops the **dumpconf** service.

status
shows current configuration status of the **dumpconf** service.

Keywords for the configuration file

ON_PANIC
Shutdown action to be taken if a kernel panic or PSW restart occurs. Possible values are:

dump
dumps Linux and stops the system.

reipl
reboots Linux.

dump_reipl
dumps Linux and reboots the system.

vmcmd
executes the specified CP commands and stops the system.

stop
stops Linux (default).

DELAY_MINUTES
specifies the number of minutes that the activation of the **dumpconf** service is to be delayed. The default is zero.

Using **reipl** or **dump_reipl** actions with **ON_PANIC** can lead to the system looping with alternating IPLs and crashes. Use **DELAY_MINUTES** to prevent such a loop. **DELAY_MINUTES** delays activating the specified panic action for a newly started system. After the specified time elapses, the **dumpconf** service activates the specified panic action. This action is taken should the system subsequently

crash. If the system crashes before the time elapses, the previously defined action is taken. If no previous action was defined, the default action (STOP) is performed.

VMCMD_<*X*>
specifies a CP command, *<X>* is a number from one to eight. You can specify up to eight CP commands that are executed in case of a kernel panic or PSW restart. z/VM commands, device addresses, and names of z/VM guest virtual machines must be uppercase.

DUMP_TYPE
specifies the type of dump device. Possible values are **ccw**, **fcp**, and **nvme**.

DEVICE
specifies the device bus-ID for CCW or SCSI dump device.

WWPN
specifies the WWPN for SCSI disk.

LUN
specifies the LUN for SCSI disk.

FID
Function ID for NVMe dump device.

NSID
Name space ID for NVMe dump device.

BOOTPROG
specifies the boot program selector.

BR_LBA
specifies the boot record logical block address.

Example configuration files for the dumpconf service

- Example configuration for a CCW dump device (DASD) using **reipl** after dump and **DELAY_MINUTES**:

```
ON_PANIC=dump_reipl
DUMP_TYPE=ccw
DEVICE=0.0.4714
DELAY_MINUTES=5
```

- Example configuration for FCP dump device (SCSI disk):

```
ON_PANIC=dump
DUMP_TYPE=fcp
DEVICE=0.0.4711
WWPN=0x5005076303004712
LUN=0x4713000000000000
BOOTPROG=0
BR_LBA=0
```

- Example configuration for an NVMe dump device (NVMe disk):

```
ON_PANIC=dump
DUMP_TYPE=nvme
FID=0x00000001
NSID=0x00000001
BOOTPROG=0
BR_LBA=0
```

- Example configuration for re-IPL if a kernel panic or PSW restart occurs:

```
ON_PANIC=reipl
```

- Example of sending a message to the z/VM guest virtual machine "OPERATOR", executing a **CP VMDUMP** command, and rebooting from device 4711 if a kernel panic or PSW restart occurs:

```
ON_PANIC=vmcmd
VMCMD_1="MSG OPERATOR Starting VMDUMP"
VMCMD_2="VMDUMP"
VMCMD_3="IPL 4711"
```

z/VM commands, device addresses, and names of z/VM guest virtual machines must be uppercase.

Examples for using the dumpconf service

Use the **dumpconf** service to enable and disable the configuration.

- To enable the configuration:

```
# systemctl start dumpconf
```

- To display the status:

```
# systemctl status dumpconf
| dumpconf.service - Configure dump on panic
   Loaded: loaded (/usr/lib/systemd/system/dumpconf.service...
   Active: active (exited) since Mon 2017-11-06 20:47:15 CET; 1s ago
   Process: 12200 ExecStart=/lib/s390-tools/dumpconf start (code=exited, status=0/SUCCESS)
 Main PID: 12200 (code=exited, status=0/SUCCESS)

Nov 06 20:47:15 s3545002 systemd[1]: Starting Configure dump on panic ...
Nov 06 20:47:15 s3545002 dumpconf[12200]: dump_reipl on panic configured: Using ccw...
Nov 06 20:47:15 s3545002 systemd[1]: Started Configure dump on panic ...
```

- To disable dump on panic:

```
# systemctl stop dumpconf
```

- To display the status again and check that the status is now stopped:

```
# systemctl status dumpconf
| dumpconf.service - Configure dump on panic
   Loaded: loaded (/usr/lib/systemd/system/dumpconf.service...
   Active: inactive (dead)
...
Nov 06 20:49:33 s3545002 systemd[1]: Stopping Configure dump on panic ...
Nov 06 20:49:33 s3545002 dumpconf[12217]: Dump on panic is disabled now
Nov 06 20:49:33 s3545002 systemd[1]: Stopped Configure dump on panic ...
```

crash - Analyze kernel dumps

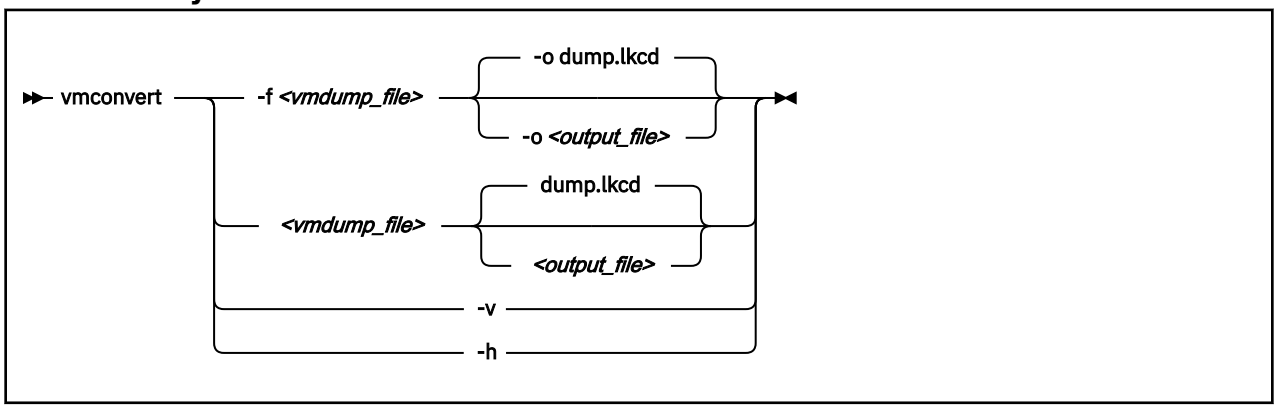
The **crash** tool is a GPL-licensed tool that is maintained by Red Hat. For more information, see the tool online help.

As of crash 7.2.6, you can analyze dumps of kernels with enabled KASLR support. Use crash with the --kaslr auto option. KASLR requires that the dump contains vmcoreinfo, which is always included with kdump. For all other dump types, such as VMDUMP, stand-alone dumps, and qemu dumps, convert the dump to an ELF dump before using crash. To convert the dump, use the command zgetdump -f elf.

vmconvert - Convert z/VM VMDUMPS for Linux

The **vmconvert** tool converts a dump that was created with VMDUMP into a file that can be analyzed with **crash**.

vmconvert syntax



Parameters

<vmdump_file>* or -f *<vmdump_file>* or --file *<vmdump_file>
specifies the VMDUMP created dump file to be converted.

<output_file>* or -o *<output_file>* or --output *<output_file>
specifies the name of the dump file to be created. The default is `dump.lkcd`.

-v or --version
displays the tool version.

-h or --help
displays the help information for the command.

Example

To convert a VMDUMP-created dump file `vmdump1` into a dump file `dump1.lkcd` that can be processed with **crash** issue:

```
# vmconvert -f vmdump1 -o dump1.lkcd
```

You can also use positional parameters:

```
# vmconvert vm.dump lkcd.dump
vmdump information:
  architecture: 32 bit
  date.........: Fri Feb 18 11:06:45 2005
  storage......: 16 MB
  cpus.........: 6
   16 of 16 |################################################################| 100%
'lkcd.dump' has been written successfully.
```

vmur - Receive dumps from the z/VM reader

The **vmur** command can receive a VMDUMP file from the z/VM reader and convert it into a file that can be analyzed with **crash**.

Issue a command of the following form:

```
# vmur receive -c <spool ID> <dump file name>
```

Parameters

<spool ID>
specifies the VMDUMP file spool ID.

<dump file name>
specifies the name of the output file to receive the reader spool file's data.

For more details, see the **vmur** man page and *Device Drivers, Features, and Commands on SUSE Linux Enterprise Server 15 SP5*, SC34-2784

Example

The vmur commands access the reader device, which has to be online. To set it online, it might need to be freed from cio_ignore:

```
# cio_ignore -r c
# chccwdev -e c
Setting device 0.0.000c online
Done
```

To receive and convert a VMDUMP spool file with spool ID 463 to a file named `dump.lkcd` on the Linux file system in the current working directory:

```
# vmur rec -c 463 dump.lkcd
```

zhmc - Triggering dumps remotely through the HMC Web Services API

You can use the HMC Web Services API to trigger dumps for Linux in LPAR mode or in a DPM partition. For information about the API, see *Hardware Management Console Web Services API* for your IBM Z or LinuxONE hardware.

The `zhmc` command is the user interface for a client application that uses this API. For the command syntax and more information about this application, see https://github.com/zhmcclient/zhmccli.

Hint: The `zhmc` command is case sensitive. For hardware and partition specifications, use the capitalization as shown in the HMC interface and the corresponding HMC API queries.

Example for triggering a DPM partition dump

The following example makes these assumptions about the hardware system, LPAR, and boot device:

- The name of the IBM Z or LinuxONE system is T46.
- The name of the DPM partition is `t46lp79`.
- The UUID of the dump volume is `0x6005076309FFD43500000000000078DA`

For the example, follow these steps:

1. Optional: Establish a console connection for viewing the progress and error information of the dump process.

```
# zhmc partition console T46 t46lp79
```

2. Start the dump process. If applicable, use a terminal session other than the session that displays the console messages.

```
# zhmc partition dump --volume 6005076309FFD43500000000000078DA T46 t46lp79
```

Example for triggering an LPAR dump to a SCSI dump device

The following example makes these assumptions about the hardware system, LPAR, and boot device:

- The name of the IBM Z or LinuxONE system is S35.
- The name of the LPAR is S35LP54.
- An FC-attached SCSI disk is prepared, with zipl, as a dump device.
- The LUN of the disk is `0x4712000000000000`.
- The disk is accessed through WWPN `0x4712076300ce93a7`.
- The FCP device to access the disk has a bus ID `0.0.4711`.

For the example, follow these steps:

1. Optional: Establish a console connection for viewing the progress and error information of the dump process:

```
# zhmc lpar console S35 S35LP54
```

2. Start the dump process. If applicable, use a terminal session other than the session that displays the console messages.

```
# zhmc lpar scsi-dump S35 S35LP54 4711 4712076300ce93a7 4712000000000000
```

Appendix C. Installing the DASD or SCSI dump tool with YaST

You can prepare ECKD DASD and SCSI dump devices with the YaST tool.

Procedure

1. Start YaST:

```
# yast
```

2. Ensure that the dump DASDs or SCSI devices (LUNs) are activated.
   a) Enter menu: **Hardware** > **DASD** or **Hardware** > **Zfcp**
   b) Activate the correct DASDs or SCSI disks, if not already activated.
3. Prepare the DASDs or SCSI device for dump use:
   a) Enter menu: **Hardware** > **Dump Devices**.
   b) Select the dump DASDs or SCSI device.
   c) Click **Create** to prepare the DASDs or SCSI disk.
4. A message is displayed when your dump device has been prepared.

What to do next

After a dump device has been prepared it is possible to configure the **dumpconf** service for automatic dump on panic:

1. Enter menu: **Hardware** > **On Panic**.
2. Configure the **dumpconf** settings, see “dumpconf - Configure panic or PSW restart action” on page 73.

Note: The dump-related YaST dialogues can also be entered directly with these commands:

- **yast dasd**
- **yast zfcp**
- **yast kdump**
- **yast dump**
- **yast onpanic**

Accessibility

Accessibility features help users who have a disability, such as restricted mobility or limited vision, to use information technology products successfully.

Documentation accessibility

The Linux on IBM Z and IBM LinuxONE publications are in Adobe Portable Document Format (PDF) and should be compliant with accessibility standards. If you experience difficulties when you use the PDF file and want to request a Web-based format for this publication send an email to eservdoc@de.ibm.com or write to:

IBM Deutschland Research & Development GmbH
Information Development
Department 3282
Schoenaicher Strasse 220
71032 Boeblingen
Germany

In the request, be sure to include the publication number and title.

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

IBM and accessibility

See the IBM Human Ability and Accessibility Center for more information about the commitment that IBM has to accessibility at

```
www.ibm.com/able
```

Notices

This information was developed for products and services offered in the U.S.A. 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.

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.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml

Adobe is either a registered trademark or trademark of Adobe Systems Incorporated in the United States, and/or other countries.

The registered trademark Linux is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Index

Special Characters

A

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

Z

SC34-2785-01