Linux on System z

Using the Dump Tools on Red Hat Enterprise Linux 6.1

SC34-2607-01

Linux on System z

Using the Dump Tools on Red Hat Enterprise Linux 6.1

SC34-2607-01

Note
Before using this information and the product it supports, read the information in “Notices” on page 47.

This edition applies to Red Hat Enterprise Linux 6.1 on IBM System z, and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC34-2607-00.

Contents

Figures

Tables

Summary of changes

This revision reflects changes to the Development stream for kernel Red Hat Enterprise Linux 6.1.

Updates for Red Hat Enterprise Linux 6.1

This revision (SC34-2607-01) contains changes related to the Red Hat Enterprise Linux 6.1 release.

New Information

- A new keyword, DELAY_MINUTES, has been introduced for the `dumpconf` configuration file to prevent potential panic-IPL-loops when using ON_PANIC with reipl and dump_reipl. See “Keywords for the configuration file” on page 39.
- Using the **makedumpfile** tool, to reduce the size of dump files to be transmitted for problem determination. See Chapter 7, “Handling large dumps,” on page 19.

Changed Information

- The **zgetdump** command has been changed to support new options for mounting and unmounting a dump file and to export a dump in ELF format.

This revision also includes maintenance and editorial changes.

Deleted Information

- The support for multivolume tape dumps has been removed.

About this document

This book describes tools for obtaining dumps of Linux for IBM® System z® instances running Red Hat Enterprise Linux 6.1. This book describes how to use DASD, tape, and SCSI dump devices, as well as how to use VMDUMP.

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

In this document, System z is taken to include all IBM mainframe systems supported by Red Hat Enterprise Linux 6.1 for System z.

You can find the latest version of this document on developerWorks® at

`www.ibm.com/developerworks/linux/linux390/documentation_red_hat.html`

On the same page you can also find:

- *Device Drivers, Features, and Commands on Red Hat Enterprise Linux 6.1*, SC34-2597

Other relevant Linux on IBM System z publications

For each of the following documents, the same web page points to the version that most closely reflects Red Hat Enterprise Linux 6.1:

- *How to use FC-attached SCSI devices with Linux on System z*, SC33-8413
- *How to Improve Performance with PAV*, SC33-8414
- *How to use Execute-in-Place Technology with Linux on z/VM*, SC34-2594
- *How to Set up a Terminal Server Environment on z/VM*, SC34-2596
- *libica Programmer's Reference*, SC34-2602

Chapter 1. Introduction

Different tools can be used for obtaining dumps of Linux on IBM System z instances running Red Hat Enterprise Linux 6.1. This chapter gives an overview of those tools.

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

Table 1 summarizes the available dump tools:

Table 1. Dump tools summary

Tool	Stand-alone tools				VMDUMP
	DASD	**Multivolume DASD**	**SCSI**	**Tape**	
Environment	VM and LPAR	VM and LPAR	VM[1] and LPAR	VM and LPAR	VM only
z/VM NSS	No	No	No	No	Yes
Size	Small, depending on disk size(2)	Large, up to 32 DASD partitions	Large, depending on SCSI disk and what other data it contains	Large, when using actual tape device generation (such as 3592)	Large, depending on the available disk space; the slow dump speed can lead to very long dump times
Speed	Fast	Fast	Fast	Slow	Slow
Medium	ECKD™ or FBA(3) DASD	ECKD DASD	Linux file system on a SCSI disk	Tape cartridges	VM reader
Compression possible	No	No	No	Yes(4)	No
Disruptive(5)	Yes	Yes	Yes	Yes	No

Note:

1. As of z/VM 5.4.
2. ECKD model 27, for example, provides 27 GB.
3. SCSI disks can be emulated as FBA disks. This dump method can, therefore, be used for SCSI-only VM installations.
4. IBM TotalStorage Enterprise Tape System 3592, 3590 and IBM 3490 Magnetic Tape Subsystem offer hardware compression.
5. The dump process kills a running operating system.

Note on device nodes

In all examples, the traditional device nodes for DASD, tape, and SCSI devices are used. You can also use the device nodes that **udev** creates for you.

Stand-alone tools

Four stand-alone dump tools are shipped in the s390utils package as part of **zipl**:

- DASD dump tool for dumps on a single DASD device
- Multivolume 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

You need to install these tools on the *dump device*. The dump device is the device you want to use for dumping the memory.

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 is destructive, that is, the running Linux operating system is killed. The IPL process writes the system memory to the IPL device (DASD and tape) or directly to a file on a SCSI disk.

You can configure a dump device that is automatically used when a kernel panic occurs. For more information, see "The dumpconf service" on page 39.

For more information on **zipl**, refer to the **zipl** man page and to the **zipl** description in *Device Drivers, Features, and Commands on Red Hat Enterprise Linux 6.1*, SC34-2597. You can find the latest version of this document on developerWorks at:

`www.ibm.com/developerworks/linux/linux390/documentation_red_hat.html`

VMDUMP

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 VM guests that use z/VM named saved systems (NSS).

Do not use **VMDUMP** to dump large 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 on **VMDUMP** see *z/VM CP Commands and Utilities Reference*, SC24-6175.

Chapter 2. Using a DASD dump device

This chapter provides information on how to install the stand-alone DASD dump tool, how to perform the dump process, and how to copy the dump to a file in a Linux file system.

DASD dumps are written directly to a DASD partition that has not been formatted with a file system. The following DASD types are supported:

- ECKD DASDs
  - 3380
  - 3390
- FBA DASDs

Installing the DASD dump tool

Requirement: 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, you should use the multivolume dump tool, see Chapter 3, “Using DASD devices for multi-volume dump,” on page 5.

This section describes how to install the DASD dump tool on an unused DASD partition. Dumps are written to this partition.

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

The steps 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:

1. Format your DASD with **dasdfmt** (ECKD only). A block size of 4 KB is recommended:

   Example:

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

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

   Example:

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

3. Install the dump tool using the **zipl** command. Specify the dump device on the command line.

   Example:

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

Note: When using an ECKD-type DASD formatted with the traditional Linux disk layout ldl, the dump tool must be reinstalled using **zipl** after each dump.

Initiating a DASD dump

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 storage!

   The dump process can take several minutes depending on the device type you are using 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
   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
   ```

4. You can IPL Linux again.

See Appendix A, “Examples for initiating dumps,” on page 27 for more details.

Copying the dump from DASD with zgetdump

This section describes how to copy a DASD dump to a file system using the **zgetdump** tool.

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.

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

```
# zgetdump /dev/dasdc1 > dump_file
```

You can also use **zgetdump** to display information about the dump. See “Checking whether a DASD dump is valid and printing the dump header” on page 38 for an example.

For general information about **zgetdump**, see “The zgetdump tool” on page 35 or the man page.

Chapter 3. Using DASD devices for multi-volume dump

This chapter describes how to prepare a set of ECKD DASD devices for a multivolume dump, how to install the stand-alone dump tool on each DASD device involved, how to perform the dump process, and how to copy the dump to a file in a Linux file system.

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

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

You can use any block size, even mixed block sizes. However, to speed up the dump process and to reduce wasted disk space, use block size 4096.

For example, Figure 1 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.

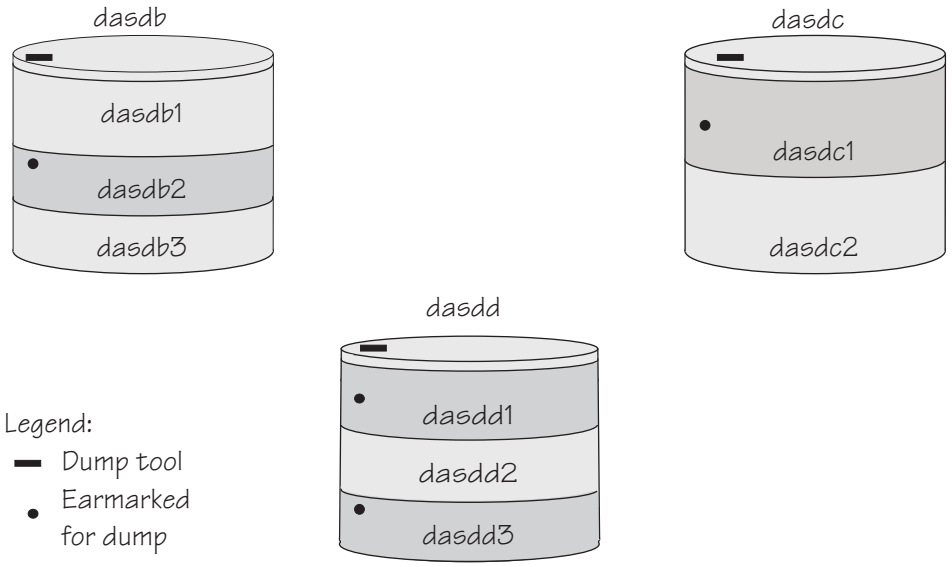


Figure 1. Three DASD volumes with four partitions for a multivolume dump

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

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

You can define a maximum of three partitions on one DASD. All three 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 multivolume 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`.

Assume that the corresponding bus IDs (as displayed by **lsdasd**) are 0.0.4711 and 0.0.4712, so the respective device numbers are 4711 and 4712.

1. Format both dump volumes with **dasdfmt**. Specify **cdl** (compatible disk layout), which is the default. Preferably, use a block size of 4 KB:

```
# 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 `sample_dump_conf` containing the device nodes of the two partitions, separated by one or more line feed characters (0x0a). The file's contents looks 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 sample_dump_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)?
```

Now the two volumes `/dev/dasdc` and `/dev/dasdd` with device numbers 4711 and 4712 are prepared for a multivolume dump. Use the **-device** option of **zgetdump** to display information about these volumes:

```
# zgetdump -d /dev/dasdc
'/dev/dasdc' is part of Version 1 multi-volume dump,
which is spread along the following DASD volumes:
0.0.4711 (online, valid)
0.0.4712 (online, valid)
Dump size limit: none
Force option specified: no
```

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 --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 B, “Obtaining a dump with limited size,” on page 33. For more details on **zipl**, refer to the description of the **zipl** command in the *Device Drivers, Features, and Commands on Red Hat Enterprise Linux 6.1*, SC34-2597.

Initiating a multivolume DASD dump

To obtain a dump with the multivolume 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 4711 or 4712.

   Note: Do not clear storage!

   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: 4711
   00000128 / 00001024 MB
   00000256 / 00001024 MB
   00000384 / 00001024 MB
   00000512 / 00001024 MB
   Dumping to: 4712
   00000640 / 00001024 MB
   00000768 / 00001024 MB
   00000896 / 00001024 MB
   00001024 / 00001024 MB
   Dump successful
   ```

4. You can IPL Linux again.

Copying a multi-volume dump to a file

At this point the two volumes `/dev/dasdc` and `/dev/dasdd` (with device numbers 4711 and 4712) contain the dump. Dump data is spread along partitions `/dev/dasdc1` and `/dev/dasdd1`.

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

```
# zgetdump /dev/dasdc > multi_volume_dump_file
Format Info:
Source: s390mv
Target: s390
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 multivolume 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 38 for an example.

Chapter 4. Using a tape dump device

This chapter provides information on how to install the stand-alone tape dump tool, how to perform the dump process, and how to copy the dump to a file in a Linux file system.

The following tape devices are supported:

- 3480
- 3490
- 3590
- 3592

Installing the tape dump tool

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

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

Perform these steps to install the tape dump tool:

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

   Example:

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

Initiating a tape dump

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 storage!

   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 should go into disabled wait.

   The following PSW indicates that the dump was taken 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
```

5. You can IPL Linux again.

See Appendix A, “Examples for initiating dumps,” on page 27 for more details.

Tape display messages

number
The number of MB dumped.

dump*end
The dump process ended successfully.

Copying the dump from tape

This section describes how to copy a tape dump to a file system using the **zgetdump** tool.

Prerequisite: You must have installed the **mt** utility.

Preparing the dump tape

You need to rewind the tape, and find the correct position on the tape to start copying from. Use the **mt** tool to do this.

1. Rewind the tape.

   Example:

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

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

   Example:

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

Using the zgetdump tool

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.

Example: Assuming that the tape is in the correct position (see “Preparing the dump tape”) and is on tape device `/dev/ntibm0`, use the following command to copy the dump from tape to a file named `dump_file` in the file system:

```
# zgetdump /dev/ntibm0 > dump_file
```

For general information on **zgetdump**, see “The zgetdump tool” on page 35 or the man page.

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

To check whether a dump is valid, use the **-i** option. For example:

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

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

2. Issue:

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

zgetdump goes through the dump until it reaches the end. See also “Using zgetdump to copy a tape dump” on page 37

Chapter 5. Using a SCSI dump device

You can use SCSI disks that are accessed through the **zfcp** device driver as dump devices. SCSI disk dumps are written as files in an existing file system on the dump partition. No copying is necessary.

This section describes how to install the SCSI dump tool and how to initiate a SCSI dump.

Installing the SCSI disk dump tool

Requirements:

- The kernel-kdump RPM (named `kernel-kdump-2.6.32-xx.el6.s390x.rpm`) must be installed on your system.
- The dump directory needs enough free space (memory size + 10 MB) to hold the system memory.

The SCSI dump tool (also referred to as the SCSI Linux System Dumper, or SD) is written to one partition, referred to here as the *target partition*. The dump can be written to a second partition, the *dump partition*, provided it is on the same physical disk. Only the target partition need be mounted when **zipl** is run. In a single-partition configuration, the target partition is also the dump partition.

SCSI dump tool parameters

When installing the SCSI disk dump tool, the following parameters can be specified in a 'parameters' line in the **zipl** configuration file or using the **-P** option in the **zipl** command line.

dump_dir=/*<directory>*

Path to the directory (relative to the root of the dump partition) to which the dump file is to be written. This directory is specified with a leading slash. The directory must exist when the dump is initiated.

Example: If the dump partition is mounted as `/dumps`, and the parameter **dump_dir=/mydumps** is defined, the dump directory would be accessed as `/dumps/mydumps`.

The default is / (the root directory of the partition).

dump_mode=interactive | auto

Action taken if there is no room on the file system for the new dump file. **interactive** prompts the user to confirm that the dump with the lowest number is to be deleted. **auto** automatically deletes this file.

The default is **interactive**.

In rare cases, you might want to complement or overwrite the SCSI dump tool parameters that have been configured with **zipl**. For example, you might want to change the dump mode setting when you initiate the dump. How you specify such parameters depends on whether your Linux instance runs in LPAR mode or as a z/VM guest operating system. For more information, see the SCSI examples in Appendix A, “Examples for initiating dumps,” on page 27.

Example 1: Combined dump and target partition

This example assumes that /dev/sda is a SCSI device that contains no data and is to be used exclusively as a dump device. Because no other data is to be stored on the device, a single partition is created that serves as both dump and target partition.

1. Create a single partition with **fdisk**, using the PC-BIOS layout:

   Example:

   ```
   # fdisk /dev/sda
   ```

   The created partition is /dev/sda1.

2. Format this partition with either the ext2, ext3, or ext4 file system.

   Example:

   ```
   # mke2fs -j /dev/sda1
   ```

3. Mount the partition at a mount point of your choice and create a subdirectory to hold the dump files.

   Example:

   ```
   # mount /dev/sda1 /dumps
   # mkdir /dumps/mydumps
   ```

4. Install the dump tool using the **zipl** command. Specify the dump device on the command line.

   Example:

   ```
   # zipl -D /dev/sda1 -t /dumps -P "dump_dir=/mydumps"
   ```

5. Unmount the file system:

   ```
   # umount /dumps
   ```

When you IPL /dev/sda1 using boot program selector 1 or 0 (default), the dump is written to directory mydumps on partition 1 of /dev/sda. The boot program selector is located on the load panel, see Figure 4 on page 31 for an example.

Initiating a SCSI dump

To initiate the dump, IPL the dump tool using the **SCSI dump** load type. See Appendix A, “Examples for initiating dumps,” on page 27.

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.

The dump process creates a new dump file in the dump directory. All dumps are named dump.*<n>*, where *<n>* is the dump number. A new dump receives the next highest dump number out of all dumps in the dump directory (see the **dump_dir** parameter under “SCSI dump tool parameters” on page 13).

Example: If there are already two dump files named dump.0 and dump.1 in the dump directory, the new dump will be named dump.2.

When the dump completes successfully, you can IPL Linux again.

See Appendix A, “Examples for initiating dumps,” on page 27 for more details.

You do not need to convert the dump or copy it to a different medium. To access the dumps, mount the dump partition.

Printing the dump header

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

```
# zgetdump -i dump.0
General dump info:
Dump format........: lkcd
Version............: 8
System arch........: s390x (64 bit)
CPU count (online).: 2
CPU count (real)...: 2
Dump memory range..: 1024 MB
Memory map:
0000000000000000 - 000000003fffffff (1024 MB)
```

Chapter 6. Using VMDUMP

Do not use **VMDUMP** to dump large 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 VM dump to a convenient format.

VMDUMP does not need to be installed separately.

Intitiating a dump with VMDUMP

Issue the following command from the guest's 3270 console:

```
#CP VMDUMP
```

Result: VM stops the Linux guest and creates a dump file in the guest's VM reader. After the dump is complete, the Linux guest continues operating.

You can use the **TO** option of the **VMDUMP** command to direct the dump to the reader of another guest of the same VM.

Example: To write the dump to a VM guest named `linux02` issue:

```
#CP VMDUMP TO LINUX02
```

For more information on **VMDUMP** refer to *z/VM CP Commands and Utilities Reference*, SC24-6175.

Copying the dump to Linux

You can use the **vmur** command under Linux or the **DUMPLOAD** command under CMS to copy the dump file.

Using the vmur command

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 above 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%
```

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

Using the DUMPLOAD command

Alternatively you can use the **DUMPLOAD** command under CMS to access the dump. The **DUMPLOAD** command copies the dump from the VM reader to the CMS file system.

From the CMS file system, you can then transfer the dump to a Linux file system, for example with **ftp**.

Chapter 7. Handling large dumps

This topic describes how to handle dumps that are especially large (greater than 10 GB in size).

About this task

Large dumps present a challenge as they will:

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

Note: Sometimes you can recreate the problem on a test system with less memory, which makes the dump handling much easier. Take this option into account 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
   - 3390 model 9 (up to 45 GB)
   - 3390 model A (up to 180 GB)

   Multivolume DASD
   Up to 32 DASDs are possible.
   - 32 x 3390 model 9 (up to 1.4 TB)
   - 32 x 3390 model A (up to 5.7 TB)

   z/VM FBA emulated SCSI dump 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 dump
   The SCSI disk size depends on your storage server setup. The ext2 and ext3 file system dump size limit using block size 4 KB is 2 TB. For the ext4 file system, the limit is 16 TB.

   Note: SCSI dump compression (the **dump_compress** option) will create smaller dumps, but due to CPU consumption it slows down the dump speed significantly. Therefore you should use this option on large systems only if dump speed is not important for your scenario.

   Dump on 3592 channel-attached tape drive
   Cartridges with up to 300 GB 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 you have a system with 50 GB memory, the dump will take 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. Send the dump. For transferring dumps in a short amount of time to a service organization, it is often useful to reduce the dump size or split the dump into several parts for easier and faster transmission. In this step, two different mechanisms: 'Use **makedumpfile**' and 'Use **gzip** and **split**' are described. Choose one of these methods.

   a. Use **makedumpfile**

      The **makedumpfile** tool can be used to compress s390 dumps and exclude memory pages that are not needed for analysis. This substantially reduces the size of dump files and the amount of time needed to transmit them from one location to another. For Red Hat Enterprise Linux 6, the **makedumpfile** tool is included in the kexec-tools RPM that you can install, for example, with **yum install kexec-tools**. 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.

      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`

         ```
         # zgetdump -m -f elf /dev/dasdb1 /mnt
         ```

         To mount a SCSI dump from file `dump.0` to `/mnt`

         ```
         # zgetdump -m -f elf dump.0 /mnt
         ```

      2) After mounting the dump in ELF format with **zgetdump**, the dump is available in the file named `/mnt/dump.elf`. In order to use **makedumpfile** with dump level greater than one, you also need the `vmlinux` file that contains necessary debug information. You find this file in the kernel debuginfo RPM. To find the location of the `vmlinux` file in the debuginfo RPM, issue the following commands (the *xx* in the example must be replaced by the appropriate kernel version that caused the dump):

         ```
         # rpm -qlp kernel-debuginfo-2.6.32-xx.el6.s390x.rpm | grep vmlinux
         ```

      3) To extract the `vmlinux` file to `./usr/lib/debug/lib/modules/2.6.32-xx.el6.s390x/`, issue the following command:

         ```
         # rpm2cpio kernel-debuginfo-2.6.32-xx.el6.s390x.rpm | cpio -idv *vmlinux*
         ./usr/lib/debug/lib/modules/2.6.32-xx.el6.s390x/vmlinux
         1079519 blocks
         ```

      4) Use the **-d** (dump level) option of **makedumpfile** to specify which pages to exclude from the dump. See the man page for **makedumpfile** for a description of the dump level and other options of **makedumpfile**.

         This example compresses the dump file named `/mnt/dump.elf` (**-c** option) and excludes pages that are typically not needed to analyze a kernel problem. Excluded pages are: pages containing only zeroes, pages used to cache file contents (cache, cache private), pages belonging to user spaces processes, and free pages (maximum dump level 31):

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

5) The newly created file, named `dump.kdump` should be much smaller than the original file, named `dump.elf`. Until your kernel problem is resolved, it is recommended to keep the original dump file. This will enable you to reduce the dump level, if it turns out that the pages that had been excluded are still needed for problem determination.

6) For initial problem analysis, you can also extract the kernel log with **makedumpfile**, and send it to your service organization:

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

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

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

b. Use **gzip** and **split**

1) Compress the dump and split it into parts of one GB 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:

```
# cat /mnt/dump.0 | gzip | split -b 1G
```

This will create 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 service organization.

4) The receiver (the service organization) must do the following:

a) Verify md5 sums:

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

b) Merge parts and uncompress the dump:

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

Chapter 8. Sharing dump devices

The ideal dump device setup is that each Linux instance has its own dump device. Then every system can be dumped independently at any time and you always have enough dump space. However, if you have many systems you might want to share dump devices due to economical considerations. This chapter describes how you can set up your system for sharing dump devices between Linux instances.

Serialization and device locking

If you share devices, some kind of serialization is needed to prevent two systems from dumping at the same time and thus corrupting the dumps. Either the involved operators must prevent concurrent dumps manually, or, in some cases, available system mechanisms can be used to prevent this.

While it is possible in many cases to use a pool of devices for sharing, for simplicity most of the following examples use only one dump device.

Possible serialization mechanisms:

External
: Operators must find an external way to ensure serialization manually.

Link
: Exclusive write for minidisk is used as a locking mechanism (see “Sharing DASD devices under z/VM” on page 24).

Attach
: Attach and detach is used as locking mechanism (see “Using attach and detach as locking mechanism under z/VM” on page 24).

vmcmd
: Use the **vmcmd** panic action (see “DASD (vmcmd panic action)” on page 25).

Alternatively, use no serialization and take the risk that dumps are overwritten, see “DASD (dump or dump_reipl panic action)” on page 25).

Table 2 shows the serialization methods available for different system configurations.

Table 2. Serialization of dump devices overview

	DASD		**SCSI**	
	z/VM	LPAR	z/VM	LPAR
Manual dump	link, attach, external	external	attach, external	external
Automatic dump	overwrite, vmcmd	overwrite	N/A	N/A

Sharing devices when dumping manually

In the following sections, it is assumed that you start the dump process manually, without using automatic dump on panic.

Sharing DASD devices on LPARs

Configure your IOCDS so that all LPARs that want to share the dump device can access the DASD device. There is no system mechanism available for serialization. Exclusive access must be ensured manually by the involved system operators.

Sharing DASD devices under z/VM

Under z/VM, DASD devices can be shared if they are defined as sharable minidisks for a **NOLOG** user. Exclusive access can be guaranteed by the **link** CP command using the exclusive write mode. Because with this mode only one DASD can be linked to one VM guest at the same time, the dump device will be locked for other systems until it is detached.

To create a dump after a system crash, do the following:

1. To link the dump device, issue a command of the form:

   ```
   #cp link <disk owner> <vdev1> <vdev2> EW
   ```

   where

   - *<disk owner>* is the user ID in the system directory whose entry is to be searched for device *<vdev1>*.
   - *<vdev1>* is the specified user's virtual device number.
   - *<vdev2>* is the virtual device number that is to be assigned to the device for your virtual machine configuration.
2. Create the dump using device *<vdev2>*
3. Reboot your Linux system.
4. On your Linux system, set dump device *<vdev2>* online.
5. On your Linux system, copy the dump using **zgetdump**.
6. On your Linux system, set dump device *<vdev2>* offline.
7. Detach the dump device:

   ```
   #cp detach <vdev2>
   ```

Afterwards the dump DASD is free again and can be used by other systems.

Sharing SCSI devices

If you want to share FCP attached SCSI disks for dump, they have to be accessible through your SAN on all Linux systems that want to use the dump device. The involved operators must ensure manually that two dumps are not taking place at the same time. Otherwise, if multiple Linux systems write to the shared dump device at the same time, you may not only corrupt the dump file but also the file system on the dump device.

Using attach and detach as locking mechanism under z/VM

When the Linux guests that use the shared dump device have the permission to attach devices (that is, class B guests) this can also be used as a locking mechanism. Only one guest can attach a device at the same time. If you use one single FCP adapter for dump on all systems, attach and detach can be also be used as locking mechanism for SCSI dump.

Sharing devices when dumping automatically

You can configure a dump to be taken automatically should a kernel panic occur. The automatic dump on panic can be configured in `/etc/sysconfig/dumpconf` (see “The dumpconf service” on page 39).

DASD (dump or dump_reipl panic action)

Technically, it is possible to share DASD devices for automatic dump on panic. However, there is no serialization mechanism available, which means that if two systems are dumping at the same time, your dumps might be corrupted.

Normally, system crashes are quite rare and therefore the chance of corrupt dumps is low, but you have to consider carefully if this is an acceptable risk. Such a dump setup is a trade-off between reliability and resource expenses. You have to consider the likelihood of two concurrent system crashes and the business impact of loosing a dump.

To share DASDs under z/VM, you must use minidisks that must be linked in access mode multiple-write (MW) to all systems where you want to configure dump on panic.

DASD (vmcmd panic action)

Under z/VM when using the panic action **vmcmd** in `/etc/sysconfig/dumpconf`, it is possible to specify up to five CP commands that are executed in case of a kernel panic. You can use this mechanism to implement locking through the exclusive link or attach method.

Example:

In this example, assume that we want to link either 4e1 or 4e2 as device number 5000 and then create the dump using device 5000. The first free DASD will be linked. If both devices are already linked to other guests, the system will stop without creating a dump.

Before you begin: Define minidisks 4e1 and 4e2 with disk owner user **SHARDISK** and prepare them as dump DASDs.

The corresponding configuration for `/etc/sysconfig/dumpconf` looks like this:

```
ON_PANIC=vmcmd
VMCMD_1="LINK SHARDISK 4E1 5000 EW"
VMCMD_2="LINK SHARDISK 4E2 5000 EW"
VMCMD_3="STORE STATUS"
VMCMD_4="IPL 5000"
```

After the dump process has finished, you must perform an IPL on the Linux system manually, copy the dump, and detach the disk 5000.

Compared to “DASD (dump or dump_reipl panic action),” this option has the advantage that you cannot get corrupted dumps and you can use more than one dump device. It has the disadvantage that automatic re-IPL is not possible.

FCP-attached SCSI devices

For automatic dump on a FCP-attached SCSI device, device sharing should not be used. Otherwise, if multiple Linux systems write to the shared dump device at the same time, you may not only corrupt the dump file but also the file system on the dump device.

Sharing dump devices between different Linux versions

Do not share dump devices between Linux installations with different major releases. For example, you should not share dump devices between Red Hat Enterprise Linux 5 and Red Hat Enterprise Linux 6.

You can share dump devices between Linux installations with different service levels. Prepare the dump device with the **zipl** tool from the lowest service level. For example, if you have systems with Red Hat Enterprise Linux 6 and Red Hat Enterprise Linux 6.1, you should prepare your dump device using the **zipl** tool from Red Hat Enterprise Linux 6. Newer tools such as **zgetdump** or dump analysis tools such as **crash** always can process dumps that have been created with older **zipl** versions. The other way around might work, but it is not guaranteed to work.

Sharing dump resources with VMDUMP

Under z/VM you can use **VMDUMP** concurrently on different guests. Note that the dump speed is slow and therefore is best for very small systems only. The shared resource here is the z/VM spool. You have to ensure that it has enough space to hold multiple **VMDUMP**s.

Appendix A. Examples for initiating dumps

The following sections describe how to initiate a dump from different control points.

z/VM

The following examples assume the 64-bit mode. Corresponding 31-bit examples would have a different PSW but be the same otherwise.

Using DASD

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.

Using tape

If 193 is the tape device:

```
#cp rewind 193
#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.

Using SCSI

Prerequisite: SCSI dump from z/VM is supported as of z/VM 5.4.

Assume your SCSI dump disk has the following parameters:

- WWPN: 4712076300ce93a7
- LUN: 4712000000000000
- FCP adapter device number: 4711
- Boot program selector: 3

To initiate the dump process, follow these steps:

```
#cp set dumpdev portname 47120763 00ce93a7 lun 47120000 00000000 bootprog 3
#cp ipl 4711 dump
```

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

You can now IPL your Linux instance and resume operations.

In rare cases, you might want to overwrite or complement the existing SCSI dump tools parameters that have been configured with **zipl**. For example, you might want to change the dump mode setting. You can use a command of this form to specify SCSI dump tools parameters to be concatenated to the existing parameters:

```
#cp set dumpdev scpdata '<parameters>'
```

Enter this command before entering the **IPL** command.

In contrast to SCSI IPL configurations, where you can use a leading equal sign to replace all kernel parameters, you cannot use a leading equal sign to replace all SCSI dump tool parameters. Specifying the parameters with a leading equal sign causes the dump to fail.

Using VMDUMP

To initialize a dump with **VMDUMP**, issue this command from your Linux guest's 3270 console:

```
#cp vmdump
```

Dumping does not force you to IPL. If the Linux instance ran as required before dumping, it will continue running when the dump is completed.

HMC or SE

You can initiate a dump process on an LPAR from an HMC (Hardware Management Console) or SE (Support Element). The following description refers to an HMC, but the steps also apply to an SE.

The steps are similar for DASD, tape, and SCSI. Differences are noted where applicable. You cannot initiate a dump with **VMDUMP** from the HMC or SE.

To initiate the dump:

1. In the left navigation pane of the HMC, expand Systems Management and Servers and select the mainframe system you want to work with. A table of LPARs is displayed in the upper content area on the right.
2. Select the LPAR for which you want to initiate the dump.
3. In the **Tasks** area, expand **Recovery**. Proceed according to your dump device:
   - If you are dumping to DASD or tape, click **Stop all** in the **Recovery** list to stop all CPUs. Confirm when you are prompted to do so.
   - If you are dumping to a SCSI disk, skip this step and proceed with step 4 on page 29.

Figure 2 shows an example of an HMC with a selected mainframe system and LPAR. The **Load** and **Stop all** tasks can be seen in the expanded **Recovery** list.

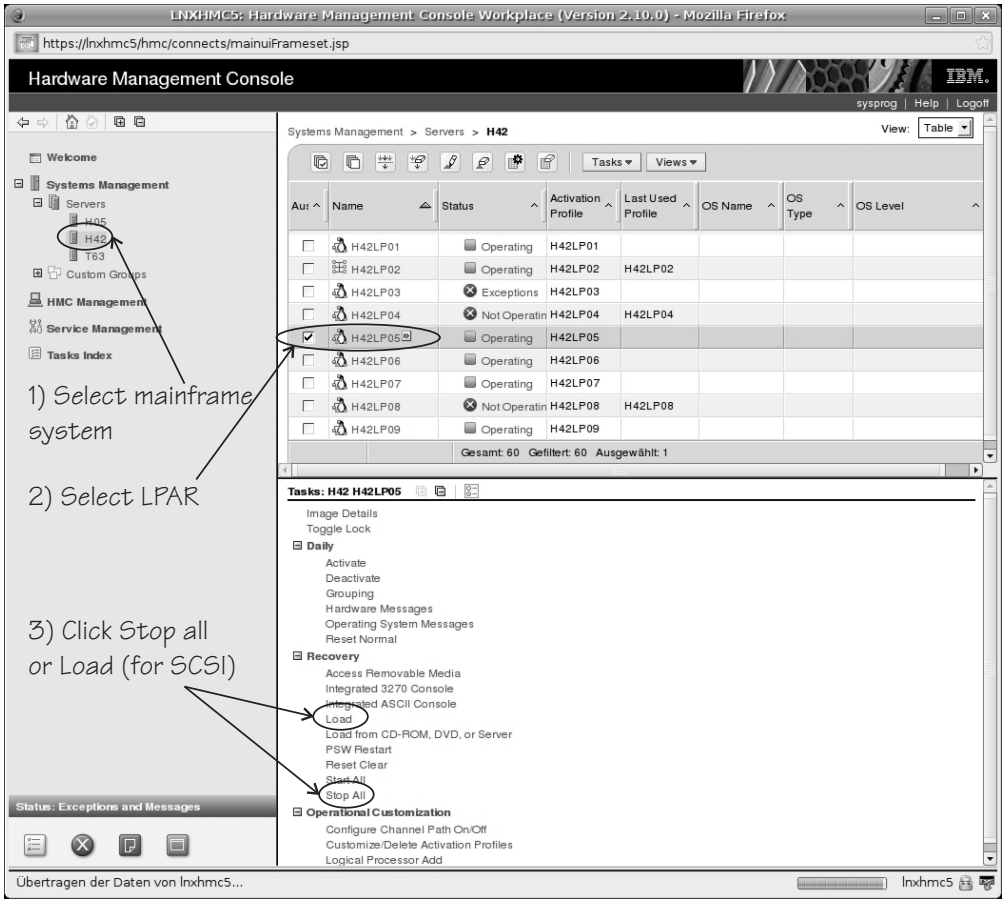


Figure 2. HMC with the **Load** *and* **Stop all** *tasks*

4. Click **Load** in the **Recovery** list to display the Load panel.

   For a dump to DASD or tape:

   a. Select **Load type** “Normal”.

   b. Select the **Store status** check box.

   c. Type the device number of the dump device into the **Load address** field.

   Figure 3 on page 30 shows a Load panel with all entries and selections required to start the dump process for a DASD or tape dump device.

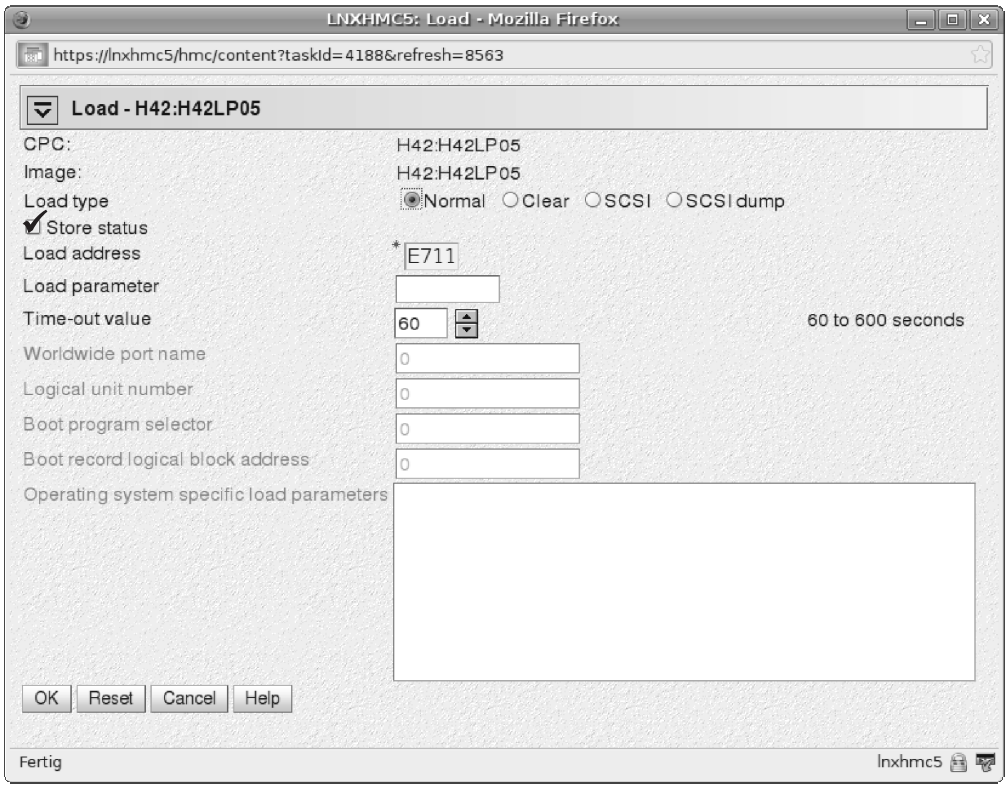


Figure 3. Load panel for dumping to DASD or tape

For a dump to SCSI disk:

a. Select **Load type** "SCSI dump".

b. Type the device number of the FCP adapter for the SCSI disk into the **Load address** field.

c. Type the World Wide Port name of the SCSI disk into the **World wide port name** field.

d. Type the Logical Unit Number of the SCSI disk into the **Logical unit number** field.

e. Type the configuration number of the dump IPL configuration in the **Boot program selector** field.

   The 'configuration number' defines the IPL or dump configuration which is to be IPLed. The numbering starts with 1 and is related to the menu of IPL/dump entries in the **zipl** configuration file for the SCSI disk. Configuration number 0 specifies the default configuration.

f. Accept the defaults for the remaining fields.

   In rare cases, you might want to overwrite or complement the existing SCSI dump tools parameters that have been configured with **zipl**. For example, you might want to change the dump mode setting. In the **Operating system specific load parameters** field, you can specify SCSI dump tools parameters to be concatenated to the existing parameters.

   In contrast to SCSI IPL configurations, where you can use a leading equal sign to replace all kernel parameters, you cannot use a leading equal sign to replace all SCSI dump tool parameters. Specifying the parameters with a leading equal sign causes the dump to fail.

Figure 4 shows a Load panel with all entries and selections required to start the SCSI dump process.

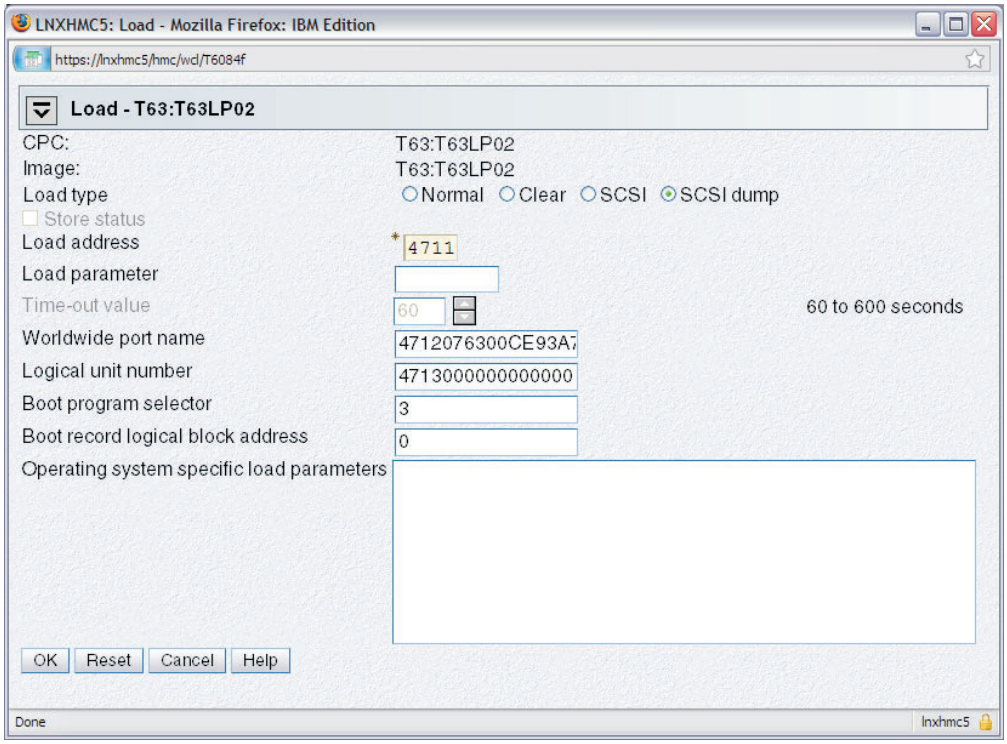


Figure 4. Load panel with enabled SCSI feature for dumping to SCSI disk

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

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

Appendix B. Obtaining a dump with limited size

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

The **size** option is available for all **zipl** based dumps: DASD, tape, and SCSI, 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 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 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 prepares a DASD dump device for a dump that is limited to 100 megabyte:

```
# zipl -d /dev/dasdc1,100M
```

Appendix C. Command summary

This chapter describes tools to work with dumps. The descriptions of the commands contain only the relevant options and parameters, for a full description refer to the man pages.

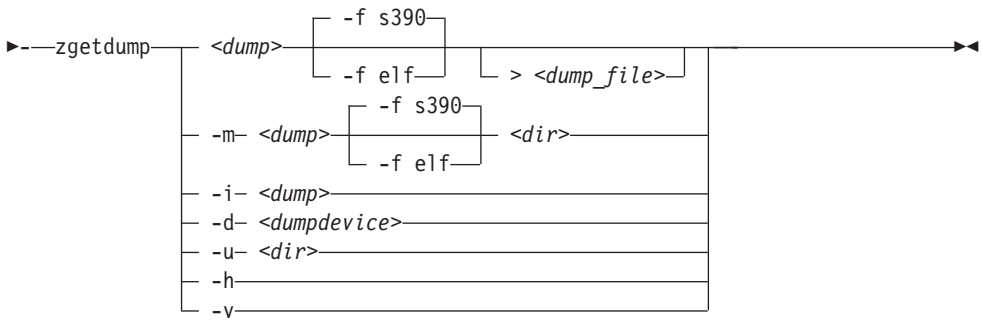
- The zgetdump tool
- The dumpconf tool
- The crash tool
- The vmconvert tool
- The vmur tool

The zgetdump tool

The **zgetdump** tool reads or converts a dump. The dump can be located either on a dump device or on a file system. The dump content is written to standard output, unless you redirect it to a specific file. You can also mount he dump content, print dump information, or check whether a DASD device contains a valid dump tool.

Before you begin: Mounting is implemented with "fuse" (file system in user space). Therefore the fuse kernel module must to be loaded before you can use the **--mount** option.

zgetdump syntax



Where:

<dump>
: is the file, DASD device or partition, or tape device node where the dump is located:

  - Regular dump file (for example `/testdir/dump.0`)
  - DASD partition device node (for example `/dev/dasdc1`)
  - DASD device node for multivolume dump (for example `/dev/dasdc`)
  - Tape device node (for example `/dev/ntibm0`)

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

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

<dumpdevice>
Specifies the dump device for the **-d** option. The device node of the DASD device, for example `/dev/dasdb`.

-m *<dump>* *<dir>* **or --mount** *<dump>* *<dir>*
Mounts the *<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 or FBA device contains a valid dump tool and prints information about it.

-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 (64 bit only)

s390 S/390® dump (default)

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

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

Examples

Using zgetdump to copy a dump

Assuming that the dump is on DASD partition `/dev/dasdb1` and that you want to copy it to a file named `dump_file`:

```
# zgetdump /dev/dasdb1 > dump_file
```

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_file_on_target_host"
```

Using zgetdump to transfer a dump with FTP

Follow these steps to transfer a dump with FTP:

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 host issue a command of the following form:

```
ftp> put |"zgetdump /dev/dasdb1" <dump_file_on_target_host>
```

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 multi_volume_dump_file:

```
# zgetdump /dev/dasdc > multi_volume_dump_file
```

For an example of the output from this command, see Chapter 3, “Using DASD devices for multi-volume dump,” on page 5.

Using zgetdump to copy a tape dump

Assuming that the tape device is /dev/ntibm0:

```
# zgetdump /dev/ntimb0 > dump_file
Format Info:
  Source: s390tape
  Target: s390

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
```

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.......: Mon, 10 May 2010 17:26:46 +0200
  Dump ended.........: Mon, 10 May 2010 17:27:58 +0200
  Dump CPU ID........: ff00012320948000
  Build arch.........: s390x (64 bit)
  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.......: Mon, 10 May 2010 17:32:36 +0200
  Dump ended.........: Mon, 10 May 2010 17:32:48 +0200
  Dump CPU ID........: ff00012320948000
  Build arch.........: s390x (64 bit)
  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 device contains a valid dump record

Checking DASD device `/dev/dasda`, 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"
```

Using the mount option

Mounting is useful for multivolume DASD dumps. After a multivolume 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 multivolume DASD dump as an ELF dump, processes it with **crash**, and unmounts it with **zgetdump**:

```
# zgetdump -m -f elf /dev/dasdx /dumps
# crash vmlinux /dumps/dump.elf
# 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. To do this, mount the dump and specify the required target dump format with the **--fmt** option.

The dumpconf service

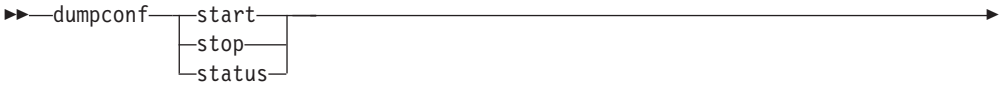
The **dumpconf** service configures the action to be taken if a kernel panic occurs. The service is installed as a script under `/etc/init.d/dumpconf` and reads the configuration file `/etc/sysconfig/dumpconf`.

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

```
# chkconfig --add dumpconf
```

Before you start: You need root permissions.

dumpconf service syntax



Where:

start Enable configuration defined in `/etc/sysconfig/dumpconf`.

stop Disable the **dumpconf** service.

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

-h or --help
Display short usage text on console. To view the man page, enter **man dumpconf**.

-v or --version
Display version number on console, and exit.

Keywords for the configuration file

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

dump Dump Linux and stop system.

reipl Reboot Linux.

dump_reipl
Dump Linux and reboot system. Note that **dump_reipl** is only available on LPAR with z9® machines and later, and on z/VM with version 5.3 and later.

vmcmd
Execute specified CP commands and stop system.

stop Stop Linux (default).

DELAY_MINUTES
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. When the specified time has elapsed, the **dumpconf** service activates the specified panic action. This

action is taken should the system subsequently crash. If the system crashes before the time has elapsed, the previously defined action is taken. If no previous action has been defined, the default action (STOP) is performed.

VMCMD_<*X*>
Specifies a CP command, <*X*> is a number from one to five. You can specify up to five CP commands that are executed in case of a kernel panic. Note that VM commands, device addresses, and VM guest names must be uppercase.

DUMP_TYPE
Type of dump device. Possible values are **ccw** and **fcp**.

DEVICE
Device number of dump device.

WWPN
WWPN for SCSI dump device.

LUN LUN for SCSI dump device.

BOOTPROG
Boot program selector

BR_LBA
Boot record logical block address.

Examples

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 re-IPL if a kernel panic occurs:

```
ON_PANIC=reipl
```

- Example of sending a message to guest "MASTER", executing a **CP VMDUMP** command, and rebooting from device 4711 if a kernel panic occurs:

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

Note that VM commands, device addresses, and VM guest names must be uppercase.

Examples of dumpconf service use

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

- To enable the configuration:

```
# service dumpconf start
ccw dump device configured. "dump" on panic configured.
```

- To display the status:

```
# service dumpconf status
type....: ccw
device..: 0.0.4714
on_panic: dump
```

- To disable dump on panic:

```
# service dumpconf stop
Dump on panic is disabled now
```

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

```
# service dumpconf status
on_panic: stop
```

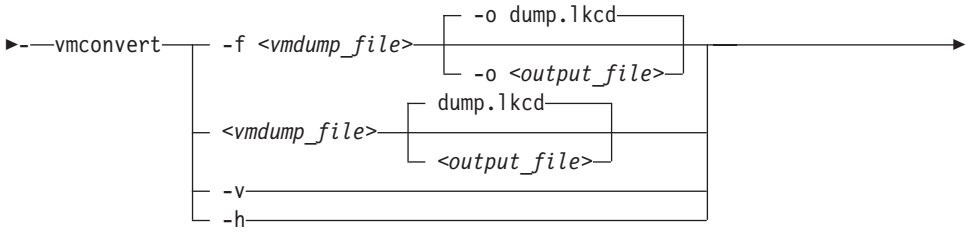
The crash tool

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

The vmconvert tool

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

vmconvert syntax



Where:

<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.
```

The vmur tool

The **vmur** command can receive a **VMDUMP** file from the 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>
```

Where:

<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 Red Hat Enterprise Linux 6.1*, SC34-2597.

Example

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

```
# vmur rec -c 463 dump_file
```

Appendix D. 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. Your distribution typically provides the additional files in RPMs.

If a Red Hat Enterprise Linux 6.1 dump is to be analyzed with **crash**, include:

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

Red Hat Enterprise Linux 6.1 debug files

The Red Hat Enterprise Linux 6.1 debug file is:

Table 3. Red Hat Enterprise Linux 6.1 debug file name

Debug file	Path
vmlinux (full)	`/usr/lib/debug/lib/modules/2.6.32-xx.el6.s390x/vmlinux`

The RPM that contains this file is:

Table 4. Red Hat Enterprise Linux 6.1 debuginfo RPM name

Red Hat Enterprise Linux version	RPM
Red Hat Enterprise Linux 6.1	`kernel-debuginfo-2.6.32-xx.el6.s390x.rpm`

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 System z 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, use the Reader Comment Form in the back of this publication, send an email to eservdoc@de.ibm.com, or write to:

IBM Deutschland Research & Development GmbH
Information Development
Department 3248
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.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Other product and service names might be trademarks of IBM or other companies.

Readers' Comments — We'd Like to Hear from You

Linux on System z
Using the Dump Tools
on Red Hat Enterprise Linux 6.1

Publication No. SC34-2607-01

We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.

For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state on this form.

Comments:

Thank you for your support.

Submit your comments using one of these channels:

- Send your comments to the address on the reverse side of this form.
- Send your comments via email to: eservdoc@de.ibm.com

If you would like a response from IBM, please fill in the following information:

Name	Address
Company or Organization	
Phone No.	Email address

Readers' Comments — We'd Like to Hear from You
SC34-2607-01

Cut or Fold
Along Line

Fold and Tape | **Please do not staple** | Fold and Tape

PLACE
POSTAGE
STAMP
HERE

IBM Deutschland
Research & Development GmbH
Information Development
Department 3248
Schoenaicher Strasse 220
71032 Boeblingen
Germany

Fold and Tape | **Please do not staple** | Fold and Tape

SC34-2607-01

Cut or Fold
Along Line

SC34-2607-01