Red Hat Enterprise Linux 7
Kernel Crash Dump Guide

Kernel Crash Dump Configuration and Analysis

Jaromír Hradílek           Petr Bokoč
Kernel Crash Dump Configuration and Analysis

Jaromír Hradílek
Red Hat Customer Content Services
jhradilek@redhat.com

Petr Bokoč
Red Hat Customer Content Services
pbokoc@redhat.com
Abstract

The Kernel Crash Dump Guide documents how to configure, test, and use the kdump crash recovery service on Red Hat Enterprise Linux 7, and provides a brief overview of how to analyze the resulting core dump using the crash debugging utility. It is oriented towards system administrators with a basic understanding of the Red Hat Enterprise Linux system.
Table of Contents

Chapter 1. Introduction to kdump ........................................... 2
  1.1. About kdump and kexec ............................................ 2
  1.2. Memory Requirements .............................................. 2

Chapter 2. Installing and Configuring kdump ................................ 3
  2.1. Installing kdump ..................................................... 3
  2.2. Configuring kdump on the Command Line ....................... 4
  2.3. Configuring kdump in the Graphical User Interface .......... 8
  2.4. Testing the kdump Configuration ............................... 13
  2.5. Additional Resources ............................................. 13

Chapter 3. Analyzing a Core Dump ........................................... 15
  3.1. Installing the crash Utility ....................................... 15
  3.2. Running the crash Utility ........................................ 15
  3.3. Displaying the Message Buffer .................................. 16
  3.4. Displaying a Backtrace ........................................... 17
  3.5. Displaying a Process Status ..................................... 18
  3.6. Displaying Virtual Memory Information ....................... 18
  3.7. Displaying Open Files ........................................... 19
  3.8. Exiting the Utility ................................................. 19

Appendix A. Frequently Asked Questions .................................. 21

Appendix B. Supported kdump Configurations and Targets ............... 22
  B.1. Memory Requirements for kdump ................................ 22
  B.2. Minimum Threshold for Automatic Memory Reservation .... 22
  B.3. Supported kdump Targets ........................................ 23
  B.4. Supported kdump Filtering Levels ............................. 23
  B.5. Supported Default Actions ...................................... 24

Appendix C. Revision History ............................................... 25
Chapter 1. Introduction to kdump

1.1. About kdump and kexec

Kdump is a kernel crash dumping mechanism that allows you to save the contents of the system's memory for later analysis. It relies on kexec, which can be used to boot a Linux kernel from the context of another kernel, bypass BIOS, and preserve the contents of the first kernel's memory that would otherwise be lost.

In case of a system crash, kdump uses kexec to boot into a second kernel (a capture kernel). This second kernel resides in a reserved part of the system memory that is inaccessible to the first kernel. The second kernel then captures the contents of the crashed kernel's memory (a crash dump) and saves it.

1.2. Memory Requirements

In order for kdump to be able to capture a kernel crash dump and save it for further analysis, a part of the system memory has to be permanently reserved for the capture kernel. When reserved, this part of the system memory is not available to main kernel.

The memory requirements vary based on certain system parameters. One of the major factors is the system's hardware architecture. To find out the exact name of the machine architecture (such as x86_64) and print it to standard output, type the following command at a shell prompt:

```
uname -m
```

Another factor which influences the amount of memory to be reserved is the total amount of installed system memory. For example, on the x86_64 architecture, the amount of reserved memory will be 160 MB + 2 bits for every 4 KB of RAM. On a system with 1 TB of total physical memory installed, this means 224 MB (160 MB + 64 MB). For a complete list of memory requirements for kdump based on the system architecture and the amount of physical memory, see Section B.1, “Memory Requirements for kdump”.

On many systems, kdump can estimate the amount of required memory and reserve it automatically. This behavior is enabled by default, but only works on systems that have more than a certain amount of total available memory, which varies based on the system architecture. See Section B.2, “Minimum Threshold for Automatic Memory Reservation” for a list of minimum requirements for automatic memory reservation based on the system architecture.

If the system has less than the minimum amount of memory required for the automatic allocation to work or if your use case requires a different value, you can configure the amount of reserved memory manually. For information on how to do so on the command line, see Section 2.2.1, “Configuring the Memory Usage”. For information on how to configure the amount of reserved memory in the graphical user interface, see Section 2.3.1, “Configuring the Memory Usage”.

Important

It is highly recommended to test the configuration after setting up the kdump service, even when using the automatic memory reservation. For instructions on how to test your configuration, see Section 2.4, “Testing the kdump Configuration”.

2
Chapter 2. Installing and Configuring kdump

2.1. Installing kdump

In many cases, the kdump service is installed and activated by default on new Red Hat Enterprise Linux 7 installations. The Anaconda installer provides a screen for kdump configuration when performing an interactive installation using the graphical or text interface. The installer screen is titled Kdump and is available from the main Installation Summary screen, and only allows limited configuration - you can only select whether kdump will be enabled and how much memory will be reserved. Information about memory requirements for kdump is available in Section B.1, “Memory Requirements for kdump”. The Kdump configuration screen in the installer is documented in the Red Hat Enterprise Linux 7 Installation Guide.

Note

In previous releases of Red Hat Enterprise Linux, kdump configuration was available in the Firstboot utility which was automatically executed after the installation finished and the system rebooted for the first time. Starting with Red Hat Enterprise Linux 7.1, kdump configuration has been moved into the installer.

Some installation options, such as custom Kickstart installations, may not install or enable kdump by default. If this is the case on your system, and you want to install kdump additionally, execute the following command as root at a shell prompt:

```
# yum install kexec-tools
```

This will install kdump and all other necessary packages, assuming your system has an active subscription or a custom repository containing the kexec-tools package for your system's architecture.

Note

If you do not know whether kdump is installed on your system, you can check using rpm:

```
$ rpm -q kexec-tools
```

Additionally, a graphical configuration tool is available, but not installed by default if you use the command described above. To install this utility, which is described in Section 2.3, “Configuring kdump in the Graphical User Interface”, use the following command as root:

```
# yum install system-config-kdump
```

For more information on how to install new packages in Red Hat Enterprise Linux 7 using the Yum package manager, see the Red Hat Enterprise Linux 7 System Administrator's Guide.
2.2. Configuring kdump on the Command Line

2.2.1. Configuring the Memory Usage

Memory reserved for the kdump kernel is always reserved during system boot, which means that the amount of memory is specified in the system's boot loader configuration. This section will explain how to change the amount of reserved memory on AMD64 and Intel 64 systems and IBM Power Systems servers using the GRUB2 boot loader, and on IBM System z using zipl.

Procedure 2.1. Changing Memory Options in GRUB2

1. Open the /etc/default/grub configuration file as root using a plain text editor such as vim or Gedit.

2. In this file, locate the line beginning with GRUB_CMDLINE_LINUX. The line will look similar to the following:

   GRUB_CMDLINE_LINUX="rd.lvm.lv=rhel/swap crashkernel=auto
d.lvm.lv=rhel/root rhgb quiet"

   Note the highlighted crashkernel= option; this is where the reserved memory is configured.

3. Change the value of the crashkernel= option to the amount of memory you want to reserve. For example, to reserve 128 MB of memory, use the following:

   crashkernel=128M

4. Finally, regenerate the GRUB2 configuration using the edited default file. If your system uses BIOS firmware, execute the following command:

   # grub2-mkconfig -o /boot/grub2/grub.cfg

   On a system with UEFI firmware, execute the following instead:

   # grub2-mkconfig -o /boot/efi/EFI/redhat/grub.cfg

Note

There are multiple ways to configure the memory reserved - for example, you can define an offset or multiple memory amounts based on how much RAM is available in the system at startup. This is described further in this section.
After finishing the procedure above, the boot loader is reconfigured and the amount of memory you have specified in its configuration file will be reserved after the next reboot.

**Procedure 2.2. Changing Memory Options in zipl**

1. Open the `/etc/zipl.conf` configuration file as `root` using a plain text editor such as `vim` or `Gedit`.

2. In this file, locate the `parameters=` section, and edit the `crashkernel=` parameter (or add it if not present). For example, to reserve 128 MB of memory, use the following:

   ```
crashkernel=128M
   ```

   **Note**
   
   There are multiple ways to configure the memory reserved - for example, you can define an offset or multiple memory amounts based on how much RAM is available in the system at startup. This is described further in this section.

   Then, save the file and exit the editor.

3. Finally, regenerate the `zipl` configuration:

   ```
   # zipl
   ```

   **Note**
   
   Executing only the `zipl` command with no additional options will use default values. See the `zipl(8)` man page for information about available options.

After finishing the procedure above, the boot loader is reconfigured and the amount of memory you have specified in its configuration file will be reserved after the next reboot.

The `crashkernel=` option can be defined in multiple ways. The `auto` value enables automatic configuration of reserved memory based on the total amount of memory in the system, following the guidelines described in **Section B.1, “Memory Requirements for kdump”**. Replace the `auto` value with a specific amount of memory to change this behavior. For example, to reserve 128 MB of memory, use the following:

```
crashkernel=128M
```

You can also set the amount of reserved memory to be variable, depending on the total amount of installed memory. The syntax is for variable memory reservation is `crashkernel=<range1>:<size1>,<range2>:<size2>`. For example:

```
crashkernel=512M-2G:64M,2G-:128M
```

The above example will reserve 64 MB of memory if the total amount of system memory is 512 MB or higher and lower than 2 GB. If the total amount of memory is more than 2 GB, 128 MB will be reserved for kdump instead.
On some systems, it might be necessary to reserve memory with a certain fixed offset. If the offset is set, the reserved memory will begin there. To offset the reserved memory, use the following syntax:

\[ \text{crashkernel}=128M@16M \]

The example above means that kdump will reserve 128 MB of memory starting at 16 MB (physical address 0x01000000). If the offset parameter is set to 0 or omitted entirely, kdump will offset the reserved memory automatically. This syntax can also be used when setting a variable memory reservation as described above; in this case, the offset is always specified last (for example, \[ \text{crashkernel}=512M-2G:64M,2G-:128M@16M \]).

### 2.2.2. Configuring the kdump Type

When a kernel crash is captured, the core dump can be either stored as a file in a local file system, written directly to a device, or sent over a network using the NFS (Network File System) or SSH (Secure Shell) protocol. Only one of these options can be set at the moment, and the default option is to store the \text{vmcore} file in the /var/crash/ directory of the local file system. To change this, as root, open the /etc/kdump.conf configuration file in a text editor and edit the options as described below.

To change the local directory in which the core dump is to be saved, remove the hash sign ("#") from the beginning of the \#path /var/crash line, and replace the value with a desired directory path.

\[ \text{path} /usr/local/cores \]

**Important**

In Red Hat Enterprise Linux 7, the directory defined as the kdump target using the \text{path} directive must exist when the \text{kdump} systemd service is started - otherwise the service will fail. This behavior is different from earlier releases of Red Hat Enterprise Linux, where the directory was being created automatically if it did not exist when starting the service.

Optionally, if you wish to write the file to a different partition, follow the same procedure with the one of the lines beginning with \#ext4. Here, you can use either a device name (the \#ext4 /dev/vg/lv_kdump line), a file system label (the \#ext4 LABEL=/boot line) or a UUID (the \#ext4 UUID=03138356-5e61-4ab3-b58e-27507ac41937 line). Change the file system type as well as the device name, label or UUID to the desired values. For example:

\[ \text{ext4 UUID}=03138356-5e61-4ab3-b58e-27507ac41937 \]

**Important**

Specifying storage devices using a \text{LABEL=} or \text{UUID=} is recommended. Disk device names such as /dev/sda3 are not guaranteed to be consistent across reboot. See the Red Hat Enterprise Linux 7 Storage Administration Guide for information about persistent disk device naming.

To write the dump directly to a device, remove the hash sign ("#") from the beginning of the \#raw /dev/vg/lv_kdump line, and replace the value with a desired device name. For example:

\[ \text{raw} /dev/sdb1 \]
To store the dump to a remote machine using the **NFS** protocol, remove the hash sign ("#") from the beginning of the `#nfs my.server.com:/export/tmp` line, and replace the value with a valid hostname and directory path. For example:

```
#nfs penguin.example.com:/export/cores
```

To store the dump to a remote machine using the **SSH** protocol, remove the hash sign ("#") from the beginning of the `#ssh user@my.server.com` line, and replace the value with a valid username and hostname. To include your SSH key in the configuration as well, remove the hash sign ("#") from the beginning of the `#sshkey /root/.ssh/kdump_id_rsa` line and change the value to the location of a key valid on the server you are trying to dump to. For example:

```
ssh john@penguin.example.com
sshkey /root/.ssh/mykey
```

For information on how to configure an SSH server and set up a key-based authentication, see the *Red Hat Enterprise Linux 7 System Administrator's Guide*.

For a complete list of currently supported and unsupported targets sorted by type, see Table B.3, “Supported kdump Targets”.

### 2.2.3. Configuring the Core Collector

To reduce the size of the `vmcore` dump file, **kdump** allows you to specify an external application (a **core collector**) to compress the data, and optionally leave out all irrelevant information. Currently, the only fully supported core collector is **makedumpfile**.

To enable the core collector, as **root**, open the `/etc/kdump.conf` configuration file in a text editor, remove the hash sign ("#") from the beginning of the `#core_collector makedumpfile -l --message-level 1 -d 31` line, and edit the command line options as described below.

To enable the dump file compression, add the `-c` parameter. For example:

```
core_collector makedumpfile -c
```

To remove certain pages from the dump, add the `-d value` parameter, where `value` is a sum of values of pages you want to omit as described in Table B.4, “Supported Filtering Levels”. For example, to remove both zero and free pages, use the following:

```
core_collector makedumpfile -d 17 -c
```

See the `makedumpfile(8)` man page for a complete list of available options.

### 2.2.4. Configuring the Default Action

By default, when **kdump** fails to create a core dump at the target location specified in Section 2.2.2, “Configuring the kdump Type”, the root file system is mounted and **kdump** attempts to save the core locally. To change this behavior, as **root**, open the `/etc/kdump.conf` configuration file in a text editor, remove the hash sign ("#") from the beginning of the `#default shell` line, and replace the value with a desired action as described in Table B.5, “Supported Default Actions”.

For example:
2.2.5. Enabling the Service

To start the kdump daemon at boot time, type the following at a shell prompt as root:

```
systemctl enable kdump.service
```

This will enable the service for multi-user.target. Similarly, typing `systemctl stop kdump` will disable it. To start the service in the current session, use the following command as root:

```
systemctl start kdump.service
```

**Important**

In Red Hat Enterprise Linux 7, the directory defined as the kdump target must exist when the kdump systemd service is started - otherwise the service will fail. This behavior is different from earlier releases of Red Hat Enterprise Linux, where the directory was being created automatically if it did not exist when starting the service.

For more information on systemd and configuring services in general, see the Red Hat Enterprise Linux 7 System Administrator's Guide.

2.3. Configuring kdump in the Graphical User Interface

To start the Kernel Dump Configuration utility, select Activities → Other → Kernel crash dumps from the panel, or type `system-config-kdump` at a shell prompt. You will be presented with a window as shown in Figure 2.1, “Basic Settings”.

The utility allows you to configure kdump as well as to enable or disable starting the service at boot time. When you are done, click Apply to save the changes. Unless you are already authenticated, you will be prompted to enter the superuser password. The utility will also remind you that you must reboot the system in order to apply any changes you have made to the configuration.

2.3.1. Configuring the Memory Usage

The Basic Settings tab enables you to configure the amount of memory that is reserved for the kdump kernel. To do so, select the Manual settings radio button, and click the up and down arrow buttons next to the New kdump Memory field to increase or decrease the amount of memory to be reserved. Notice that the Usable Memory field changes accordingly showing you the remaining memory that will be available to the system. See Section 1.2, “Memory Requirements” for more information on kdump’s memory requirements.
Chapter 2. Installing and Configuring kdump

2.3.2. Configuring the kdump Type

The **Target Settings** tab allows you to specify the target location for the *vmcore* dump. The dump can be either stored as a file in a local file system, written directly to a device, or sent over a network using the **NFS** (Network File System) or **SSH** (Secure Shell) protocol.
To save the dump to the local file system, select the **Local filesystem** radio button. Optionally, you can customize the settings by choosing a different partition from the **Partition** drop-down list and a target directory using the **Path** field.

**Important**

In Red Hat Enterprise Linux 7, the directory defined as the kdump target must exist when the **kdump** systemd service is started - otherwise the service will fail. This behavior is different from earlier releases of Red Hat Enterprise Linux, where the directory was being created automatically if it did not exist when starting the service.

To write the dump directly to a device, select the **Raw device** radio button, and choose the desired target device from the drop-down list next to it.

To send the dump to a remote machine over a network connection, select the **Network** radio button. To use the **NFS** protocol, select the **NFS** radio button, and fill the **Server name** and **Path to directory** fields. To use the **SSH** protocol, select the **SSH** radio button, and fill the **Server name**, **Path to directory**, and **User name** fields with the remote server address, target directory, and a valid user name respectively.
For information on how to configure an SSH server and set up a key-based authentication, see the Red Hat Enterprise Linux 7 System Administrator's Guide. For a complete list of currently supported targets, see Table B.3, “Supported kdump Targets”.

### 2.3.3. Configuring the Core Collector

The **Filtering Settings** tab enables you to select the filtering level for the *vmcore* dump.

![Figure 2.3. Filtering Settings](image)

To exclude the **zero page**, **cache page**, **cache private**, **user data**, or **free page** from the dump, select the checkbox next to the appropriate label.

### 2.3.4. Configuring the Default Action

To choose what action to perform when *kdump* fails to create a core dump, select an appropriate option from the **Default action** drop-down list. Available options are **dump to rootfs and reboot** (the default action which attempts to save the core locally and then reboots the system), **reboot** (to reboot the system), **shell** (to present a user with an interactive shell prompt), **halt** (to halt the system), and **poweroff** (to power the system off).
Figure 2.4. Filtering Settings

To customize the options that are passed to the `makedumpfile` core collector, edit the **Core collector** text field; see Section 2.2.3, “Configuring the Core Collector” for more information.

### 2.3.5. Enabling the Service

To start the `kdump` service at boot time, click the **Enable** button on the toolbar and then click the **Apply** button. This will enable and activate the service for **multi-user.target**. Clicking the **Disable** button and confirming by clicking the **Apply** button will disable the service immediately.

**Important**

In Red Hat Enterprise Linux 7, the directory defined as the kdump target must exist when the `kdump` systemd service is started - otherwise the service will fail. This behavior is different from earlier releases of Red Hat Enterprise Linux, where the directory was being created automatically if it did not exist when starting the service.

For more information on systemd targets and configuring services in general, see the **Red Hat Enterprise Linux 7 System Administrator’s Guide**.
2.4. Testing the kdump Configuration

Warning

The commands below will cause the kernel to crash. Use caution when following these steps, and by no means use them on a production system.

To test the configuration, reboot the system with **kdump** enabled, and make sure that the service is running:

~
```
$ systemctl is-active kdump
active
```

Then type the following commands at a shell prompt:

```
$ echo 1 > /proc/sys/kernel/sysrq
$ echo c > /proc/sysrq-trigger
```

This will force the Linux kernel to crash, and the **address-YYYY-MM-DD-HH:MM:SS/vmcore** file will be copied to the location you have selected in the configuration (that is, to **/var/crash/** by default).

2.5. Additional Resources

2.5.1. Installed Documentation

- **kdump.conf**(5) — a manual page for the **/etc/kdump.conf** configuration file containing the full documentation of available options.
- **zipl.conf**(5) — a manual page for the **/etc/zipl.conf** configuration file.
- **zipl**(8) — a manual page for the **zipl** boot loader utility for IBM System z.
- **makedumpfile**(8) — a manual page for the **makedumpfile** core collector.
- **kexec**(8) — a manual page for **kexec**.
- **crash**(8) — a manual page for the **crash** utility.
- **/usr/share/doc/kexec-tools-version/kexec-kdump-howto.txt** — an overview of the **kdump** and **kexec** installation and usage.

2.5.2. Online Documentation

- [https://access.redhat.com/site/solutions/6038](https://access.redhat.com/site/solutions/6038)
  The Red Hat Knowledgebase article about the **kexec** and **kdump** configuration.
- [https://access.redhat.com/site/solutions/223773](https://access.redhat.com/site/solutions/223773)
  The Red Hat Knowledgebase article about supported **kdump** targets.
- [http://people.redhat.com/anderson/](http://people.redhat.com/anderson/)
The crash utility homepage.

https://www.gnu.org/software/grub/

The GRUB2 boot loader homepage and documentation.
Chapter 3. Analyzing a Core Dump

To determine the cause of the system crash, you can use the `crash` utility, which provides an interactive prompt very similar to the GNU Debugger (GDB). This utility allows you to interactively analyze a running Linux system as well as a core dump created by `netdump`, `diskdump`, `xendump`, or `kdump`.

3.1. Installing the crash Utility

To install the `crash` analyzing tool, execute the following command from a shell prompt as `root`:

```
yum install crash
```

In addition to `crash`, it is also necessary to install the `kernel-debuginfo` package, which provides the data necessary for dump analysis. To install this package, you will first need to enable the relevant repository. To do this, execute the following command from the command line as `root`:

```
yum --enablerepo=*debuginfo
```

After enabling the repository, install the `kernel-debuginfo` package by executing the following command as `root`:

```
deginfo-install kernel
```

For more information on how to install new packages in Red Hat Enterprise Linux using the `Yum` package manager, see the *Red Hat Enterprise Linux 7 System Administrator's Guide*.

3.2. Running the crash Utility

To start the utility, type the command in the following form at a shell prompt:

```
crash /var/crash/<timestamp>/vmcore
      /usr/lib/debug/lib/modules/<kernel>/vmlinux
```

Note that the `<kernel>` version should be the same that was captured by `kdump`. To find out which kernel you are currently running, use the `uname -r` command.

Example 3.1. Running the crash utility

```
~]$ crash /usr/lib/debug/lib/modules/2.6.32-69.el6.i686/vmlinux  
   /var/crash/127.0.0.1-2010-08-25-08:45:02/vmcore
```

crash 5.0.0-23.el6
Copyright (C) 2002-2010 Red Hat, Inc.
Copyright (C) 2004, 2005, 2006 IBM Corporation
Copyright (C) 1999-2006 Hewlett-Packard Co
Copyright (C) 2005, 2006 Fujitsu Limited
Copyright (C) 2006, 2007 VA Linux Systems Japan K.K.
Copyright (C) 2005 NEC Corporation
This program is free software, covered by the GNU General Public License, and you are welcome to change it and/or distribute copies of it under certain conditions. Enter "help copying" to see the conditions. This program has absolutely no warranty. Enter "help warranty" for details.

GNU gdb (GDB) 7.0
Copyright (C) 2009 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later
<http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it. There is NO WARRANTY, to the extent permitted by law. Type "show copying" and "show warranty" for details.
This GDB was configured as "i686-pc-linux-gnu"...

KERNEL: /usr/lib/debug/lib/modules/2.6.32-69.el6.i686/vmlinux
DUMPFILE: /var/crash/127.0.0.1-2010-08-25-08:45:02/vmcore [PARTIAL DUMP]
CPUS: 4
UPTIME: 00:09:02
LOAD AVERAGE: 0.00, 0.01, 0.00
TASKS: 140
NODENAME: hp-dl320g5-02.lab.bos.redhat.com
RELEASE: 2.6.32-69.el6.i686
VERSION: #1 SMP Tue Aug 24 10:31:45 EDT 2010
MACHINE: i686 (2394 Mhz)
MEMORY: 8 GB
PANIC: "Oops: 0002 [#1] SMP " (check log for details)
PID: 5591
COMMAND: "bash"
TASK: f196d560 [THREAD_INFO: ef4da000]
CPU: 2
STATE: TASK_RUNNING (PANIC)

3.3. Displaying the Message Buffer

To display the kernel message buffer, type the log command at the interactive prompt.

Example 3.2. Displaying the kernel message buffer

```
crash> log
... several lines omitted ...
EIP: 0060:[<c068124f>] EFLAGS: 00010096 CPU: 2
EIP is at sysrq_handle_crash+0xf/0x20
EAX: 00000063 EBX: 00000063 ECX: c09e1c8c EDX: 00000000
ESI: c0a09ca0 EDI: 00000286 EBP: 00000000 ESP: ef4dbf24
DS: 007b ES: 007b FS: 00d8 GS: 00e0 SS: 0068
Process bash (pid: 5591, ti=ef4da000 task=f196d560 task.ti=ef4da000)
```
Type **help log** for more information on the command usage.

---

### Note

The kernel message buffer includes the most essential information about the system crash and, as such, it is always dumped first into the **vmcore-dmesg.txt** file. This is useful when an attempt to get the full **vmcore** file failed, for example because of lack of space on the target location. By default, **vmcore-dmesg.txt** is located in the **/var/crash/** directory.

---

### 3.4. Displaying a Backtrace

To display the kernel stack trace, type the **bt** command at the interactive prompt. You can use **bt <pid>** to display the backtrace of a single process.

---

#### Example 3.3. Displaying the kernel stack trace

```
crash> bt
PID: 5591   TASK: f196d560   CPU: 2   COMMAND: "bash"
#0 [ef4dbdcc] crash_kexec at c0494922
#1 [ef4dbec0] oops_end at c080e402
#2 [ef4dfe34] no_context at c043089d
#3 [ef4dfe58] bad_area at c0430b26
#4 [ef4dfe6c] do_page_fault at c080fb9b
#5 [ef4dfe44] error_code (via page_fault) at c080d809
   EAX: 00000063  EBX: 00000063  ECX: c09e1c8c  EDX: 00000000  EBP:
   00000000
   DS: 007b   ESI: c0a09ca0  ES: 007b  EDI: 00000286  GS: 00e0
   CS: 0060   EIP: c068124f  ERR: ffffffff  EFLAGS: 00010096
```
3.5. Displaying a Process Status

To display status of processes in the system, type the `ps` command at the interactive prompt. You can use `ps <pid>` to display the status of a single process.

Example 3.4. Displaying the status of processes in the system

```
crash> ps
     PID    PPID  CPU   TASK    ST  %MEM     VSZ    RSS  COMM
>     0      0   0  c09dc560  RU   0.0       0      0  [swapper]
>     0      0   1  f7072030  RU   0.0       0      0  [swapper]
>     0      0   2  f70a3a90  RU   0.0       0      0  [swapper]
>     0      0   3  f70ac560  RU   0.0       0      0  [swapper]
  1      0   1  f705ba90  IN   0.0    2828   1424  init
... several lines omitted ...
  5566      1   1  f2592560  IN   0.0   12876    784  auditd
  5567      1   2  ef427560  IN   0.0   12876    784  auditd
  5587   5132   0  f196d030  IN   0.0   11064   3184  sshd
>  5591  5587   2  f196d560  RU   0.0    5084   1648  bash
```

Type `help ps` for more information on the command usage.

3.6. Displaying Virtual Memory Information

To display basic virtual memory information, type the `vm` command at the interactive prompt. You can use `vm <pid>` to display information on a single process.

Example 3.5. Displaying virtual memory information of the current context

```
crash> vm
   PID: 5591   TASK: f196d560   CPU: 2   COMMAND: "bash"
MM       PGD      RSS    TOTAL_VM
f19b5900  ef9c6000  1648k    5084k
VMA       START      END    FLAGS  FILE
f1bb0310    242000    260000 8000875  /lib/ld-2.12.so
f26af0b8    260000   261000 8100871  /lib/ld-2.12.so
```
3.7. Displaying Open Files

To display information about open files, type the `files` command at the interactive prompt. You can use `files <pid>` to display files opened by only one selected process.

Example 3.6. Displaying information about open files of the current context

```
    crash> files
    PID: 5591   TASK: f196d5d0  CPU: 2   COMMAND: "bash"
    ROOT: /    CWD: /root
    FD    FILE     DENTRY    INODE    TYPE  PATH
    0  f734f640  eedc2c6c  eecd6048  CHR   /pts/0
    1  efae5c0  eee14090  f00431d4  REG   /proc/sysrq-trigger
    2  f734f640  eedc2c6c  eecd6048  CHR   /pts/0
    10 f734f640  eedc2c6c  eecd6048  CHR   /pts/0
    255 f734f640  eedc2c6c  eecd6048  CHR   /pts/0
```

Type `help files` for more information on the command usage.

3.8. Exiting the Utility

To exit the interactive prompt and terminate `crash`, type `exit` or `q`.

Example 3.7. Exiting the crash utility

```
    crash> exit
    ~]#
```
Appendix A. Frequently Asked Questions

Q: How do I upload a large dump file to Red Hat Support Services?

A: In some cases, it might be necessary to send a kernel crash dump file to Red Hat Global Support Services for analysis. However, the dump file can be very large, even after being filtered. Since files larger than 250 MB cannot be uploaded directly through the Red Hat Customer Portal when opening a new support case, an FTP server is provided by Red Hat for uploading large files.

The FTP server's address is dropbox.redhat.com and the files are to be uploaded in the /incoming/ directory. Your FTP client needs to be set into passive mode; if your firewall does not allow this mode, you may use the origin-dropbox.redhat.com server using active mode.

Make sure that the uploaded files are compressed using a program such as gzip and properly and descriptively named. Using your support case number in the file name is recommended. After successfully uploading all necessary files, provide the engineer in charge of your support case with the exact file name and its SHA1 or MD5 checksum.

For more specific instructions and additional information, see https://access.redhat.com/site/solutions/2112.
Appendix B. Supported kdump Configurations and Targets

B.1. Memory Requirements for kdump

In order for kdump to be able to capture a kernel crash dump and save it for further analysis, a part of the system memory has to be permanently reserved for the capture kernel. The table below contains a list of minimum memory requirements for kdump based on the system's architecture and total available physical memory.

For information on how to change memory settings on the command line, see Section 2.2.1, “Configuring the Memory Usage”. For instructions on how to set up the amount of reserved memory in the graphical user interface, see Section 2.3.1, “Configuring the Memory Usage”.

Table B.1. Minimum Amount of Reserved Memory Required for kdump

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Available Memory</th>
<th>Minimum Reserved Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMD64 and Intel 64 (x86_64)</td>
<td>2 GB and more</td>
<td>160 MB + 2 bits for every 4 KB of RAM. For a system with 1 TB of memory, 224 MB is the minimum (160 + 64 MB).</td>
</tr>
<tr>
<td>IBM POWER (ppc64)</td>
<td>2 GB to 4 GB</td>
<td>256 MB of RAM.</td>
</tr>
<tr>
<td></td>
<td>4 GB to 32 GB</td>
<td>512 MB of RAM.</td>
</tr>
<tr>
<td></td>
<td>32 GB to 64 GB</td>
<td>1 GB of RAM.</td>
</tr>
<tr>
<td></td>
<td>64 GB to 128 GB</td>
<td>2 GB or RAM.</td>
</tr>
<tr>
<td></td>
<td>128 GB and more</td>
<td>4 GB of RAM.</td>
</tr>
<tr>
<td>IBM System z (s390x)</td>
<td>2 GB and more</td>
<td>160 MB + 2 bits for every 4 KB of RAM. For a system with 1 TB of memory, 224 MB is the minimum (160 + 64 MB).</td>
</tr>
</tbody>
</table>

B.2. Minimum Threshold for Automatic Memory Reservation

On some systems, it is possible to allocate memory for kdump automatically, either by using the crashkernel=auto parameter in the bootloader's configuration file, or by enabling this option in the graphical configuration utility. For this automatic reservation to work, however, a certain amount of total memory needs to be available in the system. This amount differs based on the system's architecture.

The table below lists the thresholds for automatic memory allocation. If the system has less memory than specified in the table, memory will have to be reserved manually.

For information on how to change these settings on the command line, see Section 2.2.1, “Configuring the Memory Usage”. For instructions on how to change the amount of reserved memory in the graphical user interface, see Section 2.3.1, “Configuring the Memory Usage”.

Table B.2. Minimum Amount of Memory Required for Automatic Memory Reservation

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Required Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMD64 and Intel 64 (x86_64)</td>
<td>2 GB</td>
</tr>
<tr>
<td>IBM POWER (ppc64)</td>
<td>2 GB</td>
</tr>
<tr>
<td>IBM System z (s390x)</td>
<td>4 GB</td>
</tr>
</tbody>
</table>
B.3. Supported kdump Targets

When a kernel crash is captured, the core dump can be either written directly to a device, stored as a file on a local file system, or sent over a network. The table below contains a complete list of dump targets that are currently supported or explicitly unsupported by kdump.

For information on how to configure the target type on the command line, see Section 2.2.2, “Configuring the kdump Type”. For information on how to do so in the graphical user interface, see Section 2.3.2, “Configuring the kdump Type”.

<table>
<thead>
<tr>
<th>Type</th>
<th>Supported Targets</th>
<th>Unsupported Targets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw device</td>
<td>All locally attached raw disks and partitions.</td>
<td>—</td>
</tr>
<tr>
<td>Local file system</td>
<td>ext2, ext3, ext4, btrfs and xfs file systems on directly attached disk drives,</td>
<td>Any local file system not explicitly listed as supported in this table, including</td>
</tr>
<tr>
<td></td>
<td>hardware RAID logical drives, LVM devices, and mdraid arrays.</td>
<td>the auto type (automatic file system detection).</td>
</tr>
<tr>
<td>Remote directory</td>
<td>Remote directories accessed using the NFS or SSH protocol over IPv4.</td>
<td>Remote directories on the rooteFs file system accessed using the NFS protocol.</td>
</tr>
<tr>
<td></td>
<td>Remote directories accessed using the iSCSI protocol over software initiators,</td>
<td>Remote directories accessed using the iSCSI protocol using iBFT.</td>
</tr>
<tr>
<td></td>
<td>unless iBFT (iSCSI Boot Firmware Table) is utilized.</td>
<td>Remote directories accessed using the iSCSI protocol over hardware initiators.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Remote directories accessed using the SMB/CIFS protocol.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Remote directories accessed using the FCoE (Fibre Channel over Ethernet) protocol.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Remote directories accessed using wireless network interfaces.</td>
</tr>
</tbody>
</table>

B.4. Supported kdump Filtering Levels

To reduce the size of the dump file, kdump uses the makedumpfile core collector to compress the data and optionally leave out irrelevant information. The table below contains a complete list of filtering levels that are currently supported by the makedumpfile utility.

For instructions on how to configure the core collector on the command line, see Section 2.2.3, “Configuring the Core Collector”. For information on how to do so in the graphical user interface, see Section 2.3.3, “Configuring the Core Collector”.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Zero pages</td>
</tr>
</tbody>
</table>
By default, when kdump fails to create a core dump, it mounts the root file system and attempts to save the core locally. You can, however, configure kdump to perform a different operation in case it fails to save the core dump to the primary target. The table below lists all default actions that are currently supported by kdump.

For detailed information on how to set up the default action on the command line, see Section 2.2.4, “Configuring the Default Action”. For information on how to do so in the graphical user interface, see Section 2.3.4, “Configuring the Default Action”.

Table B.5. Supported Default Actions

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dump_to_rootfs</td>
<td>Attempt to save the core dump to the root file system. This option is especially useful in combination with a network target: if the network target is unreachable, this option configures kdump to save the core dump locally. The system is rebooted afterwards.</td>
</tr>
<tr>
<td>reboot</td>
<td>Reboot the system, losing the core dump in the process.</td>
</tr>
<tr>
<td>halt</td>
<td>Halt the system, losing the core dump in the process.</td>
</tr>
<tr>
<td>poweroff</td>
<td>Power off the system, losing the core dump in the process.</td>
</tr>
<tr>
<td>shell</td>
<td>Run a shell session from within the initramfs, allowing the user to record the core dump manually.</td>
</tr>
</tbody>
</table>
### Appendix C. Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Author</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.3-0</td>
<td>Thu 12th Nov 2015</td>
<td>Mark Flitter</td>
<td>Updating for RHEL 7.2.</td>
</tr>
<tr>
<td>1.2-0</td>
<td>Fri 06 Mar 2015</td>
<td>Petr Bokoč</td>
<td>Update fixing several issues such as wrong information for memory configuration and outdated screenshots.</td>
</tr>
<tr>
<td>1.1-3</td>
<td>Wed 18 Feb 2015</td>
<td>Petr Bokoč</td>
<td>Red Hat Enterprise Linux 7.1 GA release of the Kernel Crash Dump Guide.</td>
</tr>
<tr>
<td>1.1-0</td>
<td>Fri 05 Dec 2014</td>
<td>Petr Bokoč</td>
<td>Red Hat Enterprise Linux 7.1 Beta release of the Kernel Crash Dump Guide.</td>
</tr>
<tr>
<td>1.0-0</td>
<td>Mon 02 Jun 2014</td>
<td>Jaromír Hradílek</td>
<td>Red Hat Enterprise Linux 7.0 GA release of the Kernel Crash Dump Guide.</td>
</tr>
<tr>
<td>0.0-8</td>
<td>Thu Jan 17 2013</td>
<td>Jaromír Hradílek</td>
<td>Initial creation of the book.</td>
</tr>
</tbody>
</table>