Blueprints
Configuring Remote Crash Dump on Linux Systems
Blueprints
Configuring Remote Crash Dump on Linux Systems
Before using this information and the product it supports, read the information in "Notices" on page 39.
Chapter 1. Scope, requirements, and support

This blueprint applies to System x® running Linux (dump server and client) and PowerLinux™ (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Systems to which this information applies
System x running Linux (dump server and client) and PowerLinux (dump server only)

Intended audience
This document is written for system administrators who install and manage IBM® System x servers running Red Hat Enterprise Linux or SUSE Linux Enterprise Server (SLES).

Scope and purpose
The blueprint covers the configuration of Kdump clients set to dump remotely to a Kdump network server. Because you can easily configure your system to dump locally using the distribution’s systems management tools, this configuration is not covered. In addition, the distributions support dumping over other network protocols besides SSH, but this configuration is also not covered.

For more information about these two configurations, see the documentation supplied with your distribution.

Hardware and software requirements
The instructions in this blueprint are written for Kdump servers and clients running the Red Hat Enterprise Linux (RHEL) 5.3 or SLES 10 SP2 operating systems. The Kdump server should have enough storage to receive the crash dumps from the clients.

Kdump clients are tested on IBM System x servers; Kdump servers are tested on IBM System x and System p® servers. The Kdump utility is not supported if the Kdump client's operating system distribution does not match the Kdump client machine's.

Author names
Max Asbock

Other contributors
Monza Lui
Kersten Richter
Robb Romans

IBM Services
Linux offers flexibility, options, and competitive total cost of ownership with a world class enterprise operating system. Community innovation integrates leading-edge technologies and best practices into Linux.
IBM is a leader in the Linux community with over 600 developers in the IBM Linux Technology Center working on over 100 open source projects in the community. IBM supports Linux on all IBM servers, storage, and middleware, offering the broadest flexibility to match your business needs.

For more information about IBM and Linux, go to [ibm.com/linux](https://www.ibm.com/linux)

**IBM Support**

Questions and comments regarding this documentation can be posted on the developerWorks® Systems Management Blueprint Community Forum:


The IBM developerWorks discussion forums let you ask questions, share knowledge, ideas, and opinions about technologies and programming techniques with other developerWorks users. Use the forum content at your own risk. While IBM will attempt to provide a timely response to all postings, the use of this developerWorks forum does not guarantee a response to every question that is posted, nor do we validate the answers or the code that are offered.

**Typographic conventions**

The following typographic conventions are used in this Blueprint:

<table>
<thead>
<tr>
<th><strong>Bold</strong></th>
<th>Identifies commands, subroutines, keywords, files, structures, directories, and other items whose names are predefined by the system. Also identifies graphical objects such as buttons, labels, and icons that the user selects.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Italics</strong></td>
<td>Identifies parameters whose actual names or values are to be supplied by the user.</td>
</tr>
<tr>
<td><strong>Monospace</strong></td>
<td>Identifies examples of specific data values, examples of text like what you might see displayed, examples of portions of program code like what you might write as a programmer, messages from the system, or information you should actually type.</td>
</tr>
</tbody>
</table>
Chapter 2. Kdump overview

A kernel crash dump is the memory image of an operating system (OS) kernel that is written to a file. Typically, the system writes a crash dump file when the OS experiences a serious problem such as a hang or crash. You can open the saved crash dump file with the Crash analysis utility to determine the root cause of OS problems.

Red Hat Enterprise Linux and SUSE Linux Enterprise Server provide Kdump software to capture and save crash dumps. The Kdump-enabled OS kernel loads an additional standby Kdump kernel into reserved memory. When a hang or crash triggers a crash dump, the OS kernel boots into the Kdump kernel without a system reset. The standby Kdump kernel then writes an image of the saved memory into a crash dump file. The Kdump utility can save crash dump files on a local disk or through the network onto a remote system configured to store the dump. This blueprint only covers the remote dumping setup.

Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.
Chapter 3. Setting up a net dump server

You can configure Kdump to dump to a remote server over a network connection. To do this, configure two systems: one that will receive the dump (the server) and another that creates a dump (the client). This section shows you how to configure a server that receives crash dumps from remote systems through Secure Shell (SSH). The instructions are the same for Red Hat Enterprise Linux 5.3 and SLES 10 SP2.

About this task

There are two considerations when choosing the net dump server:

- The server should be accessible for the Kdump client directly through SSH without additional authentication.
- Also, the server must have enough disk space to handle the dumps.

Because the dump files from remote systems are exact replicas of their system memory, the size of each dump file will be equivalent to the memory size.

Procedure

1. Create a net dump user account and a directory to hold the crash dumps. In this example, netdumpuser is used as the user and group name, and /var/netdumpuser is used as the directory to hold the dumps. You can choose another user name and directory.
   a. Create a new group named netdumpuser:
      
      ```bash
      # groupadd -r netdumpuser
      ```
   b. Create the new user (system account) netdumpuser with the following attributes:
      - group netdumpuser
      - home directory /var/netdumpuser, which contains the crash dump files
      - shell /bin/bash
      - user account comment Network crash dump user

      To create the new user with these attributes, enter the following command:
      
      ```bash
      # useradd -c "Network Crash dump user" -r -g netdumpuser -s /bin/bash \\
      -d /var/netdumpuser netdumpuser
      ```
   c. Create all components of the /var/netdumpuser/.ssh directory, including the /var/netdumpuser directory, with the following attributes:
      - user and group ownership set to netdumpuser
      - permission mode set to 700

      To create these components, enter the following command:
      
      ```bash
      # install -o netdumpuser -g netdumpuser -m 700 -d /var/netdumpuser/.ssh
      ```
   d. Verify that the previous commands executed successfully by typing:
      
      ```bash
      # ls -al /var/netdumpuser/|grep .ssh
      ```
      You will see output similar to the following:
      
      ```bash
      drwx------ 2 netdumpuser netdumpuser 4096 Jul  1 22:10 .ssh
      ```
   e. Change the password of the netdumpuser user, as follows:
      
      ```bash
      # passwd netdumpuser
      Changing password for user netdumpuser.
      ```
      You should see the following output:
      
      ```bash
      passwd: all authentication tokens updated successfully.
      ```
Note: This password will be needed later on to authenticate for dumping to this machine.

f. If you want to configure the server to receive network dumps from all machines in your network, you can create multiple categorical subdirectories within the /var/netdumpuser directory. This helps you to easily manage dumps from different machines. To do so, enter the following commands:

```bash
# mkdir /var/netdumpuser/client1
# mkdir /var/netdumpuser/client2
# mkdir /var/netdumpuser/client3
```

g. Type the following commands to set the required permissions, ownership, and group ownership on the /var/netdumpuser directory:

```bash
# chown -R netdumpuser:netdumpuser /var/netdumpuser
# chmod -R 700 /var/netdumpuser
```

h. Verify that the commands in Step g ran successfully. You should see output from the command similar to the following output that shows that all files are owned by the netdumpuser user and group:

```bash
# ls -al /var/netdumpuser/
```

2. Try connecting to the server from the client with ssh. If the connection is successful, you are done with this section and can continue with Chapter 4, “Setting up Kdump on the dump client,” on page 9. If the connection ssh is not successful, continue to Step 3.

3. Ensure that the sshd daemon is running to receive the dumps. To verify, type the following command:

```bash
# service sshd status
```

If sshd is running, you should see output similar to the following:

```
Checking for service sshd running
```

Or you might see output similar to the following:

```
openssh-daemon (pid 4290) is running...
```

If you do not see the previous output, type the following commands:

```bash
# chkconfig sshd on
# service sshd start
```

4. If a firewall is enabled on the net dump server and the ssh service is not allowed, change the security settings to allow ssh. Complete the steps appropriate for the distribution you are using:

- Allow SSH through the firewall on Red Hat Enterprise Linux 5.3 with the following steps:
  a. In a X Window System environment, start the graphical security configuration interface by typing:

```bash
# system-config-securitylevel
```

  **Note:** If you are not running X Window System, you can run system-config-securitylevel-tui to use the text-based user interface.

  b. If the firewall is disabled, you do not have to make any additional changes.

  c. If the firewall is enabled, in the **Trusted services** panel, click **SSH** if it is not selected, and complete Step d through Step f.

  d. Click **OK** to accept the change.

  e. Click **Yes** to confirm overriding of previous configuration.

  f. Test the configuration by trying an ssh connection to the machine.

- Allow SSH through the firewall on SLES 10 SP2 with the following steps:
  a. Start the firewall configuration graphical user interface by entering the following command:
# yast2 firewall

b. Optional: If the firewall is disabled, you do not have to make any additional changes. Go to Chapter 4, “Setting up Kdump on the dump client,” on page 9.

c. Optional: If the firewall is enabled, click **Allowed Services** in the left panel and complete Step d through Step h.

d. If **Secure Shell Server** or **SSH** is not listed in the **Allowed Services** list, select it from the **Services to Allow** menu and click **Add**.

e. Go back to the previous screen by clicking **Start-Up** in the left panel.

f. Click **Save Settings** and **Restart Firewall Now**.

g. Click **Next**, and then click **Finish**.

h. Test the configuration by trying an ssh connection to the machine.

**What to do next**

When you are finished, go to Chapter 4, “Setting up Kdump on the dump client,” on page 9.

**Related reference:**

Chapter 1, “Scope, requirements, and support,” on page 1

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.
Chapter 4. Setting up Kdump on the dump client

You can configure Kdump on RHEL 5.3 or SLES 10 SP2. The configuration is supported only if the client and the server are running on the same operating system distribution. Skip to the section which corresponds to the distribution running your dump client.

Related reference:
Chapter 1, “Scope, requirements, and support,” on page 1

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Configuring Kdump on a Red Hat Enterprise Linux 5.3 system

During installation, you are prompted to configure Kdump. However the configuration options are limited to enabling it and setting the Kdump memory size. Enabling Kdump at that time guarantees a local Kdump to the /var/crash directory.

About this task

For a remote setup, configure Kdump on an installed system. The following steps configure the client machine Kdump to the remote net dump server that you set up in the previous section.

Procedure

1. Start the Kdump graphical interface by typing the following command in an X Window System environment (using ssh -X for example):

   # system-config-kdump

2. Check Enable kdump, as follows:
3. In the Path box, enter the path to write the crash dumps to the net dump server. This example uses var/netdumpuser/client1
4. Click **Edit Location**. In the window, select **ssh** from the **Select a location type** menu. Enter `<username>@<hostname of the dump server>` in the **Enter Location** field. This example uses `netdumpuser@netdump_server`. Click **OK**.

5. On the **system-config-kdump** window, verify that the values in the **Location** field and the **Path** field are correct and click **OK**.

The output on your system is similar to the following example:

```
[root@x206f ~]# system-config-kdump
Stopping kdump: [ OK ]
Detected change(s) the following file(s):
/etc/kdump.conf
Rebuilding /boot/initrd-2.6.18-128.e15kdump.img
```
6. Enter the following command to allow authentication with the net dump server and set up a public transfer:
   
   # service kdump propagate

7. Either reboot the system or restart the kdump service, using one of the following commands:
   
   - If you are enabling Kdump for the first time, reboot the system so that memory is reserved for the Kdump kernel. Enter the following command:
   
     # reboot

   - If Kdump is already enabled, restart the kdump service by entering the following command:

     # service kdump restart

8. Verify that Kdump daemon is running by entering the following command:

   # service kdump status

   You should see the following output:

   Kdump is operational

9. Verify that the public key authentication works by entering the following command:

   # ssh netdumpuser@netdump_server

   You should now be logged into netdump_server as netdumpuser without a password.

Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

### Configuring Kdump on SLES 10 SP2

During installation, you are prompted to configure Kdump. However, the configuration options are limited to enabling Kdump and setting the Kdump memory size. If you enabled Kdump at the time of installation, a local Kdump saves to the /var/crash directory. Follow this procedure to change the configuration of Kdump on a SLES 10 SP2 client to save dumps to a remote server through the SSH protocol.

**Before you begin**

Kdump must be enabled on an installed system.

**Procedure**

1. Configure Kdump using the yast2 configuration tool.
   
   a. In an X Window System environment, start the Kdump configuration tool by entering the following command:

      # yast2 kdump

   b. Select Enable Kdump if it is not already selected.

   c. In the left panel, click Dump Target.

   d. Select SSH from the Saving Target for Kdump Image list.

   e. Specify the location and authentication values for receiving and storing the crash dumps:

      - In the Server Name field, enter the host name of the remote server receiving the crash dumps.

      - In the Directory on Server field, enter the directory where the receiving server stores the crash dumps.
• In the **User Name** field, enter the user name to be used for SSH authentication on the receiving server.

**Note:** The directory and user name specified here must match the settings specified in Chapter 3, “Setting up a net dump server,” on page 5.

2. Transfer an SSH public key to the net dump server. Transferring the SSH public key to the net dump server grants you access to the net dump server without a password during future logins.

   a. Generate an SSH private/public key pair by entering the following command:

   ```
   # ssh-keygen -N '' -C 'passthrough key' -t dsa
   ```

   You should see the following output:

   ```
   Generating public/private dsa key pair.
   Enter file in which to save the key (/root/.ssh/id_dsa):
   Your identification has been saved in /root/.ssh/id_dsa.
   Your public key has been saved in /root/.ssh/id_dsa.pub.
   The key fingerprint is:
   ```

   b. Add the public SSH key to the `/home/netdumpuser/ssh/authorized_keys` file on the remote dump server. Doing so allows password-less login to the dump server as netdumpuser. Type the following commands:

   ```
   # ssh netdumpuser@netdump_server 'cat >> ~/.ssh/authorized_keys' < /root/.ssh/id_dsa.pub
   ```

---

**Figure 3. Dump Target**

2. Transfer an SSH public key to the net dump server. Transferring the SSH public key to the net dump server grants you access to the net dump server without a password during future logins.

   a. Generate an SSH private/public key pair by entering the following command:

   ```
   # ssh-keygen -N '' -C 'passthrough key' -t dsa
   ```

   You should see the following output:

   ```
   Generating public/private dsa key pair.
   Enter file in which to save the key (/root/.ssh/id_dsa):
   Your identification has been saved in /root/.ssh/id_dsa.
   Your public key has been saved in /root/.ssh/id_dsa.pub.
   The key fingerprint is:
   ```

   b. Add the public SSH key to the `/home/netdumpuser/ssh/authorized_keys` file on the remote dump server. Doing so allows password-less login to the dump server as netdumpuser. Type the following commands:

   ```
   # ssh netdumpuser@netdump_server 'cat >> ~/.ssh/authorized_keys' < /root/.ssh/id_dsa.pub
   ```
c. When you are prompted for a password, enter the password for netdumpuser.

d. Verify that the public key authentication works by entering the following command:

```
# ssh netdumpuser@netdump_server
```

You should now be able to log in to **netdump_server** as netdumpuser without entering a password.

**Related reference:**

[Chapter 1, “Scope, requirements, and support,” on page 1](#)

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.
Chapter 5. Testing Kdump

Follow this procedure to test if the Kdump setup is working.

About this task

Initiating a Kdump causes the client system to reboot. Before starting this procedure, ensure that the client system is ready to be rebooted. Also ensure that the remote server has enough disk space to save the dump file.

On the client system:

Procedure

1. Write any outstanding data to disk by entering the following command:
   
   # sync

2. Trigger the crash dump by entering the following command:
   
   # echo c > /proc/sysrq-trigger

   The client system becomes unresponsive.

3. A crash dump file is created in the /var/netdumpuser/client1 directory on the remote server. It might take several minutes for the file to be created. Enter the following command to view the crash dump file:

   # ls /var/netdumpuser/client1/*
   /var/netdumpuser/client1/2009-07-16-13:22:
   vmcore

What to do next

When the dumping is done, the client system reboots to its regular kernel. You can find a record of the remote Kdump in the /var/log/messages directory, similar to the following entry:


Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.
Chapter 6. Setting up additional conditions to trigger crash dumps

Set up triggers for crash dumps in conditions besides kernel panic.

The Linux kernel typically calls the panic routine when it encounters an unrecoverable software or hardware error. The route prints a message on the console and then halts all operating system functions. If a system is configured with Kdump as described in this blueprint, the panic routine also saves a crash dump. There are, however, types of system failures that do not lead to a panic where an analysis of the crash dumps is still desirable.

Related reference:
Chapter 1, “Scope, requirements, and support,” on page 1

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Triggering crash dump on responsive systems using Magic SysRq keys

Linux supports the Magic SysRq keys, which are useful in failure situations where the system is still responsive to console input. For emergency shutdown, Magic SysRq keys allow you to trigger a crash dump. You can also use Magic SysRq keys to display debug information.

Procedure
1. To verify that the Magic SysRq keys are enabled, enter the following command:
   ```
   # sysctl kernel.sysrq
   ```
   If the command reports a value 0, like in this example, the Magic SysRq key are disabled.
   ```
   kernel.sysrq = 0
   ```
2. To enable the Magic SysRq keys, edit the `/etc/sysctl.conf` file and ensure that the following line is in the file:
   ```
   kernel.sysrq=1
   ```
3. Enter the following command to reload the setting:
   ```
   # sysctl -p
   ```
   You should see the report of the new value as part of the reload output:
   ```
   net.ipv4.icmp_echo_ignore_broadcasts = 1
   net.ipv4.conf.all.rp_filter = 1
   kernel.sysrq = 1
   ```
4. To verify that the setting is correct, trigger a crash dump using one of the following methods.
   In some cases of failure, the system console, which is often connected through the USB bus, might not be functional and the Magic SysRq keys might not work. Serial consoles often still work in these cases. Where possible, make sure that a serial console is available.
   - If the USB bus is functional, and the Magic SysRq keys work, use the following steps to trigger a crash dump:
     a. From the system console, enter the following key combination to see the help menu:
        ```
        Alt-SysRq-h
        ```
     b. Then enter the following key combination to create a crash dump:
If the USB bus is not functional, and the Magic SysRq key do not work, use a serial console to trigger a crash dump. From a serial console, which is accessed through Telnet, complete the following steps:

a. Press the Ctrl key and the right bracket (]) key.

```
$ telnet 192.168.0.3 2089
Trying 192.168.0.3...
Connected to 192.168.0.3.
Escape character is '^]'.
telnet>
```

b. Type `send brk` at the Telnet prompt, press Enter once, then immediately press the C key.

```
telnet> send brk
```

### What to do next

After you trigger the crash dump, the system is unresponsive while it saves the dump file. Subsequently, the system reboots. You should find a new crash dump file in your net dump server. If so, you have successfully enabled the magic SysRq keys in your machine. You can use this feature to create a crash dump whenever your machine fails but is still responsive over system or serial console.

**Related reference:**

[Chapter 1, “Scope, requirements, and support,” on page 1](#)

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

---

**Triggering crash dumps on non-responsive systems using NMI**

In failures where the system hangs and normal interrupts are disabled, you can use a non maskable interrupt (NMI) to trigger a panic and a crash dump. There are two ways to do trigger an NMI, but they cannot be used simultaneously.

**Related reference:**

[Chapter 1, “Scope, requirements, and support,” on page 1](#)

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

---

**Using the NMI Watchdog to detect hangs**

When the NMI watchdog is enabled, the system hardware is programmed to periodically generate an NMI. Each NMI invokes a handler in the Linux kernel to check the count of certain interrupts. If the handler detects that this count has not increased over a certain period of time, it assumes the system is hung. It then invokes the panic routine. If Kdump is enabled, the routine also saves a crash dump.

**Determining whether the NMI Watchdog is enabled**

To determine whether NMI watchdog is enabled, enter the following command. The included output indicates that there is no NMI count in all processors, thus NMI watchdog is disabled on this system:

```
# grep NMI /proc/interrupts
NMI: 0 0 0 0
```

**Enabling the NMI watchdog**

To enable the NMI watchdog, add `nmi_watchdog=1` or `nmi_watchdog=2` to your boot entry.
Note: Not all hardware supports the \texttt{nmi\_watchdog=1} boot parameter. Some hardware supports the \texttt{nmi\_watchdog=2} parameter, and some hardware supports neither parameter.

Edit the \texttt{/boot/grub/menu.lst} file to add the \texttt{nmi\_watchdog=1} parameter or the \texttt{nmi\_watchdog=2} parameter to your boot entry. This example shows the \texttt{/boot/grub/menu.lst} file in Red Hat Enterprise Linux:

\begin{verbatim}
title Red Hat Enterprise Linux Server (2.6.18-128.el5)
  root (hd0,0)
  kernel /vmlinuz-2.6.18-128.el5 ro root=/dev/sda nmi\_watchdog=1
  initrd /initrd-2.6.18-128.el5.img
\end{verbatim}

Reboot the machine. Enter the \texttt{grep} command repeatedly to view the NMI count. The following output shows that the NMI count on each processor increases rapidly. Thus the NMI watchdog is enabled by the \texttt{nmi\_watchdog=1} boot parameter.

\begin{verbatim}
# grep NMI /proc/interrupts
NMI: 2123797 2123681 2123608 2123535
# grep NMI /proc/interrupts
NMI: 2124855 2124739 2124666 2124593
# grep NMI /proc/interrupts
NMI: 2125981 2125865 2125792 2125719
# grep NMI /proc/interrupts
NMI: 2126692 2126576 2126503 2126430
# grep NMI /proc/interrupts
NMI: 2127406 2127290 2127217 2127144
\end{verbatim}

The following output shows the NMI count on each processor when the NMI watchdog is enabled by the \texttt{nmi\_watchdog=2} kernel boot option. The NMI counts increase slowly because the count depends on processor utilization, and the system in this example is idle.

\begin{verbatim}
Note: There are exceptional cases where a small number of NMI appears in the \texttt{/proc/interrupts} file when NMI watchdog is disabled.
\end{verbatim}

\begin{verbatim}
grep NMI /proc/interrupts
NMI: 187 107 293 199
# grep NMI /proc/interrupts
NMI: 187 107 293 199
# grep NMI /proc/interrupts
NMI: 187 107 293 200
\end{verbatim}

Now your system is ready to generate a crash dump in case it becomes unresponsive, but does not go into a panic state.

Related reference: \texttt{Chapter 1, “Scope, requirements, and support,” on page 13}

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Generating an NMI manually

An alternative to the NMI watchdog is to generate an NMI manually. This section describes how to configure the Linux kernel to call the panic routine when it receives an NMI with an unknown code. In many cases it is possible to generate such an NMI on a hung system in order to call the panic routine and cause a crash dump.

There are two methods to generate NMIs manually: by pressing the NMI button, and by using IPMI. Both methods depend on whether an implementation is provided on the particular hardware model. Also, both methods are incompatible with NMI watchdog.
Setting up Linux to call the panic routine when it receives an unknown NMI

Before you can generate a crash dump with a manually-generated NMI, you must configure the kernel to call the panic routine when it receives an unknown NMI.

**Procedure**

1. If you have enabled NMI watchdog, disable it by removing the `nmi_watchdog` kernel boot parameter from your boot entry, and then reboot.

   **Note:** Setting both the `unknown_nmi_panic` kernel boot parameter and the `nmi_watchdog` kernel boot parameter might cause the panic routine to start immediately on SLES 10. On Red Hat Enterprise Linux 5.3, the `unknown_nmi_panic` kernel boot parameter may disable the `nmi_watchdog` kernel boot parameter.

2. Enter the following command to determine whether the system is set to call the panic routine when it receives an unknown NMI:

   ```bash
   # sysctl kernel.unknown_nmi_panic
   ```

   If the system is not configured to call the panic routine when it receives and unknown NMI, this command produces the following output:

   ```bash
   kernel.unknown_nmi_panic = 0
   ```

   When using the default kernel setting, the Linux kernel logs the occurrence of an unknown NMI in `/var/log/messages`, but the system does not call the panic routine. The following messages are an example of the messages logged in the `/var/log/messages` file when the systems is not configured to call the panic routine for an unknown NMI:

   ```
   Jun 11 10:26:46 testsystem kernel: Uhhuh. NMI received for unknown reason 30.
   Jun 11 10:26:46 testsystem kernel: Do you have a strange power saving mode enabled?
   Jun 11 10:26:46 testsystem kernel: Dazed and confused, but trying to continue
   ```

3. To enable the kernel to call the panic routine when it receives an unknown NMI, edit the `/etc/sysctl.conf` file and ensure that the following line is in the file:

   ```bash
   kernel.unknown_nmi_panic = 1
   ```

4. Enter the following command to reload the setting. You should see the report of the new value as part of the reload output:

   ```bash
   # sysctl -p
   net.ipv4.icmp_echo_ignore_broadcasts = 1
   net.ipv4.conf.all.rp_filter = 1
   kernel.unknown_nmi_panic = 1
   ```

**What to do next**

Your system is now ready to call the panic routine when an unknown NMI is detected.
Generating an NMI by pressing the NMI button

Many IBM System x servers are equipped with NMI buttons. When the NMI button is pressed, the system generates an NMI. You can refer to the hardware manual to find out if your system has an NMI button and where it is located.

If the Linux kernel is configured as described in “Setting up Linux to call the panic routine when it receives an unknown NMI” on page 20, pressing the NMI button causes the kernel to call the panic routine and generate a crash dump.

Generating an NMI through IPMI

Many IBM System x platforms have the baseboard management controller (BMC) that supports the IPMI platform management standard. The BMC can be set up to allow out-of-band management commands issued from a remote system through the LAN interface. The IPMI standard describes a command that causes the BMC to issue an NMI to the system. This command does not work on all System x platforms.

For more information about configuring IPMI to allow remote access to the LAN interface, and how to use the `ipmitool` command, see the IPMI blueprint [http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/topic/liaai/ipmi/ipmikick.htm](http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/topic/liaai/ipmi/ipmikick.htm).

To issue an NMI through IPMI, run the following command from a remote system:

```
ipmitool -I lan -H <Host> -U <User ID> -a chassis power diag
```

If the system experiences a failure and remains unresponsive to the console or other input, the BMC is still be functioning and can be directed from a remote system to issue an NMI. If the Linux kernel is configured as described in “Setting up Linux to call the panic routine when it receives an unknown NMI” on page 20, this causes the system to call the panic routine and to save a crash dump.

Related reference:

Chapter 1, “Scope, requirements, and support,” on page 1

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.
Chapter 7. Configuring Remote Crash Dumps on Linux Systems

This blueprint shows you how to enable kernel crash dumps with the Kdump utility on systems running Red Hat Enterprise Linux 5.3 and SUSE Linux Enterprise Server (SLES) 10 SP2. It then walks you through the process of setting up a remote server to receive crash dumps. You can set up the remote net dump server on the same operating system version as the client. The blueprint ends with a discussion on how to enable the Kdump utility for additional unresponsive and hang conditions other than kernel panic.

Scope, requirements, and support

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Systems to which this information applies

System x running Linux (dump server and client) and PowerLinux (dump server only)

Intended audience

This document is written for system administrators who install and manage IBM System x servers running Red Hat Enterprise Linux or SUSE Linux Enterprise Server (SLES).

Scope and purpose

The blueprint covers the configuration of Kdump clients set to dump remotely to a Kdump network server. Because you can easily configure your system to dump locally using the distribution’s systems management tools, this configuration is not covered. In addition, the distributions support dumping over other network protocols besides SSH, but this configuration is also not covered.

For more information about these two configurations, see the documentation supplied with your distribution.

Hardware and software requirements

The instructions in this blueprint are written for Kdump servers and clients running the Red Hat Enterprise Linux (RHEL) 5.3 or SLES 10 SP2 operating systems. The Kdump server should have enough storage to receive the crash dumps from the clients.

Kdump clients are tested on IBM System x servers; Kdump servers are tested on IBM System x and System p servers. The Kdump utility is not supported if the Kdump client’s operating system distribution does not match the Kdump client machine’s.

Author names

Max Asbock

Other contributors

Monza Lui
Kersten Richter
IBM Services

Linux offers flexibility, options, and competitive total cost of ownership with a world class enterprise operating system. Community innovation integrates leading-edge technologies and best practices into Linux.

IBM is a leader in the Linux community with over 600 developers in the IBM Linux Technology Center working on over 100 open source projects in the community. IBM supports Linux on all IBM servers, storage, and middleware, offering the broadest flexibility to match your business needs.

For more information about IBM and Linux, go to [ibm.com/linux](https://www.ibm.com/linux)

IBM Support


The IBM developerWorks discussion forums let you ask questions, share knowledge, ideas, and opinions about technologies and programming techniques with other developerWorks users. Use the forum content at your own risk. While IBM will attempt to provide a timely response to all postings, the use of this developerWorks forum does not guarantee a response to every question that is posted, nor do we validate the answers or the code that are offered.

Typographic conventions

The following typographic conventions are used in this Blueprint:

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

Kdump overview

A kernel crash dump is the memory image of an operating system (OS) kernel that is written to a file. Typically, the system writes a crash dump file when the OS experiences a serious problem such as a hang or crash. You can open the saved crash dump file with the Crash analysis utility to determine the root cause of OS problems.

Red Hat Enterprise Linux and SUSE Linux Enterprise Server provide Kdump software to capture and save crash dumps. The Kdump-enabled OS kernel loads an additional standby Kdump kernel into reserved memory. When a hang or crash triggers a crash dump, the OS kernel boots into the Kdump kernel without a system reset. The standby Kdump kernel then writes an image of the saved memory into a crash dump file. The Kdump utility can save crash dump files on a local disk or through the network onto a remote system configured to store the dump. This blueprint only covers the remote dumping setup.
Setting up a net dump server

You can configure Kdump to dump to a remote server over a network connection. To do this, configure two systems: one that will receive the dump (the server) and another that creates a dump (the client). This section shows how to configure a server that receives crash dumps from remote systems through Secure Shell (SSH). The instructions are the same for Red Hat Enterprise Linux 5.3 and SLES 10 SP2.

About this task

There are two considerations when choosing the net dump server:

- The server should be accessible for the Kdump client directly through SSH without additional authentication.
- Also, the server must have enough disk space to handle the dumps.

Because the dump files from remote systems are exact replicas of their system memory, the size of each dump file will be equivalent to the memory size.

Procedure

1. Create a net dump user account and a directory to hold the crash dumps. In this example, netdumpuser is used as the user and group name, and /var/netdumpuser is used as the directory to hold the dumps. You can choose another user name and directory.
   a. Create a new group named netdumpuser:
      
      ```
      # groupadd -r netdumpuser
      ```
   b. Create the new user (system account) netdumpuser with the following attributes:
      
      - group netdumpuser
      - home directory /var/netdumpuser, which contains the crash dump files
      - shell /bin/bash
      - user account comment Network crash dump user

      To create the new user with these attributes, enter the following command:
      
      ```
      # useradd -c "Network Crash dump user" -r -g netdumpuser -s /bin/bash \
      -d /var/netdumpuser netdumpuser
      ```
   c. Create all components of the /var/netdumpuser/ssh directory, including the /var/netdumpuser directory, with the following attributes:
      
      - user and group ownership set to netdumpuser
      - permission mode set to 700

      To create these components, enter the following command:
      
      ```
      # install -o netdumpuser -g netdumpuser -m 700 -d /var/netdumpuser/.ssh
      ```
   d. Verify that the previous commands executed successfully by typing:
      
      ```
      # ls -al /var/netdumpuser/*
      ```
      
      You will see output similar to the following:
      
      ```
      drwx------ 2 netdumpuser netdumpuser 4096 Jul 1 22:10 .ssh
      ```
   e. Change the password of the netdumpuser user, as follows:
# passwd netdumpuser
Changing password for user netdumpuser.
You should see the following output:
passwd: all authentication tokens updated successfully.

Note: This password will be needed later on to authenticate for dumping to this machine.

f. If you want to configure the server to receive network dumps from all machines in your network, you can create multiple categorical subdirectories within the `/var/netdumpuser` directory. This helps you to easily manage dumps from different machines. To do so, enter the following commands:

    # mkdir /var/netdumpuser/client1
    # mkdir /var/netdumpuser/client2
    # mkdir /var/netdumpuser/client3

g. Type the following commands to set the required permissions, ownership, and group ownership on the `/var/netdumpuser` directory:

    # chown -R netdumpuser:netdumpuser /var/netdumpuser
    # chmod -R 700 /var/netdumpuser

h. Verify that the commands in Step g ran successfully. You should see output from the command similar to the following output that shows that all files are owned by the `netdumpuser` user and group:

    # ls -al /var/netdumpuser/
    ...
    drwx------ 5 netdumpuser netdumpuser 4096 Jul 1 22:24 .
    drwxr-xr-x 24 root root 4096 Jul 1 22:09 ..
    drwx------ 2 netdumpuser netdumpuser 4096 Jul 1 22:09 client1
    drwx------ 2 netdumpuser netdumpuser 4096 Jul 1 22:09 client2
    drwx------ 2 netdumpuser netdumpuser 4096 Jul 1 22:09 client3
    drwx------ 2 netdumpuser netdumpuser 4096 Jul 1 22:10 .ssh

2. Try connecting to the server from the client with ssh. If the connection is successful, you are done with this section and can continue with Chapter 4, “Setting up Kdump on the dump client,” on page 9. If the connection is not successful, continue to Step 3.

3. Ensure that the sshd daemon is running to receive the dumps. To verify, type the following command:

    # service sshd status

If sshd is running, you should see output similar to the following:

    Checking for service sshd running

Or you might see output similar to the following:

    openssh-daemon (pid 4290) is running...

If you do not see the previous output, type the following commands:

    # chkconfig sshd on
    # service sshd start

4. If a firewall is enabled on the net dump server and the ssh service is not allowed, change the security settings to allow ssh. Complete the steps appropriate for the distribution you are using:
   - Allow SSH through the firewall on Red Hat Enterprise Linux 5.3 with the following steps:
     a. In a X Window System environment, start the graphical security configuration interface by typing:

         # system-config-securitylevel

         Note: If you are not running X Window System, you can run system-config-securitylevel-tui to use the text-based user interface.

     b. If the firewall is disabled, you do not have to make any additional changes.

     c. If the firewall is enabled, in the Trusted services panel, click SSH if it is not selected, and complete Step d through Step f.

     d. Click OK to accept the change.
e. Click Yes to confirm overriding of previous configuration.

f. Test the configuration by trying an ssh connection to the machine.

- Allow SSH through the firewall on SLES 10 SP2 with the following steps:
  a. Start the firewall configuration graphical user interface by entering the following command:
     ```bash
     # yast2 firewall
     ```
  
b. Optional: If the firewall is disabled, you do not have to make any additional changes. Go to
     [Chapter 4, “Setting up Kdump on the dump client,” on page 9](#).
  
c. Optional: If the firewall is enabled, click Allowed Services in the left panel and complete Step d through Step h.
  
d. If Secure Shell Server or SSH is not listed in the Allowed Services list, select it from the Services to
    Allow menu and click Add.
  
e. Go back to the previous screen by clicking Start-Up in the left panel.
  
f. Click Save Settings and Restart Firewall Now.
  
g. Click Next, and then click Finish.
  
h. Test the configuration by trying an ssh connection to the machine.

**What to do next**

When you are finished, go to [Chapter 4, “Setting up Kdump on the dump client,” on page 9](#).

**Related reference:**

[Chapter 1, “Scope, requirements, and support,” on page 1](#)

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

---

**Setting up Kdump on the dump client**

You can configure Kdump on RHEL 5.3 or SLES 10 SP2. The configuration is supported only if the client and the server are running on the same operating system distribution. Skip to the section which corresponds to the distribution running your dump client.

**Related reference:**

[Chapter 1, “Scope, requirements, and support,” on page 1](#)

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

**Configuring Kdump on a Red Hat Enterprise Linux 5.3 system**

During installation, you are prompted to configure Kdump. However the configuration options are limited to enabling it and setting the Kdump memory size. Enabling Kdump at that time guarantees a local Kdump to the `/var/crash` directory.

**About this task**

For a remote setup, configure Kdump on an installed system. The following steps configure the client machine Kdump to the remote net dump server that you set up in the previous section.

**Procedure**

1. Start the Kdump graphical interface by typing the following command in an X Window System environment (using ssh -X for example):
2. Check Enable kdump, as follows:

3. In the Path box, enter the path to write the crash dumps to the net dump server. This example uses var/netdumpuser/client1

4. Click Edit Location. In the window, select ssh from the Select a location type menu. Enter <username>@<hostname of the dump server> in the Enter Location field.

5. On the system-config-kdump window, verify that the values in the Location field and the Path field are correct and click OK.

The output on your system is similar to the following example:
6. Enter the following command to allow authentication with the net dump server and set up a public transfer:
   
   ```
   # service kdump propagate
   ```

7. Either reboot the system or restart the **kdump** service, using one of the following commands:
   - If you are enabling Kdump for the first time, reboot the system so that memory is reserved for the Kdump kernel. Enter the following command:
     ```
     # reboot
     ```
   - If Kdump is already enabled, restart the **kdump** service by entering the following command:
     ```
     # service kdump restart
     ```

8. Verify that **Kdump** daemon is running by entering the following command:
   ```
   # service kdump status
   ```
   You should see the following output:
   Kdump is operational

9. Verify that the public key authentication works by entering the following command:
   ```
   # ssh netdumpuser@netdump_server
   ```
   You should now be logged into **netdump_server** as netdumpuser without a password.

**Related reference:**

- [Chapter 1, “Scope, requirements, and support,” on page 1](#)

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

### Configuring Kdump on SLES 10 SP2

During installation, you are prompted to configure Kdump. However, the configuration options are limited to enabling Kdump and setting the Kdump memory size. If you enabled Kdump at the time of installation, a local Kdump saves to the `/var/crash` directory. Follow this procedure to change the configuration of Kdump on a SLES 10 SP2 client to save dumps to a remote server through the SSH protocol.

**Before you begin**

Kdump must be enabled on an installed system.

**Procedure**

1. Configure Kdump using the yast2 configuration tool.
   a. In an X Window System environment, start the Kdump configuration tool by entering the following command:
      ```
      # yast2 kdump
      ```
   b. Select **Enable Kdump** if it is not already selected.
   c. In the left panel, click **Dump Target**.
   d. Select **SSH** from the **Saving Target for Kdump Image** list.
e. Specify the location and authentication values for receiving and storing the crash dumps:
   - In the **Server Name** field, enter the host name of the remote server receiving the crash dumps.
   - In the **Directory on Server** field, enter the directory where the receiving server stores the crash dumps.
   - In the **User Name** field, enter the user name to be used for SSH authentication on the receiving server.

   **Note:** The directory and user name specified here must match the settings specified in Chapter 3, “Setting up a net dump server,” on page 5.

2. Transfer an SSH public key to the net dump server. Transferring the SSH public key to the net dump server grants you access to the net dump server without a password during future logins.
   a. Generate an SSH private/public key pair by entering the following command:

   ```
   # ssh-keygen -N '' -C 'passthrough key' -t dsa
   ```

   You should see the following output:
   Generating public/private dsa key pair.
   Enter file in which to save the key (/root/.ssh/id_dsa):
   Your identification has been saved in /root/.ssh/id_dsa.
   Your public key has been saved in /root/.ssh/id_dsa.pub.
   The key fingerprint is:
b. Add the public SSH key to the `/home/netdumpuser/.ssh/authorized_keys` file on the remote dump server. Doing so allows password-less login to the dump server as netdumpuser. Type the following commands:

```bash
# ssh netdumpuser@netdump_server 'cat >> ~/.ssh/authorized_keys' < /root/.ssh/id_dsa.pub
```

c. When you are prompted for a password, enter the password for netdumpuser.
d. Verify that the public key authentication works by entering the following command:

```bash
# ssh netdumpuser@netdump_server
```

You should now be able to log in to `netdump_server` as netdumpuser without entering a password.

**Related reference:**

Chapter 1, “Scope, requirements, and support,” on page 1

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

---

### Testing Kdump

Follow this procedure to test if the Kdump setup is working.

**About this task**

Initiating a Kdump causes the client system to reboot. Before starting this procedure, ensure that the client system is ready to be rebooted. Also ensure that the remote server has enough disk space to save the dump file.

On the client system:

**Procedure**

1. Write any outstanding data to disk by entering the following command:

   ```bash
   # sync
   ```

2. Trigger the crash dump by entering the following command:

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

   The client system becomes unresponsive.

3. A crash dump file is created in the `/var/netdumpuser/client1` directory on the remote server. It might take several minutes for the file to be created. Enter the following command to view the crash dump file:

   ```bash
   # ls /var/netdumpuser/client1/*
   ```

**What to do next**

When the dumping is done, the client system reboots to its regular kernel. You can find a record of the remote Kdump in the `/var/log/messages` directory, similar to the following entry:

```
Jul 16 13:22:13 client1 kdump[3070]: Saving 3970 MB crash dump to
ssh://netdumpuser@192.168.0.15/var/netdumpuser/client1/2009-07-16-13:22
```
Setting up additional conditions to trigger crash dumps

Set up triggers for crash dumps in conditions besides kernel panic.

The Linux kernel typically calls the panic routine when it encounters an unrecoverable software or hardware error. The route prints a message on the console and then halts all operating system functions. If a system is configured with Kdump as described in this blueprint, the panic routine also saves a crash dump. There are, however, types of system failures that do not lead to a panic where an analysis of the crash dumps is still desirable.

Triggering crash dump on responsive systems using Magic SysRq keys

Linux supports the Magic SysRq keys, which are useful in failure situations where the system is still responsive to console input. For emergency shutdown, Magic SysRq keys allow you to trigger a crash dump. You can also use Magic SysRq keys to display debug information.

Procedure

1. To verify that the Magic SysRq keys are enabled, enter the following command:
   
   ```bash
   # sysctl kernel.sysrq
   ```
   
   If the command reports a value 0, like in this example, the Magic SysRq key are disabled.
   
   ```bash
   kernel.sysrq = 0
   ```
   
2. To enable the Magic SysRq keys, edit the `/etc/sysctl.conf` file and ensure that the following line is in the file:
   
   ```bash
   kernel.sysrq = 1
   ```
   
3. Enter the following command to reload the setting:
   
   ```bash
   # sysct1 -p
   ```
   
   You should see the report of the new value as part of the reload output:
   
   ```bash
   net.ipv4.icmp_echo_ignore_broadcasts = 1
   net.ipv4.conf.all.rp_filter = 1
   kernel.sysrq = 1
   ```
   
4. To verify that the setting is correct, trigger a crash dump using one of the following methods.

   In some cases of failure, the system console, which is often connected through the USB bus, might not be functional and the Magic SysRq keys might not work. Serial consoles often still work in these cases. Where possible, make sure that a serial console is available.

   - If the USB bus is functional, and the Magic SysRq keys work, use the following steps to trigger a crash dump:
     
     a. From the system console, enter the following key combination to see the help menu:
Then enter the following key combination to create a crash dump:

Alt-SysRq-c

- If the USB bus is not functional, and the Magic SysRq key do not work, use a serial console to trigger a crash dump. From a serial console, which is accessed through Telnet, complete the following steps:

  a. Press the Ctrl key and the right bracket (]) key.

    $ telnet 192.168.0.3 2089
    Trying 192.168.0.3...
    Connected to 192.168.0.3.
    Escape character is '^].'
    telnet>

  b. Type send brk at the Telnet prompt, press Enter once, then immediately press the C key.

    telnet> send brk

**What to do next**

After you trigger the crash dump, the system is unresponsive while it saves the dump file. Subsequently, the system reboots. You should find a new crash dump file in your net dump server. If so, you have successfully enabled the magic SysRq keys in your machine. You can use this feature to create a crash dump whenever your machine fails but is still responsive over system or serial console.

Related reference:
Chapter 1, “Scope, requirements, and support,” on page 1

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

**Triggering crash dumps on non-responsive systems using NMI**

In failures where the system hangs and normal interrupts are disabled, you can use a non maskable interrupt (NMI) to trigger a panic and a crash dump. There are two ways to do trigger an NMI, but they cannot be used simultaneously.

Related reference:
Chapter 1, “Scope, requirements, and support,” on page 1

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

**Using the NMI Watchdog to detect hangs**

When the NMI watchdog is enabled, the system hardware is programmed to periodically generate an NMI. Each NMI invokes a handler in the Linux kernel to check the count of certain interrupts. If the handler detects that this count has not increased over a certain period of time, it assumes the system is hung. It then invokes the panic routine. If Kdump is enabled, the routine also saves a crash dump.

**Determining whether the NMI Watchdog is enabled**

To determine whether NMI watchdog is enabled, enter the following command. The included output indicates that there is no NMI count in all processors, thus NMI watchdog is disabled on this system:

```
# grep NMI /proc/interrupts
NMI: 0 0 0 0
```
Enabling the NMI watchdog

To enable the NMI watchdog, add `nmi_watchdog=1` or `nmi_watchdog=2` to your boot entry.

**Note:** Not all hardware supports the `nmi_watchdog=1` boot parameter. Some hardware supports the `nmi_watchdog=2` parameter, and some hardware supports neither parameter.

Edit the `/boot/grub/menu.lst` file to add the `nmi_watchdog=1` parameter or the `nmi_watchdog=2` parameter to your boot entry. This example shows the `/boot/grub/menu.lst` file in Red Hat Enterprise Linux:

```
title Red Hat Enterprise Linux Server (2.6.18-128.el5)
  root (hd0,0)
  kernel /vmlinuz-2.6.18-128.el5 ro root=/dev/sda nmi_watchdog=1
  initrd /initrd-2.6.18-128.el5.img
```

Reboot the machine. Enter the `grep` command repeatedly to view the NMI count. The following output shows that the NMI count on each processor increases rapidly. Thus the NMI watchdog is enabled by the `nmi_watchdog=1` boot parameter.

```
# grep NMI /proc/interrupts
NMI: 2123797 2123681 2123608 2123535
# grep NMI /proc/interrupts
NMI: 2124855 2124739 2124666 2124593
# grep NMI /proc/interrupts
NMI: 2125981 2125865 2125792 2125719
# grep NMI /proc/interrupts
NMI: 2126692 2126576 2126503 2126430
# grep NMI /proc/interrupts
NMI: 2127406 2127290 2127217 2127144
```

The following output shows the NMI count on each processor when the NMI watchdog is enabled by the `nmi_watchdog=2` kernel boot option. The NMI counts increase slowly because the count depends on processor utilization, and the system in this example is idle.

**Note:** There are exceptional cases where a small number of NMI appears in the `/proc/interrupts` file when NMI watchdog is disabled.

```
grep NMI /proc/interrupts
NMI: 187 107 293 199
```

Now your system is ready to generate a crash dump in case it becomes unresponsive, but does not go into a panic state.

**Related reference:**
[Chapter 1, “Scope, requirements, and support,” on page 1](#)

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

**Generating an NMI manually**

An alternative to the NMI watchdog is to generate an NMI manually. This section describes how to configure the Linux kernel to call the panic routine when it receives an NMI with an unknown code. In many cases it is possible to generate such an NMI on a hung system in order to call the panic routine and cause a crash dump.
There are two methods to generate NMIs manually: by pressing the NMI button, and by using IPMI. Both methods depend on whether an implementation is provided on the particular hardware model. Also, both methods are incompatible with NMI watchdog.

Related reference:
Chapter 1, “Scope, requirements, and support,” on page 1

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Setting up Linux to call the panic routine when it receives an unknown NMI:

Before you can generate a crash dump with a manually-generated NMI, you must configure the kernel to call the panic routine when it receives an unknown NMI.

Procedure
1. If you have enabled NMI watchdog, disable it by removing the nmi_watchdog kernel boot parameter from your boot entry, and then reboot.

   **Note:** Setting both the unknown_nmi_panic kernel boot parameter and the nmi_watchdog kernel boot parameter might cause an the panic routine to start immediately on SLES 10. On Red Hat Enterprise Linux 5.3, the unknown_nmi_panic kernel boot parameter may disable the nmi_watchdog kernel boot parameter.

2. Enter the following command to determine whether the system is set to call the panic routine when it receives an unknown NMI:

   \# sysctl kernel.unknown_nmi_panic

   If the system is not configured to call the panic routine when it receives and unknown NMI, this command produces the following output:

   kernel.unknown_nmi_panic = 0

   When using the default kernel setting, the Linux kernel logs the occurrence of an unknown NMI in /var/log/messages, but the system does not call the panic routine. The following messages are an example of the messages logged in the /var/log/messages file when the systems is not configured to call the panic routine for an unknown NMI:

   Jun 11 10:26:46 testsystem kernel: Uhhuh. NMI received for unknown reason 30.
   Jun 11 10:26:46 testsystem kernel: Do you have a strange power saving mode enabled?
   Jun 11 10:26:46 testsystem kernel: Dazed and confused, but trying to continue

3. To enable the kernel to call the panic routine when it receives an unknown NMI, edit the /etc/sysctl.conf file and ensure that the following line is in the file:

   kernel.unknown_nmi_panic = 1

4. Enter the following command to reload the setting. You should see the report of the new value as part of the reload output:

   \# sysctl -p

   net.ipv4.icmp_echo_ignore_broadcasts = 1
   net.ipv4.conf.all rp_filter = 1
   kernel.unknown_nmi_panic = 1

What to do next

Your system is now ready to call the panic routine when an unknown NMI is detected.
Generating an NMI by pressing the NMI button:

Many IBM System x servers are equipped with NMI buttons. When the NMI button is pressed, the system generates an NMI. You can refer to the hardware manual to find out if your system has an NMI button and where it is located.

If the Linux kernel is configured as described in "Setting up Linux to call the panic routine when it receives an unknown NMI" on page 20, pressing the NMI button causes the kernel to call the panic routine and generate a crash dump.

Related reference:
Chapter 1, “Scope, requirements, and support,” on page 1
This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Generating an NMI through IPMI:

Many IBM System x platforms have the baseboard management controller (BMC) that supports the IPMI platform management standard. The BMC can be set up to allow out-of-band management commands issued from a remote system through the LAN interface. The IPMI standard describes a command that causes the BMC to issue an NMI to the system. This command does not work on all System x platforms.

For more information about configuring IPMI to allow remote access to the LAN interface, and how to use the `ipmitool` command, see the IPMI blueprint [http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/topic/liaai/ipmi/ipmikick.htm](http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/topic/liaai/ipmi/ipmikick.htm)

To issue an NMI through IPMI, run the following command from a remote system:

```
ipmitool -I lan -H <Host> -U <User ID> -a chassis power diag
```

If the system experiences a failure and remains unresponsive to the console or other input, the BMC is still be functioning and can be directed from a remote system to issue an NMI. If the Linux kernel is configured as described in "Setting up Linux to call the panic routine when it receives an unknown NMI" on page 20, this causes the system to call the panic routine and to save a crash dump.

Related reference:
Chapter 1, “Scope, requirements, and support,” on page 1
This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.
Appendix. Related information and downloads

Related information

- Using Intelligent Platform Management Interface (IPMI) on IBM Linux Platforms
  http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/topic/liaai/ipmi/ipmikick.htm

- developerWorks Systems Management Blueprint Community Forum

Related reference:
Chapter 1, “Scope, requirements, and support,” on page 1

This blueprint applies to System x running Linux (dump server and client) and PowerLinux (dump server only). You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.
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 grant 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.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Dept. LRAS/Bldg. 903
11501 Burnet Road
Austin, TX 78758-3400
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document 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.
For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

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

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

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.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

**Trademarks**

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® and ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at Copyright and trademark information at www.ibm.com/legal/copytrade.shtml

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Java™ and all Java-based trademarks and logos are registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.