Red Hat Enterprise Linux 8

Configuring device mapper multipath

Using the Device Mapper Multipath feature

Last Updated: 2020-08-04

Red Hat Enterprise Linux 8 Configuring device mapper multipath

Using the Device Mapper Multipath feature

Legal Notice

Abstract

This documentation collection provides instructions on how to configure and manage the Device Mapper Multipath (DM-Multipath) feature on Red Hat Enterprise Linux 8.

Table of Contents

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your input on our documentation. Please let us know how we could make it better. To do so:

- For simple comments on specific passages:
  1. Make sure you are viewing the documentation in the *Multi-page HTML* format. In addition, ensure you see the **Feedback** button in the upper right corner of the document.
  2. Use your mouse cursor to highlight the part of text that you want to comment on.
  3. Click the **Add Feedback** pop-up that appears below the highlighted text.
  4. Follow the displayed instructions.
- For submitting more complex feedback, create a Bugzilla ticket:
  1. Go to the Bugzilla website.
  2. As the Component, use **Documentation**.
  3. Fill in the **Description** field with your suggestion for improvement. Include a link to the relevant part(s) of documentation.
  4. Click **Submit Bug**.

CHAPTER 1. OVERVIEW OF DEVICE MAPPER MULTIPATHING

With Device mapper multipathing (DM Multipath), you can configure multiple I/O paths between server nodes and storage arrays into a single device. These I/O paths are physical Storage Area Network (SAN) connections that can include separate cables, switches, and controllers. Multipathing aggregates the I/O paths and creates a new device that consists of the aggregated paths.

DM Multipath provides:

- Redundancy
  DM Multipath can provide failover in an active/passive configuration. In an active/passive configuration, only half the paths are used at any time for I/O. If any element of an I/O path (the cable, switch, or controller) fails, DM Multipath switches to an alternate path.

- Improved Performance
  DM Multipath can be configured in an active/active mode, where I/O is spread over the paths in a round-robin fashion. In some configurations, DM Multipath can detect loading on the I/O paths and dynamically rebalance the load.

1.1. DIFFERENT DM MULTIPATH CONFIGURATIONS

Following are the few DM Multipath configuration examples:

- Section 1.1.1, “Active/Passive multipath configuration with one RAID device”
- Section 1.1.2, “Active/Passive multipath configuration with two RAID devices”
- Section 1.1.3, “Active/Active multipath configuration with one RAID device”

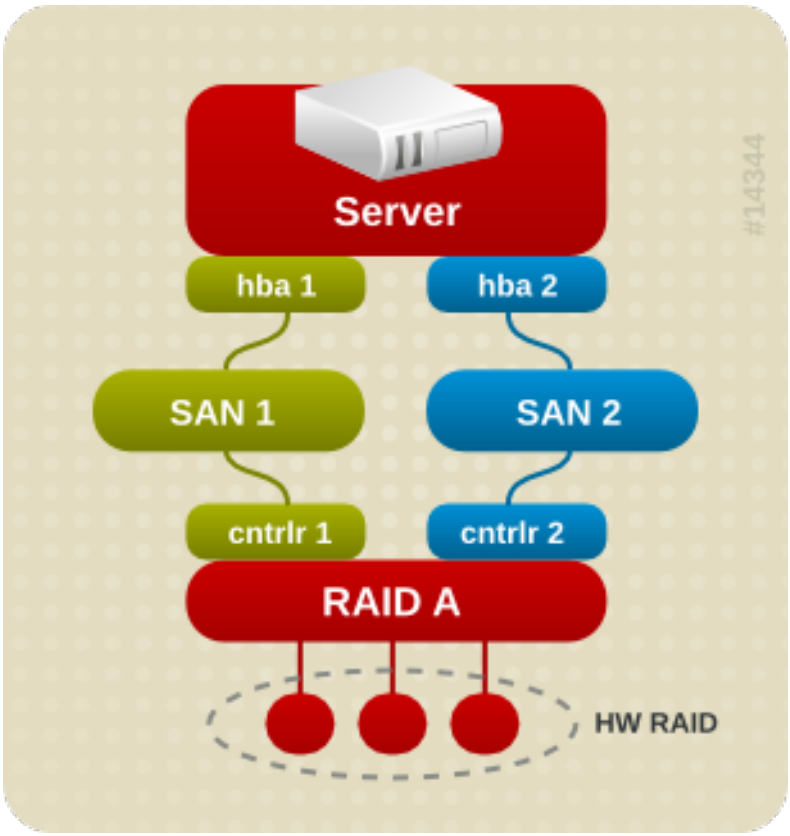
1.1.1. Active/Passive multipath configuration with one RAID device

In this configuration, there are two Host Bus Adapters (HBAs) on the server, two SAN switches, and two RAID controllers. Following are the possible failure in this configuration:

- HBA failure
- Fibre Channel cable failure
- SAN switch failure
- Array controller port failure

With DM Multipath configured, a failure at any of these points causes DM Multipath to switch to the alternate I/O path. Figure 1.1, “Active/Passive multipath configuration with one RAID device” describes the configuration with two I/O paths from the server to a RAID device. Here, there is one I/O path that goes through **hba1**, **SAN1**, and **cntrlr1** and a second I/O path that goes through **hba2**, **SAN2**, and **cntrlr2**.

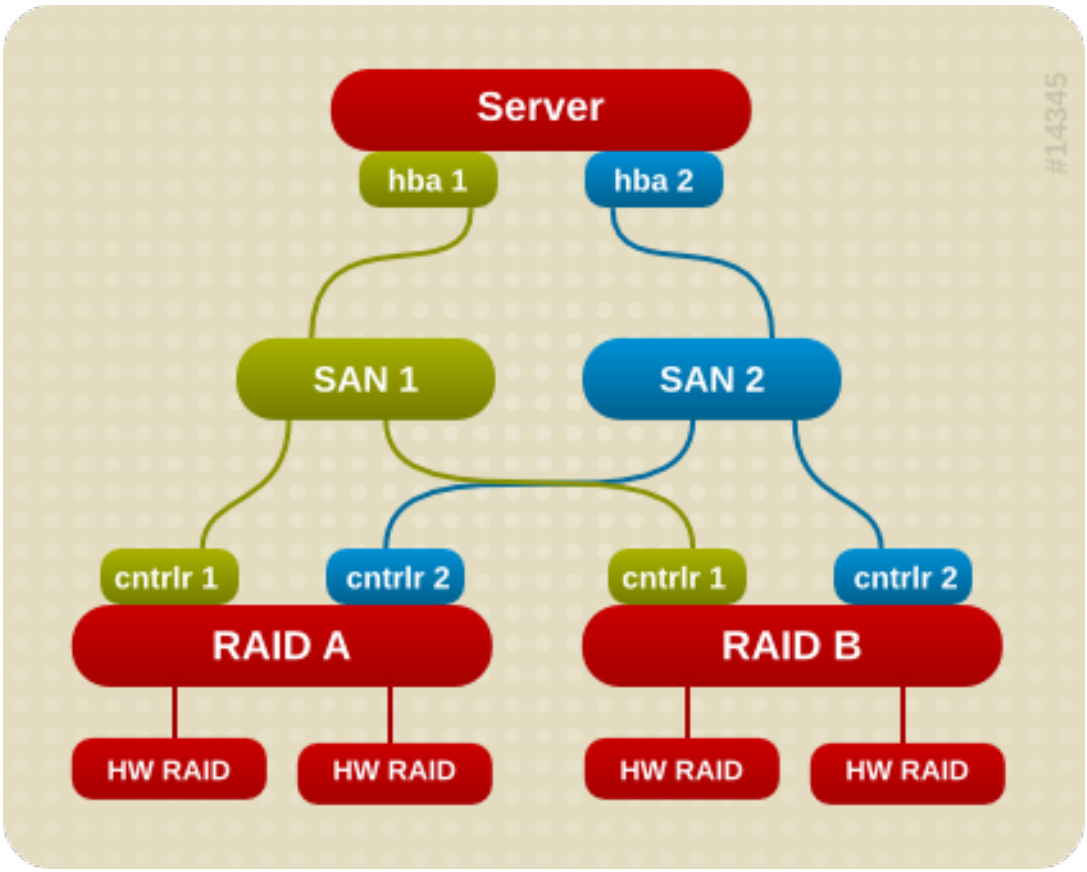
Figure 1.1. Active/Passive multipath configuration with one RAID device



1.1.2. Active/Passive multipath configuration with two RAID devices

In this configuration, there are two HBAs on the server, two SAN switches, and two RAID devices with two RAID controllers each. With DM Multipath configured, a failure at any of the points of the I/O path to either of the RAID devices causes DM Multipath to switch to the alternate I/O path for that device. Figure 1.2, “Active/Passive multipath configuration with two RAID device” describes the configuration with two I/O paths to each RAID device. Here, there are two I/O paths to each RAID device.

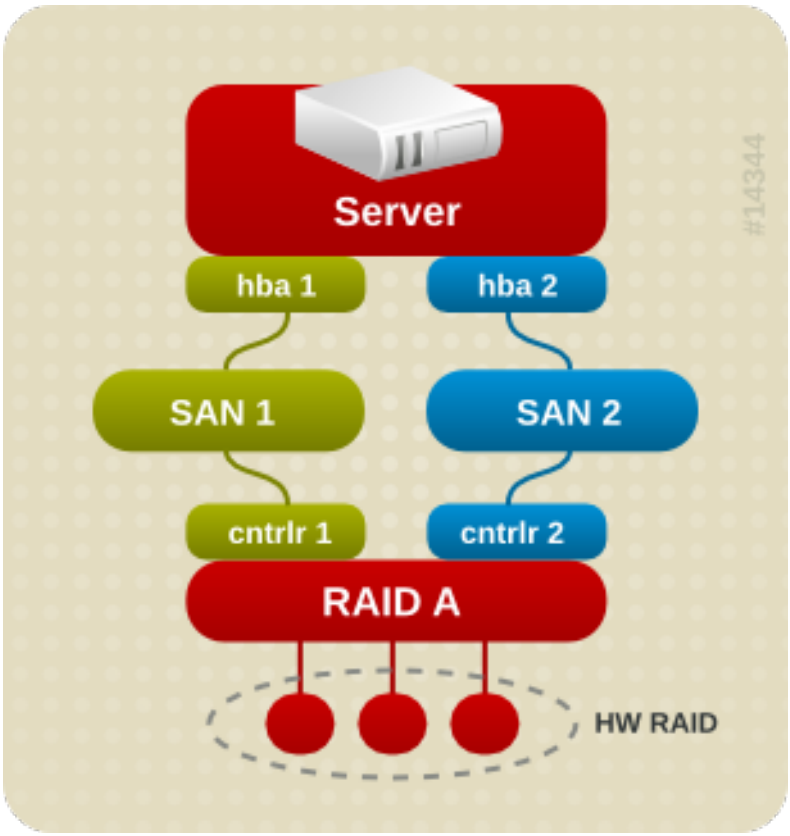
Figure 1.2. Active/Passive multipath configuration with two RAID device



1.1.3. Active/Active multipath configuration with one RAID device

In this configuration, there are two HBAs on the server, two SAN switch, and two RAID controllers. Figure 1.3, "Active/Active multipath configuration with one RAID device" describes the configuration with two I/O paths from the server to a storage device. Here, I/O can be spread among these two paths.

Figure 1.3. Active/Active multipath configuration with one RAID device



1.2. DM MULTIPATH COMPONENTS

Table 1.1, “Components of DM Multipath” describes the DM Multipath components.

Table 1.1. Components of DM Multipath

Component	Description
dm_multipath kernel module	Reroutes I/O and supports failover for paths and path groups.
mpathconf utility	Configures and enables device mapper multipathing.
multipath command	Lists and configures the multipath devices. It is also executed by **udev** whenever a block device is added, to determine if the device should be part of a multipath device or not.
multipathd daemon	Automatically creates and removes multipath devices and monitors paths; as paths fail and come back, it may update the multipath device. Allows interactive changes to multipath devices. Reload the service if there are any changes to the **/etc/multipath.conf** file.

kpartx command	Creates device mapper devices for the partitions on a device. This command is automatically executed by **udev** when multipath devices are created to create partition devices on top of them. The **kpartx** command is provided in its own package, but the **device-mapper-multipath** package depends on it.
mpathpersist	Sets up **SCSI-3** persistent reservations on multipath devices. This command works similarly to the way **sg_persist** works for SCSI devices that are not multipathed, but it handles setting persistent reservations on all paths of a multipath device. It coordinates with **multipathd** to ensure that the reservations are set up correctly on paths that are added later. To use this functionality, the **reservation_key** attribute must be defined in the **/etc/multipath.conf** file. Otherwise the **multipathd** daemon will not check for persistent reservations for newly discovered paths or reinstated paths.

Additional resources

- The **multipath** man page.
- The **multipathd** man page.
- The **/etc/multipath.conf** file.

CHAPTER 2. MULTIPATH DEVICES

Without DM Multipath, system treats each path from a server node to a storage controller as a separate device, even when the I/O path connects the same server node to the same storage controller. DM Multipath provides a way of organizing the I/O paths logically, by creating a single multipath device on top of the underlying devices.

2.1. MULTIPATH DEVICE IDENTIFIERS

When new devices are under the control of DM Multipath, these devices are created in the **/dev/mapper/** and **/dev/** directory. Any devices of the form **/dev/dm-X** are for internal use only and should never be used by the administrator directly.

Multipath device names:

- When the **user_friendly_names** configuration option is set to **no**, the name of the multipath device is set to World Wide Identifier (WWID). By default, the name of a multipath device is set to its WWID. The device name would be **/dev/mapper/WWID**. It is also created in the **/dev/** directory, named as **/dev/dm-X**.

- Alternately, you can set the **user_friendly_names** option to **yes** in the **/etc/multipath.conf** file. This sets the **alias** in the **multipath** section to a node-unique name of the form **mpathN**. The device name would be **/dev/mapper/mpathN** and **/dev/dm-X**. But the device name is not guaranteed to be the same on all nodes using the multipath device. Similarly, if you set the **alias** option in the **/etc/multipath.conf** file, the name is not automatically consistent across all nodes in the cluster.
  This should not cause any difficulties if you use LVM to create logical devices from the multipath device. To keep your multipath device names consistent in every node, Red Hat recommends to disable the **user_friendly_names** option.

For example, a node with two HBAs attached to a storage controller with two ports by means of a single unzoned FC switch sees four devices: **/dev/sda**, **/dev/sdb**, **/dev/sdc**, and **/dev/sdd**. DM Multipath creates a single device with a unique WWID that reroutes I/O to those four underlying devices according to the multipath configuration.

In addition to the **user_friendly_names** and **alias** options, a multipath device also has other attributes. You can modify these attributes for a specific multipath device by creating an entry for that device in the **multipaths** section of the **/etc/multipath.conf** file.

Additional resources

- The **multipath** man page.
- The **multipath.conf** man page.
- The **/etc/multipath.conf** file.
- Section 1.2, “DM Multipath components” .

2.2. MULTIPATH DEVICES IN LOGICAL VOLUMES

After creating multipath devices, you can use the multipath device names just as you would use a physical device name when creating an LVM physical volume. For example, if **/dev/mapper/mpatha** is the name of a multipath device, the **pvcreate /dev/mapper/mpatha** command marks **/dev/mapper/mpatha** as a physical volume.

You can use the resulting LVM physical device when you create an LVM volume group just as you would use any other LVM physical device.

NOTE

If you attempt to create an LVM physical volume on a whole device on which you have configured partitions, the **pvcreate** command fails. The Anaconda and Kickstart installation programs create empty partition tables if you do not specify otherwise for every block device. If you want to use the whole device instead of creating a partition, remove the existing partitions from the device. You can remove existing partitions with the **kpartx -d** device command and the **fdisk** utility. If your system has block devices that are greater than 2Tb, use the **parted** utility to remove partitions.

When you create an LVM logical volume that uses **active/passive** multipath arrays as the underlying physical devices, you can optionally, include filters in the **/etc/lvm/lvm.conf** file to exclude the disks that underline the multipath devices. This is because if the array automatically changes the active path to the passive path when it receives I/O, multipath will failover and failback whenever LVM scans the passive path if these devices are not filtered. For **active/passive** arrays that require a command to make the passive path active, LVM prints a warning message when this occurs.

To filter all the **sd** devices in the **/etc/lvm/lvm.conf** file, add the **filter = ["r/block/", "r/disk/", "r/sd./", "a/./"]** filter in the **devices** section of the file.

Additional resources

- The **lvm.conf** man page.
- Section 1.2, “DM Multipath components” .

CHAPTER 3. SETTING UP DM MULTIPATH

Before setting up DM Multipath on your system, ensure that your system has been updated and includes the **device-mapper-multipath** package.

3.1. BASIC DM MULTIPATH SETUP

You set up DM Multipath with the **mpathconf** utility, which creates the multipath configuration file **/etc/multipath.conf**.

- If the **/etc/multipath.conf** file already exists, the **mpathconf** utility will edit it.
- If the **/etc/multipath.conf** file does not exist, the **mpathconf** utility will create the **/etc/multipath.conf** file from scratch.

For more information on the **mpathconf** utility, see the **mpathconf(8)** man page.

If you do not need to edit the **/etc/multipath.conf** file, you can set up DM Multipath for a basic failover configuration by running the following **mpathconf** command. This command enables the multipath configuration file and starts the **multipathd** daemon.

```
# mpathconf --enable --with_multipathd y
```

If you need to edit the **/etc/multipath.conf** file before starting the **multipathd** daemon. use the following procedure to set up DM Multipath for a basic failover configuration.

1. Enter the **mpathconf** command with the **--enable** option specified:

   ```
   # mpathconf --enable
   ```

   For information on additional options to the **mpathconf** command you may require, see the **mpathconf(8)** man page or enter the **mpathconf** command with the **--help** option specified.

   ```
   # mpathconf --help
   usage: /sbin/mpathconf <command>

   Commands:
   Enable: --enable
   Disable: --disable
   Set user_friendly_names (Default y): --user_friendly_names <y|n>
   Set find_multipaths (Default y): --find_multipaths <y|n>
   Load the dm-multipath modules on enable (Default y): --with_module  <y|n>
   start/stop/reload multipathd (Default n): --with_multipathd  <y|n>
   ```

2. Edit the **/etc/multipath.conf** file if necessary. The default settings for DM Multipath are compiled in to the system and do not need to be explicitly set in the **/etc/multipath.conf** file. The default value of **path_grouping_policy** is set to **failover**, so in this example you do not need to edit the **/etc/multipath.conf** file.

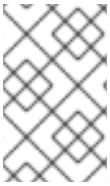
   The initial defaults section of the configuration file configures your system so that the names of the multipath devices are of the form **/dev/mapper/mpath***n*; without this setting, the names of the multipath devices would be aliased to the WWID of the device. If you do not want to use user friendly names, you can enter the following command:

```
# mpathconf --enable --user_friendly_names n
```

3. Save the configuration file and exit the editor, if necessary.

4. Execute the following command:

```
# systemctl start multipathd.service
```



NOTE

If you find that you need to edit the multipath configuration file after you have started the multipath daemon, you must execute the **systemctl reload multipathd.service** command for the changes to take effect.

3.2. IGNORING LOCAL DISKS WHEN GENERATING MULTIPATH DEVICES

Some machines have local SCSI cards for their internal disks. DM Multipath is not recommended for these devices. If you set the **find_multipaths** configuration parameter to **on**, you should not have to blacklist these devices. For information on the **find_multipaths** configuration parameter and the meaning of the values to which you can set this parameter to, see the **multipath.conf(5)** man page.

If you do not set the **find_multipaths** configuration parameter to **on**, you can use the following procedure to modify the multipath configuration file to ignore the local disks when configuring multipath.

1. Determine which disks are the internal disks and mark them as the ones to blacklist.
   In this example, **/dev/sda** is the internal disk. Note that as originally configured in the default multipath configuration file, executing the **multipath -v2** command shows the local disk, **/dev/sda**, in the multipath map.

   This examples specifies the **-d** option of the **multipath** command to indicate that this is a dry run that will not create the multipath devices.

```
# multipath -v2 -d
: SIBM-ESXSST336732LC____F3ET0EP0Q000072428BX1 undef WINSYS,SF2372
size=33 GB features="0" hwhandler="0" wp=undef
`-- policy='round-robin 0' prio=1 status=undef
  |- 0:0:0:0 sda 8:0  [---------

: 3600a0b80001327d80000006d43621677 undef WINSYS,SF2372
size=12G features='0' hwhandler='0' wp=undef
`-- policy='round-robin 0' prio=1 status=undef
  |- 2:0:0:0 sdb 8:16  undef ready  running
  `- 3:0:0:0 sdf 8:80 undef ready  running

: 3600a0b80001327510000009a436215ec undef WINSYS,SF2372
size=12G features='0' hwhandler='0' wp=undef
`-- policy='round-robin 0' prio=1 status=undef
  |- 2:0:0:1 sdc 8:32 undef ready  running
  `- 3:0:0:1 sdg 8:96 undef ready  running

: 3600a0b80001327d800000070436216b3 undef WINSYS,SF2372
size=12G features='0' hwhandler='0' wp=undef
```

```
`-- policy='round-robin 0' prio=1 status=undef
  |- 2:0:0:2 sdd 8:48 undef ready  running
  `- 3:0:0:2 sdg 8:112 undef ready  running

: 3600a0b80001327510000009b4362163e undef WINSYS,SF2372
size=12G features='0' hwhandler='0' wp=undef
`-+- policy='round-robin 0' prio=1 status=undef
  |- 2:0:0:3 sdd 8:64 undef ready  running
  `- 3:0:0:3 sdg 8:128 undef ready  running
```

2. In order to prevent the device mapper from mapping **/dev/sda** in its multipath maps, edit the blacklist section of the **/etc/multipath.conf** file to include this device. Although you could blacklist the **sda** device using a **devnode** type, that would not be a safe procedure since **/dev/sda** is not guaranteed to be the same on reboot. To blacklist individual devices, you can blacklist using the WWID of that device.
   Note that in the output to the **multipath -v2** command, the WWID of the **/dev/sda** device is SIBM-ESXSST336732LC____F3ET0EP0Q000072428BX1. To blacklist this device, include the following in the **/etc/multipath.conf** file.

```
blacklist {
      wwid SIBM-ESXSST336732LC____F3ET0EP0Q000072428BX1
}
```

3. After you have updated the **/etc/multipath.conf** file, you must manually tell the **multipathd** daemon to reload the file. The following command reloads the updated **/etc/multipath.conf** file.

```
# systemctl reload multipathd.service
```

3.3. CONFIGURING ADDITIONAL STORAGE DEVICES

By default, DM Multipath includes support for the most common storage arrays that themselves support DM Multipath. For information on the default configuration value, including supported devices, run either of the following commands.

```
# multipathd show config
# multipath -t
```

If you need to add a storage device that is not supported by default as a known multipath device, edit the **/etc/multipath.conf** file and insert the appropriate device information.

For example, to add information about the HP Open-V series the entry looks like this. This example sets the device to queue for a minute (or 12 retries and 5 seconds per retry) after all paths have failed.

```
devices {
        device {
                vendor "HP"
                product "OPEN-V"
                no_path_retry 12
        }
}
```

3.4. SETTING UP MULTIPATHING IN THE INITRAMFS FILE SYSTEM

You can set up multipathing in the **initramfs** file system. After configuring multipath, you can rebuild the **initramfs** file system with the multipath configuration files by executing the **dracut** command with the following options:

```
# dracut --force --add multipath
```

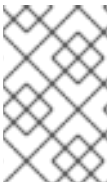
If you run multipath from the **initramfs** file system and you make any changes to the multipath configuration files, you must rebuild the **initramfs** file system for the changes to take effect.

CHAPTER 4. MODIFYING THE DM-MULTIPATH CONFIGURATION FILE

By default, DM Multipath provides configuration values for the most common uses of multipathing. In addition, DM Multipath includes support for the most common storage arrays that themselves support DM Multipath. For information on the default configuration values, including supported devices, run either of the following commands.

```
# multipathd show config
# multipath -t
```

You can override the default configuration values for DM Multipath by editing the **/etc/multipath.conf** configuration file. If necessary, you can also add a storage array that is not supported by default to the configuration file.



NOTE

You can run set up multipathing in the **initramfs** file system. If you run multipath from the **initramfs** file system and you make any changes to the multipath configuration files, you must rebuild the **initramfs** file system for the changes to take effect.

In the multipath configuration file, you need to specify only the sections that you need for your configuration, or that you wish to change from the default values. If there are sections of the file that are not relevant to your environment or for which you do not need to override the default values, you can leave them commented out, as they are in the initial file.

The configuration file allows regular expression description syntax.

Further information about the configuration file can be found on the **multipath.conf**(5) man page.

4.1. CONFIGURATION FILE OVERVIEW

The multipath configuration file is divided into the following sections:

blacklist

Listing of specific devices that will not be considered for multipath.

blacklist_exceptions

Listing of multipath candidates that would otherwise be blacklisted according to the parameters of the blacklist section.

defaults

General default settings for DM Multipath.

multipaths

Settings for the characteristics of individual multipath devices. These values overwrite what is specified in the **overrides**, **devices**, and **defaults** sections of the configuration file.

devices

Settings for the individual storage controllers. These values overwrite what is specified in the **defaults** section of the configuration file. If you are using a storage array that is not supported by default, you may need to create a **devices** subsection for your array.

overrides

Settings that are applied to all devices. These values overwrite what is specified in the **devices** and **defaults** sections of the configuration file.

When the system determines the attributes of a multipath device, first it checks the multipath settings, then the devices settings, then the multipath system defaults.

4.2. DM MULTIPATH OVERRIDES OF THE DEVICE TIMEOUT

The **recovery_tmo sysfs** option controls the timeout for a particular iSCSI device. The following options globally override **recovery_tmo** values:

- The **replacement_timeout** configuration option globally overrides the **recovery_tmo** value for all iSCSI devices.
- For all iSCSI devices that are managed by DM Multipath, the **fast_io_fail_tmo** option in DM Multipath globally overrides the **recovery_tmo** value.
  The **fast_io_fail_tmo** option in DM Multipath also overrides the **fast_io_fail_tmo** option in Fibre Channel devices.

The DM Multipath **fast_io_fail_tmo** option takes precedence over **replacement_timeout**. Red Hat does not recommend using **replacement_timeout** to override **recovery_tmo** in devices managed by DM Multipath because DM Multipath always resets **recovery_tmo** when the **multipathd** service reloads.

4.3. BLACKLISTING DEVICES FROM DM MULTIPATH

The **blacklist** section of the multipath configuration file specifies the devices that will not be used when the system configures multipath devices. Devices that are blacklisted will not be grouped into a multipath device.

If the **find_multipaths** configuration parameter is set to **off**, multipath always tries to create a multipath device for every path that is not explicitly blacklisted. If the **find_multipaths** configuration parameter is set to **on**, then multipath will create a device only if one of three conditions are met:

- There are at least two paths that are not blacklisted with the same WWID.
- The user manually forces the creation of the device by specifying a device with the **multipath** command.
- A path has the same WWID as a multipath device that was previously created (even if that multipath device does not currently exist). Whenever a multipath device is created, multipath remembers the WWID of the device so that it will automatically create the device again as soon as it sees a path with that WWID. This allows you to have multipath automatically choose the correct paths to make into multipath devices, without have to edit the multipath blacklist.
  If you have previously created a multipath device without using the **find_multipaths** parameter and then you later set the parameter to **on**, you may need to remove the WWIDs of any device you do not want created as a multipath device from the **/etc/multipath/wwids** file. The following shows a sample **/etc/multipath/wwids** file. The WWIDs are enclosed by slashes (/):

```
# Multipath wwids, Version : 1.0
# NOTE: This file is automatically maintained by multipath and multipathd.
# You should not need to edit this file in normal circumstances.
#
# Valid WWIDs:
/3600d0230000000000e13955cc3757802/
/3600d0230000000000e13955cc3757801/
```

```
/3600d0230000000000e13955cc3757800/
/3600d02300069c9ce09d41c31f29d4c00/
/SWINSYS  SF2372        0E13955CC3757802/
/3600d0230000000000e13955cc3757803/
```

In addition to **on** and **off**, you can also set **find_multipaths** to the following values:

- **strict**: multipath never accepts paths that have not previously been multipathed and are therefore not in the **/etc/multipath/wwids** file.
- **smart**: multipath always accepts non-blacklisted devices in udev as soon as they appear but if **multipathd** does not create the device within a timeout set with the **find_multipaths_timeout** parameter, it will release its claim on the device. For information on the **find_multipaths_timeout** parameter, see the **multipath.conf**(5) man page.

The built-in default value of **find_multipaths** is **off**. The default **multipath.conf** file created by **mpathconf**, however, will set the value of **find_multipaths** to **on**.

For more information on the values you can set for **find_multipaths**, see the **multipath.conf**(5) man page.

With the **find_multipaths** parameter set to **on**, you need to blacklist only the devices with multiple paths that you do not want to be multipathed. Because of this, it will generally not be necessary to blacklist devices.

If you do need to blacklist devices, you can blacklist devices by WWID, device name, device type, property, and protocol. For every device, these five blacklist criteria are evaluated in the the order "property, devnode, device, protocol, wwid". If a device turns out to be blacklisted by any criterion, it is excluded from handling by **multipathd**, and the later criteria are not evaluated. For each criterion, the whitelist takes precedence over the blacklist if a device matches both.

By default, a variety of device types are blacklisted, even after you comment out the initial blacklist section of the configuration file. For information, see Section 4.3.2, “Blacklisting by device name” .

4.3.1. Blacklisting by WWID

You can specify individual devices to blacklist by their World-Wide IDentification with a **wwid** entry in the **blacklist** section of the configuration file.

The following example shows the lines in the configuration file that would blacklist a device with a WWID of 26353900f02796769.

```
blacklist {
        wwid 26353900f02796769
}
```

4.3.2. Blacklisting by device name

You can blacklist device types by device name so that they will not be grouped into a multipath device by specifying a **devnode** entry in the **blacklist** section of the configuration file.

The following example shows the lines in the configuration file that would blacklist all SCSI devices, since it blacklists all sd* devices.

```
blacklist {
        devnode "^sd[a-z]"
}
```

You can use a **devnode** entry in the **blacklist** section of the configuration file to specify individual devices to blacklist rather than all devices of a specific type. This is not recommended, however, since unless it is statically mapped by **udev** rules, there is no guarantee that a specific device will have the same name on reboot. For example, a device name could change from **/dev/sda** to **/dev/sdb** on reboot.

By default, the following **devnode** entries are compiled in the default blacklist; the devices that these entries blacklist do not generally support DM Multipath. To enable multipathing on any of these devices, you would need to specify them in the **blacklist_exceptions** section of the configuration file, as described in Section 4.3.6, "Blacklist exceptions".

```
blacklist {
        devnode "^(ram|raw|loop|fd|md|dm-|sr|scd|st)[0-9]*"
        devnode "^(td|ha)d[a-z]"
}
```

4.3.3. Blacklisting by device type

You can specify specific device types in the **blacklist** section of the configuration file with a **device** section. The following example blacklists all IBM DS4200 and HP devices.

```
blacklist {
        device {
                vendor  "IBM"
                product "3S42"       #DS4200 Product 10
        }
        device {
                vendor  "HP"
                product ".*"
        }
}
```

4.3.4. Blacklisting by udev property

The **blacklist** and **blacklist_exceptions** sections of the **multipath.conf** configuration file support the **property** parameter. This parameter allows users to blacklist certain types of devices. The **property** parameter takes a regular expression string that is matched against the **udev** environment variable name for the device.

The following example blacklists all devices with the udev property **ID_ATA**.

```
blacklist {
        property "ID_ATA"
}
```

4.3.5. Blacklisting by device protocol

You can specify the protocol for a device to be excluded from multipathing in the **blacklist** section of the configuration file with a **protocol** section. The protocol strings that multipath recognizes are scsi:fcp, scsi:spi, scsi:ssa, scsi:sbp, scsi:srp, scsi:iscsi, scsi:sas, scsi:adt, scsi:ata, scsi:unspec, ccw, cciss, nvme, and

undef. The protocol that a path is using can be viewed by running the command **multipathd show paths format "%d %P"**.

The following example blacklists all devices with an undefined protocol or an unknown SCSI transport type.

```
blacklist {
        protocol "scsi:unspec"
        protocol "undef"
}
```

4.3.6. Blacklist exceptions

You can use the **blacklist_exceptions** section of the configuration file to enable multipathing on devices that have been blacklisted by default.

For example, if you have a large number of devices and want to multipath only one of them (with the WWID of 3600d0230000000000e13955cc3757803), instead of individually blacklisting each of the devices except the one you want, you could instead blacklist all of them, and then allow only the one you want by adding the following lines to the **/etc/multipath.conf** file.

```
blacklist {
        wwid ".*"
}

blacklist_exceptions {
        wwid "3600d0230000000000e13955cc3757803"
}
```

When specifying devices in the **blacklist_exceptions** section of the configuration file, you must specify the exceptions in the same way they were specified in the blacklist. For example, a WWID exception will not apply to devices specified by a **devnode** blacklist entry, even if the blacklisted device is associated with that WWID. Similarly, **devnode** exceptions apply only to **devnode** entries, and **device** exceptions apply only to device entries.

The **property** parameter works differently than the other **blacklist_exception** parameters. If the parameter is set, the device must have a **udev** variable that matches. Otherwise, the device is blacklisted. This parameter allows users to blacklist SCSI devices that multipath should ignore, such as USB sticks and local hard drives. To allow only SCSI devices that could reasonably be multipathed, set this parameter to **SCSI_IDENT_|ID_WWN)** as in the following example.

```
blacklist_exceptions {
        property "(SCSI_IDENT_|ID_WWN)"
}
```

4.4. MODIFYING MULTIPATH CONFIGURATION FILE DEFAULTS

The **/etc/multipath.conf** configuration file includes a **defaults** section that sets the **user_friendly_names** parameter to **yes**, as follows.

```
defaults {
        user_friendly_names yes
}
```

This overwrites the default value of the **user_friendly_names** parameter.

The configuration file includes a template of configuration defaults. This section is commented out, as follows.

```
#defaults {
#       polling_interval        10
#       path_selector           "round-robin 0"
#       path_grouping_policy    multibus
#       uid_attribute           ID_SERIAL
#       prio                    alua
#       path_checker            readsector0
#       rr_min_io               100
#       max_fds                 8192
#       rr_weight               priorities
#       failback                immediate
#       no_path_retry           fail
#       user_friendly_names     yes
#}
```

To overwrite the default value for any of the configuration parameters, you can copy the relevant line from this template into the **defaults** section and uncomment it. For example, to overwrite the **path_grouping_policy** parameter so that it is **multibus** rather than the default value of **failover**, copy the appropriate line from the template to the initial **defaults** section of the configuration file, and uncomment it, as follows.

```
defaults {
        user_friendly_names     yes
        path_grouping_policy    multibus
}
```

For information on the attributes that are set in the **defaults** section of the **multipath.conf** configuration file see the **multipath.conf**(5) man page. These values are used by DM Multipath unless they are overwritten by the attributes specified in the **devices**, **multipaths**, or **overrides** sections of the **multipath.conf** file.

4.5. MODIFYING MULTIPATH SETTINGS FOR SPECIFIC DEVICES

The attributes in the **multipaths** section of the **multipath.conf** configuration file apply only to the one specified multipath. These defaults are used by DM Multipath and override attributes set in the **overrides**, **defaults**, and **devices** sections of the **multipath.conf** file.

For information on the attributes that are set in the **multipaths** section of the **multipath.conf** configuration file see the **multipath.conf**(5) man page.

The following example shows multipath attributes specified in the configuration file for two specific multipath devices. The first device has a WWID of **3600508b4000156d70001200000b0000** and a symbolic name of **yellow**.

The second multipath device in the example has a WWID of **1DEC_321816758474** and a symbolic name of **red**. In this example, the **rr_weight** attributes is set to **priorities**.

```
multipaths {
        multipath {
                wwid                    3600508b4000156d70001200000b0000
```

```
        alias                yellow
        path_grouping_policy  multibus
        path_selector         "round-robin 0"
        failback              manual
        rr_weight             priorities
        no_path_retry         5
    }
    multipath {
        wwid                  1DEC_321816758474
        alias                 red
        rr_weight             priorities
     }
}
```

4.6. MODIFYING MULTIPATH SETTINGS FOR STORAGE CONTROLLERS

The **devices** section of the **multipath.conf** configuration file sets attributes for individual storage devices. These attributes are used by DM Multipath unless they are overwritten by the attributes specified in the **multipaths** or **overrides** sections of the **multipath.conf** file for paths that contain the device. These attributes override the attributes set in the **defaults** section of the **multipath.conf** file.

For information on the attributes that are set in the **devices** section of the **multipath.conf** configuration file see the **multipath.conf**(5) man page.

Many devices that support multipathing are included by default in a multipath configuration. For information on the default configuration value, including supported devices, run either of the following commands.

```
# multipathd show config
# multipath -t
```

You probably will not need to modify the values for these devices, but if you do you can overwrite the default values by including an entry in the configuration file for the device that overwrites those values. You can copy the device configuration defaults for the device that the **multipathd show config** command displays and override the values that you want to change.

To add a device that is not configured automatically by default to this section of the configuration file, you need to set the **vendor** and **product** parameters. You can find these values by looking at **/sys/block/*device_name*/device/vendor** and **/sys/block/*device_name*/device/model** where *device_name* is the device to be multipathed, as in the following example:

```
# cat /sys/block/sda/device/vendor
WINSYS
# cat /sys/block/sda/device/model
SF2372
```

The additional parameters to specify depend on your specific device. If the device is active/active, you will usually not need to set additional parameters. You may want to set **path_grouping_policy** to **multibus**. Other parameters you may need to set are **no_path_retry** and **rr_min_io**.

If the device is active/passive, but it automatically switches paths with I/O to the passive path, you need to change the checker function to one that does not send I/O to the path to test if it is working (otherwise, your device will keep failing over). This almost always means that you set the **path_checker**

to **tur**; this works for all SCSI devices that support the Test Unit Ready command, which most do.

If the device needs a special command to switch paths, then configuring this device for multipath requires a hardware handler kernel module. The current available hardware handler is **emc**. If this is not sufficient for your device, you may not be able to configure the device for multipath.

The following example shows a **device** entry in the multipath configuration file.

```
# }
# device {
#  vendor   "COMPAQ  "
#  product   "MSA1000         "
#  path_grouping_policy multibus
#  path_checker  tur
#  rr_weight  priorities
# }
#}
```

4.7. SETTING MULTIPATH VALUES FOR ALL DEVICES

The **overrides** section of the **multipath.conf** configuration file allows you to set a configuration value for all of your devices. For example, you may want all devices to set **no_path_retry** to **fail**. This section supports all of the attributes that are supported by both the **devices** and **defaults** section of the **multipath.conf** configuration file, which is all of the **devices** section attributes except **vendor**, **product** and **revision**. These attributes are used by DM Multipath for all devices unless they are overwritten by the attributes specified in the **multipaths** section of the **multipath.conf** file for paths that contain the device. These attributes override the attributes set in the **devices** and **defaults** sections of the **multipath.conf** file.

For information on the attributes that are set in the **devices** and **defaults** sections of the **multipath.conf** configuration file see the **multipath.conf**(5) man page.

CHAPTER 5. MANAGING MULTIPATHED VOLUMES

DM-Multipath provides a variety of tools and commands you can use to manage multipath volumes.

5.1. THE MULTIPATH COMMAND

The **multipath** command is used to detect and coalesce multiple paths to devices. It provides a variety of option you can use to administer your multipathed devices.

Table 5.1, "Useful **multipath** Command Options" describes some options of the **multipath** command that you may find useful.

Table 5.1. Useful multipath Command Options

Option	Description
-l	Display the current multipath configuration gathered from **sysfs** and the device mapper.
-ll	Display the current multipath configuration gathered from **sysfs**, the device mapper, and all other available components on the system.
-f *device*	Remove the named multipath device.
-F	Remove all unused multipath devices.
-w *device*	Remove the **wwid** of the specified device from the **wwids** file.
-W	Reset the **wwids** file to include only the current multipath devices.

5.1.1. Multipath command output

When you create, modify, or list a multipath device, you get a display of the current device setup. The format is as follows.

For each multipath device:

```
action_if_any: alias (wwid_if_different_from_alias) dm_device_name_if_known
vendor,product size=size features='features' hwhandler='hardware_handler'
wp=write_permission_if_known
```

For each path group:

```
-+- policy='scheduling_policy' prio=prio_if_known status=path_group_status_if_known
```

For each path:

```
  `- host:channel:id:lun devnode major:minor dm_status_if_known path_status online_status
```

For example, the output of a multipath command might appear as follows:

```
3600d0230000000000e13955cc3757800 dm-1 WINSYS,SF2372
size=269G features='0' hwhandler='0' wp=rw
|-+- policy='round-robin 0' prio=1 status=active
| `- 6:0:0:0 sdb 8:16  active ready  running
`-+- policy='round-robin 0' prio=1 status=enabled
  `- 7:0:0:0 sdf 8:80  active ready  running
```

If the path is up and ready for I/O, the status of the path is **ready** or **ghost**. If the path is down, the status is **faulty** or **shaky**. The path status is updated periodically by the **multipathd** daemon based on the polling interval defined in the **/etc/multipath.conf** file.

Additional possible path status values are as follows.

- **i/o pending**: The checker is actively checking this path, and the state will be updated shortly.
- **i/o timeout**: This is the same as **faulty**. It lets the user know that the checker did not return either success or failure before the timeout period.
- **removed**: The path has been removed from the system, and will shortly be removed from the multipath device. It is treated the same as **faulty**.
- **wild**: **multipathd** was unable to run the path checker, because of an internal error or configuration issue. This is roughly the same as **faulty**, except multipath will skip many actions on the path.
- **unchecked**: The path checker has not run on this path, either because it has just been discovered, it does not have an assigned path checker, or the path checker encountered an error. This is treated the same as **wild**.
- **delayed**: The path checker returned that the path is up, but multipath is delaying the reinstatement of the path because the path has recently failed multiple times and multipath has been configured to delay paths in this case.

The dm status is similar to the path status, but from the kernel's point of view. The **active** dm state covers the **ready** and **ghost** path states. The **pending** path state has no equivalent dm state. All other path states map to the **failed** dm state. The dm state will retain its current status until the path checker has completed.

The possible values for **online_status** are **running** and **offline**. A status of **offline** means that this SCSI device has been disabled.

NOTE

When a multipath device is being created or modified, the path group status, the dm device name, the write permissions, and the dm status are not known. Also, the features are not always correct.

5.1.2. Displaying multipath configuration

You can use the **-l** and **-ll** options of the **multipath** command to display the current multipath configuration. The **-l** option displays multipath topology gathered from information in **sysfs** and the device mapper. The **-ll** option displays the information the **-l** option displays in addition to all other available components of the system.

When displaying the multipath configuration, you can specify a verbosity level with the **-v** option of the **multipath** command. Specifying **-v0** yields no output. Specifying **-v1** outputs the created or updated multipath names only, which you can then feed to other tools such as **kpartx**. Specifying **-v2** prints all detected paths, multipaths, and device maps. For even more detailed information, you can also specify **-v3, -v4**, or **-v5**.

The following example shows the output of a **multipath -l** command.

```
# multipath -l
3600d0230000000000e13955cc3757800 dm-1 WINSYS,SF2372
size=269G features='0' hwhandler='0' wp=rw
|-- policy='round-robin 0' prio=1 status=active
| `- 6:0:0:0 sdb 8:16  active ready  running
`-- policy='round-robin 0' prio=1 status=enabled
  `- 7:0:0:0 sdf 8:80  active ready  running
```

The following example shows the output of a **multipath -ll** command.

```
# multipath -ll
3600d0230000000000e13955cc3757801 dm-10 WINSYS,SF2372
size=269G features='0' hwhandler='0' wp=rw
|-- policy='round-robin 0' prio=1 status=enabled
| `- 19:0:0:1 sdc 8:32  active ready  running
`-- policy='round-robin 0' prio=1 status=enabled
  `- 18:0:0:1 sdh 8:112 active ready  running
3600d0230000000000e13955cc3757803 dm-2 WINSYS,SF2372
size=125G features='0' hwhandler='0' wp=rw
`-+- policy='round-robin 0' prio=1 status=active
  |- 19:0:0:3 sde 8:64  active ready  running
  `- 18:0:0:3 sdj 8:144 active ready  running
```

5.2. RESIZING AN ONLINE MULTIPATH DEVICE

If you need to resize an online multipath device, use the following procedure.

1. Resize your physical device.
2. Execute the following command to find the paths to the LUN:

   ```
   # multipath -l
   ```

3. Resize your paths. For SCSI devices, writing a 1 to the **rescan** file for the device causes the SCSI driver to rescan, as in the following command:

   ```
   # echo 1 > /sys/block/path_device/device/rescan
   ```

   Ensure that you run this command for each of the path devices. For example, if your path devices are **sda**, **sdb**, **sde**, and **sdf**, you would run the following commands:

   ```
   # echo 1 > /sys/block/sda/device/rescan
   # echo 1 > /sys/block/sdb/device/rescan
   # echo 1 > /sys/block/sde/device/rescan
   # echo 1 > /sys/block/sdf/device/rescan
   ```

4. Resize your multipath device by executing the **multipathd resize** command:

```
# multipathd resize map multipath_device
```

5. Resize the file system (assuming no LVM or DOS partitions are used):

```
# resize2fs /dev/mapper/mpatha
```

5.3. MOVING A ROOT FILE SYSTEM FROM A SINGLE PATH DEVICE TO A MULTIPATH DEVICE

If you have installed your system on a single-path device and later add another path to the root file system, you will need to move your root file system to a multipathed device. This section documents the procedure for moving from a single-path to a multipathed device.

After ensuring that you have installed the **device-mapper-multipath** package, perform the following procedure:

1. Execute the following command to create the **/etc/multipath.conf** configuration file, load the multipath module, and set **chkconfig** for the **multipathd** to **on**:

```
# mpathconf --enable
```

2. If the **find_multipaths** configuration parameter is not set to **yes**, edit the **blacklist** and **blacklist_exceptions** sections of the **/etc/multipath.conf** file, as described in Section 4.3, "Blacklisting devices from DM Multipath".

3. In order for multipath to build a multipath device on top of the root device as soon as it is discovered, enter the following command. This command also ensures that **find_multipaths** will allow the device, even if it only has one path.

```
# multipath -a root_devname
```

   For example, if the root device is **/dev/sdb**, enter the following command.

```
# multipath -a /dev/sdb
wwid '3600d02300069c9ce09d41c4ac9c53200' added
```

4. To confirm that your configuration file is set up correctly, you can enter the **multipath** command and search the output for a line of the following format. This indicates that the command failed to create the multipath device.

```
date  wwid: ignoring map
```

   For example, if the WWID if the device is 3600d02300069c9ce09d41c4ac9c53200, you would see a line in the output such as the following:

```
# multipath
Oct 21 09:37:19 | 3600d02300069c9ce09d41c4ac9c53200: ignoring map
```

5. To rebuild the **initramfs** file system with **multipath**, execute the **dracut** command with the following options:

```
# dracut --force -H --add multipath
```

6. Shut the machine down.
7. Configure the FC switch so that other paths are visible to the machine.
8. Boot the machine.
9. Check whether the root file system ('/') is on the multipathed device.

5.4. MOVING A SWAP FILE SYSTEM FROM A SINGLE PATH DEVICE TO A MULTIPATH DEVICE

By default, swap devices are set up as logical volumes. This does not require any special procedure for configuring them as multipath devices as long as you set up multipathing on the physical volumes that constitute the logical volume group. If your swap device is not an LVM volume, however, and it is mounted by device name, you may need to edit the **/etc/fstab** file to switch to the appropriate multipath device name.

1. Determine the WWID number of the swap device by running the **/sbin/multipath** command with the **-v3** option. The output from the command should show the swap device in the paths list. You should look in the command output for a line of the following format, showing the swap device:

   ```
   WWID  H:B:T:L devname MAJOR:MINOR
   ```

   For example, if your swap file system is set up on **sda** or one of its partitions, you would see a line in the output such as the following:

   ```
   ===== paths list =====
   ...
   1ATA     WDC WD800JD-75MSA3                      WD-WMAM9F 1:0:0:0 sda 8:0
   ...
   ```

2. Set up an alias for the swap device in the **/etc/multipath.conf** file:

   ```
   multipaths {
       multipath {
           wwid WWID_of_swap_device
           alias swapdev
       }
   }
   ```

3. Edit the **/etc/fstab** file and replace the old device path to the root device with the multipath device.
   For example, if you had the following entry in the **/etc/fstab** file:

   ```
   /dev/sda2 swap                 swap    defaults        0 0
   ```

   You would change the entry to the following:

   ```
   /dev/mapper/swapdev swap          swap    defaults        0 0
   ```

5.5. DETERMINING DEVICE MAPPER ENTRIES WITH THE DMSETUP COMMAND

You can use the **dmsetup** command to find out which device mapper entries match the multipathed devices.

The following command displays all the device mapper devices and their major and minor numbers. The minor numbers determine the name of the dm device. For example, a minor number of 3 corresponds to the multipathed device /**dev**/**dm-3**.

```
# dmsetup ls
mpathd  (253:4)
mpathep1        (253:12)
mpathfp1        (253:11)
mpathb  (253:3)
mpathgp1        (253:14)
mpathhp1        (253:13)
mpatha  (253:2)
mpathh  (253:9)
mpathg  (253:8)
VolGroup00-LogVol01     (253:1)
mpathf  (253:7)
VolGroup00-LogVol00     (253:0)
mpathe  (253:6)
mpathbp1        (253:10)
mpathd  (253:5)
```

5.6. ADMINISTERING THE MULTIPATHD DAEMON

The **multipathd** commands can be used to administer the **multipathd** daemon. For information on the available **multipathd** commands, see the **multipathd**(8) man page.

The following command shows the standard default format for the output of the **multipathd show maps** command.

```
# multipathd show maps
name sysfs uuid
mpathc dm-0 360a98000324669436c2b45666c567942
```

Some **multipathd** commands include a **format** option followed by a wildcard. You can display a list of available wildcards with the following command.

```
# multipathd show wildcards
```

The **multipathd** command supports format commands that show the status of multipath devices and paths in "raw" format versions. In raw format, no headers are printed and the fields are not padded to align the columns with the headers. Instead, the fields print exactly as specified in the format string. This output can then be more easily used for scripting. You can display the wildcards used in the format string with the **multipathd show wildcards** command.

The following **multipathd** commands show the multipath devices that **multipathd** is monitoring, using a format string with multipath wildcards, in regular and raw format.

```
list|show maps|multipaths format $format
list|show maps|multipaths raw format $format
```

The following **multipathd** commands show the paths that **multipathd** is monitoring, using a format string with multipath wildcards, in regular and raw format.

```
list|show paths format $format
list|show paths raw format $format
```

The following commands show the difference between the non-raw and raw formats for the **multipathd show maps**. Note that in **raw** format there are no headers and only a single space between the columns.

```
# multipathd show maps format "%n %w %d %s"
name   uuid                             sysfs vend/prod/rev
mpathc 360a98000324669436c2b45666c567942 dm-0  NETAPP,LUN

# multipathd show maps raw format "%n %w %d %s"
mpathc 360a98000324669436c2b45666c567942 dm-0 NETAPP,LUN
```

5.7. CLEANING UP MULTIPATH FILES ON PACKAGE REMOVAL

If you should have occasion to remove the **device-mapper-multipath rpm**. file, note that this does not remove the **/etc/multipath.conf**, **/etc/multipath/bindings**, and **/etc/multipath/wwids** files. You may need to remove those files manually on subsequent installations of the **device-mapper-multipath** package.

CHAPTER 6. TROUBLESHOOTING DM MULTIPATH

If you have trouble implementing a multipath configuration, there are a variety of issues you can check for.

6.1. DM MULTIPATH TROUBLESHOOTING CHECKLIST

The following issues may result in a slow or non-functioning multipath configuration.

6.1.1. The multipath daemon is not running

If you find you have trouble implementing a multipath configuration, you should ensure that the multipath daemon is running, as described in Chapter 3, *Setting up DM Multipath*.

The **multipathd** daemon must be running in order to use multipathed devices.

6.1.2. Issues with queue_if_no_path feature

If a multipath device is configured with **features "1 queue_if_no_path"**, then any process that issues I/O will hang until one or more paths are restored. To avoid this, set the **no_path_retry N** parameter in the **/etc/multipath.conf** file (where **N** is the number of times the system should retry a path).

If you need to use the **features "1 queue_if_no_path"** option and you experience the issue noted here, you can disable queueing policy at runtime for a particular LUN (that is, for which all the paths are unavailable). The following command disables queueing for a specific device.

```
multipathd disablequeueing map device
```

The following command disables queueing for all devices.

```
multipathd disablequeueing maps
```

After you have disabled queueing for a device, it will remain disabled until **multipathd** is restarted or reloaded or until you one of the following commands.

The following command resets queueing to its previous value for a specific device.

```
multipathd restorequeueing map device
```

The following command resets queueing to its previous value for all devices.

```
multipathd restorequeueing maps
```

6.2. TROUBLESHOOTING WITH THE MULTIPATHD INTERACTIVE CONSOLE

The **multipathd -k** command is an interactive interface to the **multipathd** daemon. Entering this command brings up an interactive multipath console. After executing this command, you can enter **help** to get a list of available commands, you can enter a interactive command, or you can enter **CTRL-D** to quit.

The **multipathd** interactive console can be used to troubleshoot problems you may be having with your system. For example, the following command sequence displays the multipath configuration, including the defaults, before exiting the console.

```
# multipathd -k
> > show config
> > CTRL-D
```

The following command sequence ensures that multipath has picked up any changes to the **multipath.conf**,

```
# multipathd -k
> > reconfigure
> > CTRL-D
```

Use the following command sequence to ensure that the path checker is working properly.

```
# multipathd -k
> > show paths
> > CTRL-D
```

CHAPTER 7. CONFIGURING MAXIMUM TIME FOR STORAGE ERROR RECOVERY WITH EH_DEADLINE

You can configure the maximum allowed time to recover failed SCSI devices. This configuration guarantees an I/O response time even when storage hardware becomes unresponsive due to a failure.

7.1. THE EH_DEADLINE PARAMETER

The SCSI error handling (EH) mechanism attempts to perform error recovery on failed SCSI devices. The SCSI host object **eh_deadline** parameter enables you to configure the maximum amount of time for the recovery. After the configured time expires, SCSI EH stops and resets the entire host bus adapter (HBA).

Using **eh_deadline** can reduce the time:

- to shut off a failed path,
- to switch a path, or
- to disable a RAID slice.

WARNING

When **eh_deadline** expires, SCSI EH resets the HBA, which affects all target paths on that HBA, not only the failing one. If some of the redundant paths are not available for other reasons, I/O errors might occur. Enable **eh_deadline** only if you have a fully redundant multipath configuration on all targets.

Scenarios when eh_deadline is useful

In most scenarios, you do not need to enable **eh_deadline**. Using **eh_deadline** can be useful in certain specific scenarios, for example if a link loss occurs between a Fibre Channel (FC) switch and a target port, and the HBA does not receive Registered State Change Notifications (RSCNs). In such a case, I/O requests and error recovery commands all time out rather than encounter an error. Setting **eh_deadline** in this environment puts an upper limit on the recovery time. That enables the failed I/O to be retried on another available path by DM Multipath.

Under the following conditions, the **eh_deadline** functionality provides no additional benefit, because the I/O and error recovery commands fail immediately, which allows DM Multipath to retry:

- If RSCNs are enabled
- If the HBA does not register the link becoming unavailable

Possible values

The value of the **eh_deadline** is specified in seconds.

The default setting is **off**, which disables the time limit and allows all of the error recovery to take place.

7.2. SETTING THE EH_DEADLINE PARAMETER

This procedure configures the value of the **eh_deadline** parameter to limit the maximum SCSI recovery time.

Procedure

- You can configure **eh_deadline** using either of the following methods:

  sysfs

  Write the number of seconds into the **/sys/class/scsi_host/host*/eh_deadline** files.

  Kernel parameter

  Set a default value for all SCSI HBAs using the **scsi_mod.eh_deadline** kernel parameter.

Additional resources

- How to set eh_deadline and eh_timeout persistently, using a udev rule