Red Hat Enterprise Linux 6

Cluster Administration

Configuring and Managing the High Availability Add-On

Red Hat Enterprise Linux 6 Cluster Administration
Configuring and Managing the High Availability Add-On
Edition 0

1801 Varsity Drive
Raleigh, NC 27606-2072 USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701

Configuring and Managing the High Availability Add-On describes the configuration and management of the High Availability Add-On for Red Hat Enterprise Linux 6.

Introduction **v**
1. Document Conventions ... vi
1.1. Typographic Conventions ... vi
1.2. Pull-quote Conventions ... vii
1.3. Notes and Warnings ... viii
2. Feedback ... viii

1. Red Hat High Availability Add-On Configuration and Management Overview **1**
1.1. Configuration Basics ... 1
1.2. Setting Up Hardware ... 1
1.3. Installing Red Hat High Availability Add-On software ... 2
1.4. Configuring Red Hat High Availability Add-On Software ... 2

2. Before Configuring the Red Hat High Availability Add-On **5**
2.1. General Configuration Considerations ... 5
2.2. Compatible Hardware ... 6
2.3. Enabling IP Ports ... 6
2.3.1. Enabling IP Ports on Cluster Nodes ... 7
2.3.2. Enabling IP Ports on Computers That Run **luci** ... 7
2.4. Configuring ACPI For Use with Integrated Fence Devices ... 7
2.4.1. Disabling ACPI Soft-Off with **`chkconfig`** Management ... 9
2.4.2. Disabling ACPI Soft-Off with the BIOS ... 9
2.4.3. Disabling ACPI Completely in the **`grub.conf`** File ... 11
2.5. Considerations for Configuring HA Services ... 12
2.6. Configuration Validation ... 14
2.7. Considerations for **NetworkManager** ... 17
2.8. Considerations for Using Quorum Disk ... 17
2.9. Red Hat High Availability Add-On and SELinux ... 18
2.10. Multicast Addresses ... 18
2.11. Considerations for **`ricci`** ... 19
2.12. Considerations for Using **Conga** ... 19

3. Configuring Red Hat High Availability Add-On With Conga **21**
3.1. Configuration Tasks ... 21
3.2. Starting **luci** and **ricci** ... 22
3.3. Creating A Cluster ... 23
3.4. Global Cluster Properties ... 25
3.5. Configuring Fence Devices ... 29
3.5.1. Creating a Fence Device ... 30
3.5.2. Modifying a Fence Device ... 30
3.5.3. Deleting a Fence Device ... 30
3.6. Configuring Fencing for Cluster Members ... 31
3.6.1. Configuring a Single Fence Device for a Node ... 31
3.6.2. Configuring a Backup Fence Device ... 32
3.6.3. Configuring A Node with Redundant Power ... 32
3.7. Configuring a Failover Domain ... 34
3.7.1. Adding a Failover Domain ... 35
3.7.2. Modifying a Failover Domain ... 36
3.7.3. Deleting a Failover Domain ... 36
3.8. Configuring Global Cluster Resources ... 37
3.9. Adding a Cluster Service to the Cluster ... 37

4. Managing Red Hat High Availability Add-On With Conga **41**
4.1. Adding an Existing Cluster to the luci Interface ... 41
4.2. Managing Cluster Nodes ... 41
4.2.1. Rebooting a Cluster Node ... 42

4.2.2. Causing a Node to Leave or Join a Cluster 42
4.2.3. Adding a Member to a Running Cluster 42
4.2.4. Deleting a Member from a Cluster 43
4.3. Starting, Stopping, Restarting, and Deleting Clusters 44
4.4. Managing High-Availability Services 44
4.5. Diagnosing and Correcting Problems in a Cluster 45

5. Configuring Red Hat High Availability Add-On With Command Line Tools 47
5.1. Configuration Tasks 48
5.2. Creating a Basic Cluster Configuration File 48
5.3. Configuring Fencing 51
5.4. Configuring Failover Domains 56
5.5. Configuring HA Services 59
5.5.1. Adding Cluster Resources 59
5.5.2. Adding a Cluster Service to the Cluster 61
5.6. Verifying a Configuration 64

6. Managing Red Hat High Availability Add-On With Command Line Tools 67
6.1. Starting and Stopping the Cluster Software 67
6.1.1. Starting Cluster Software 68
6.1.2. Stopping Cluster Software 68
6.2. Deleting or Adding a Node 69
6.2.1. Deleting a Node from a Cluster 69
6.2.2. Adding a Node to a Cluster 72
6.2.3. Examples of Three-Node and Two-Node Configurations 74
6.3. Managing High-Availability Services 76
6.3.1. Displaying HA Service Status with **`clustat`** 77
6.3.2. Managing HA Services with **`clusvcadm`** 77
6.4. Updating a Configuration 80
6.4.1. Updating a Configuration Using **`cman_tool version -r`** 80
6.4.2. Updating a Configuration Using **`scp`** 82
6.5. Diagnosing and Correcting Problems in a Cluster 84

A. Fence Device Parameters 85

B. HA Resource Parameters 93

C. HA Resource Behavior 101
C.1. Parent, Child, and Sibling Relationships Among Resources 102
C.2. Sibling Start Ordering and Resource Child Ordering 102
C.2.1. Typed Child Resource Start and Stop Ordering 103
C.2.2. Non-typed Child Resource Start and Stop Ordering 105
C.3. Inheritance, the <resources> Block, and Reusing Resources 106
C.4. Failure Recovery and Independent Subtrees 107
C.5. Debugging and Testing Services and Resource Ordering 108

D. Command Line Tools Summary 111

E. Revision History 113

Index 115

Introduction

This document provides information about installing, configuring and managing Red Hat High Availability Add-On components. Red Hat High Availability Add-On components allow you to connect a group of computers (called *nodes* or *members*) to work together as a cluster. In this document, the use of the word *cluster* or *clusters* is used to refer to a group of computers running the Red Hat High Availability Add-On.

The audience of this document should have advanced working knowledge of Red Hat Enterprise Linux and understand the concepts of clusters, storage, and server computing.

This document is organized as follows:

- *Chapter 1, Red Hat High Availability Add-On Configuration and Management Overview*
- *Chapter 2, Before Configuring the Red Hat High Availability Add-On*
- *Chapter 3, Configuring Red Hat High Availability Add-On With* ***Conga***
- *Chapter 4, Managing Red Hat High Availability Add-On With* ***Conga***
- *Chapter 5, Configuring Red Hat High Availability Add-On With Command Line Tools*
- *Chapter 6, Managing Red Hat High Availability Add-On With Command Line Tools*
- *Appendix A, Fence Device Parameters*
- *Appendix B, HA Resource Parameters*
- *Appendix C, HA Resource Behavior*
- *Appendix D, Command Line Tools Summary*
- *Appendix E, Revision History*

For more information about Red Hat Enterprise Linux 6, refer to the following resources:

- *Red Hat Enterprise Linux Installation Guide* — Provides information regarding installation of Red Hat Enterprise Linux 6.
- *Red Hat Enterprise Linux Deployment Guide* — Provides information regarding the deployment, configuration and administration of Red Hat Enterprise Linux 6.

For more information about the High Availability Add-On and related products for Red Hat Enterprise Linux 6, refer to the following resources:

- *Red Hat Cluster Suite Overview* — Provides a high-level overview of the High Availability Add-On, Resilient Storage Add-On, and the Load Balancer Add-On.
- *Logical Volume Manager Administration* — Provides a description of the Logical Volume Manager (LVM), including information on running LVM in a clustered environment.
- *Global File System 2: Configuration and Administration* — Provides information about installing, configuring, and maintaining Red Hat GFS2 (Red Hat Global File System 2), which is included in the Resilient Storage Add-On.
- *DM Multipath* — Provides information about using the Device-Mapper Multipath feature of Red Hat Enterprise Linux 6.

- *Linux Virtual Server Administration* — Provides information on configuring high-performance systems and services with the Red Hat Load Balancer Add-On (Formerly known as Linux Virtual Server [LVS]).
- *Release Notes* — Provides information about the current release of Red Hat products.

High Availability Add-On documentation and other Red Hat documents are available in HTML, PDF, and RPM versions on the Red Hat Enterprise Linux Documentation CD and online at *http://docs.redhat.com/*.

1. Document Conventions

This manual uses several conventions to highlight certain words and phrases and draw attention to specific pieces of information.

In PDF and paper editions, this manual uses typefaces drawn from the *Liberation Fonts*[1] set. The Liberation Fonts set is also used in HTML editions if the set is installed on your system. If not, alternative but equivalent typefaces are displayed. Note: Red Hat Enterprise Linux 5 and later includes the Liberation Fonts set by default.

1.1. Typographic Conventions

Four typographic conventions are used to call attention to specific words and phrases. These conventions, and the circumstances they apply to, are as follows.

`Mono-spaced Bold`

Used to highlight system input, including shell commands, file names and paths. Also used to highlight keycaps and key combinations. For example:

> To see the contents of the file **`my_next_bestselling_novel`** in your current working directory, enter the **`cat my_next_bestselling_novel`** command at the shell prompt and press **`Enter`** to execute the command.

The above includes a file name, a shell command and a keycap, all presented in mono-spaced bold and all distinguishable thanks to context.

Key combinations can be distinguished from keycaps by the hyphen connecting each part of a key combination. For example:

> Press **`Enter`** to execute the command.
>
> Press **`Ctrl`**+**`Alt`**+**`F2`** to switch to the first virtual terminal. Press **`Ctrl`**+**`Alt`**+**`F1`** to return to your X-Windows session.

The first paragraph highlights the particular keycap to press. The second highlights two key combinations (each a set of three keycaps with each set pressed simultaneously).

If source code is discussed, class names, methods, functions, variable names and returned values mentioned within a paragraph will be presented as above, in **`mono-spaced bold`**. For example:

> File-related classes include **`filesystem`** for file systems, **`file`** for files, and **`dir`** for directories. Each class has its own associated set of permissions.

[1] https://fedorahosted.org/liberation-fonts/

Proportional Bold

This denotes words or phrases encountered on a system, including application names; dialog box text; labeled buttons; check-box and radio button labels; menu titles and sub-menu titles. For example:

> Choose **System** → **Preferences** → **Mouse** from the main menu bar to launch **Mouse Preferences**. In the **Buttons** tab, click the **Left-handed mouse** check box and click **Close** to switch the primary mouse button from the left to the right (making the mouse suitable for use in the left hand).
>
> To insert a special character into a **gedit** file, choose **Applications** → **Accessories** → **Character Map** from the main menu bar. Next, choose **Search** → **Find…** from the **Character Map** menu bar, type the name of the character in the **Search** field and click **Next**. The character you sought will be highlighted in the **Character Table**. Double-click this highlighted character to place it in the **Text to copy** field and then click the **Copy** button. Now switch back to your document and choose **Edit** → **Paste** from the **gedit** menu bar.

The above text includes application names; system-wide menu names and items; application-specific menu names; and buttons and text found within a GUI interface, all presented in proportional bold and all distinguishable by context.

Mono-spaced Bold Italic or ***Proportional Bold Italic***

Whether mono-spaced bold or proportional bold, the addition of italics indicates replaceable or variable text. Italics denotes text you do not input literally or displayed text that changes depending on circumstance. For example:

> To connect to a remote machine using ssh, type **ssh *username@domain.name*** at a shell prompt. If the remote machine is **example.com** and your username on that machine is john, type **ssh john@example.com**.
>
> The **mount -o remount *file-system*** command remounts the named file system. For example, to remount the **/home** file system, the command is **mount -o remount /home**.
>
> To see the version of a currently installed package, use the **rpm -q *package*** command. It will return a result as follows: ***package-version-release***.

Note the words in bold italics above — username, domain.name, file-system, package, version and release. Each word is a placeholder, either for text you enter when issuing a command or for text displayed by the system.

Aside from standard usage for presenting the title of a work, italics denotes the first use of a new and important term. For example:

> Publican is a *DocBook* publishing system.

1.2. Pull-quote Conventions

Terminal output and source code listings are set off visually from the surrounding text.

Output sent to a terminal is set in **mono-spaced roman** and presented thus:

```
books        Desktop   documentation  drafts  mss    photos   stuff  svn
```

```
books_tests  Desktop1  downloads      images  notes  scripts  svgs
```

Source-code listings are also set in **`mono-spaced roman`** but add syntax highlighting as follows:

```
package org.jboss.book.jca.ex1;

import javax.naming.InitialContext;

public class ExClient
{
   public static void main(String args[])
       throws Exception
   {
      InitialContext iniCtx = new InitialContext();
      Object         ref    = iniCtx.lookup("EchoBean");
      EchoHome       home   = (EchoHome) ref;
      Echo           echo   = home.create();

      System.out.println("Created Echo");

      System.out.println("Echo.echo('Hello') = " + echo.echo("Hello"));
   }
}
```

1.3. Notes and Warnings

Finally, we use three visual styles to draw attention to information that might otherwise be overlooked.

Note

Notes are tips, shortcuts or alternative approaches to the task at hand. Ignoring a note should have no negative consequences, but you might miss out on a trick that makes your life easier.

Important

Important boxes detail things that are easily missed: configuration changes that only apply to the current session, or services that need restarting before an update will apply. Ignoring a box labeled 'Important' will not cause data loss but may cause irritation and frustration.

Warning

Warnings should not be ignored. Ignoring warnings will most likely cause data loss.

2. Feedback

If you spot a typo, or if you have thought of a way to make this manual better, we would love to hear from you. Please submit a report in Bugzilla (*http://bugzilla.redhat.com/bugzilla/*) against the component **doc-Cluster_Administration**.

Be sure to mention the manual identifier:

```
Cluster_Administration(EN)-6 (2010-10-15T16:26)
```

By mentioning this manual's identifier, we know exactly which version of the guide you have.

If you have a suggestion for improving the documentation, try to be as specific as possible. If you have found an error, please include the section number and some of the surrounding text so we can find it easily.

Red Hat High Availability Add-On Configuration and Management Overview

Red Hat High Availability Add-On allows you to connect a group of computers (called *nodes* or *members*) to work together as a cluster. You can use Red Hat High Availability Add-On to suit your clustering needs (for example, setting up a cluster for sharing files on a GFS2 file system or setting up service failover).

1.1. Configuration Basics

To set up a cluster, you must connect the nodes to certain cluster hardware and configure the nodes into the cluster environment. This chapter provides an overview of cluster configuration and management, and tools available for configuring and managing the Red Hat High Availability Add-On.

Configuring and managing the Red Hat High Availability Add-On consists of the following basic steps:

1. Setting up hardware. Refer to *Section 1.2, "Setting Up Hardware"*.

2. Installing Red Hat High Availability Add-On software. Refer to *Section 1.3, "Installing Red Hat High Availability Add-On software"*.

3. Configuring Red Hat High Availability Add-On Software. Refer to *Section 1.4, "Configuring Red Hat High Availability Add-On Software"*.

1.2. Setting Up Hardware

Setting up hardware consists of connecting cluster nodes to other hardware required to run the Red Hat High Availability Add-On. The amount and type of hardware varies according to the purpose and availability requirements of the cluster. Typically, an enterprise-level cluster requires the following type of hardware (refer to *Figure 1.1, "Red Hat High Availability Add-On Hardware Overview"*). For considerations about hardware and other cluster configuration concerns, refer to *Chapter 2, Before Configuring the Red Hat High Availability Add-On* or check with an authorized Red Hat representative.

- High Availability Add-On nodes — Computers that are capable of running Red Hat Enterprise Linux 6 software, with at least 1GB of RAM.

- Ethernet switch or hub for public network — This is required for client access to the cluster.

- Ethernet switch or hub for private network — This is required for communication among the cluster nodes and other cluster hardware such as network power switches and Fibre Channel switches.

- Network power switch — A network power switch is recommended to perform fencing in an enterprise-level cluster.

- Fibre Channel switch — A Fibre Channel switch provides access to Fibre Channel storage. Other options are available for storage according to the type of storage interface; for example, iSCSI. A Fibre Channel switch can be configured to perform fencing.

- Storage — Some type of storage is required for a cluster. The type required depends on the purpose of the cluster.

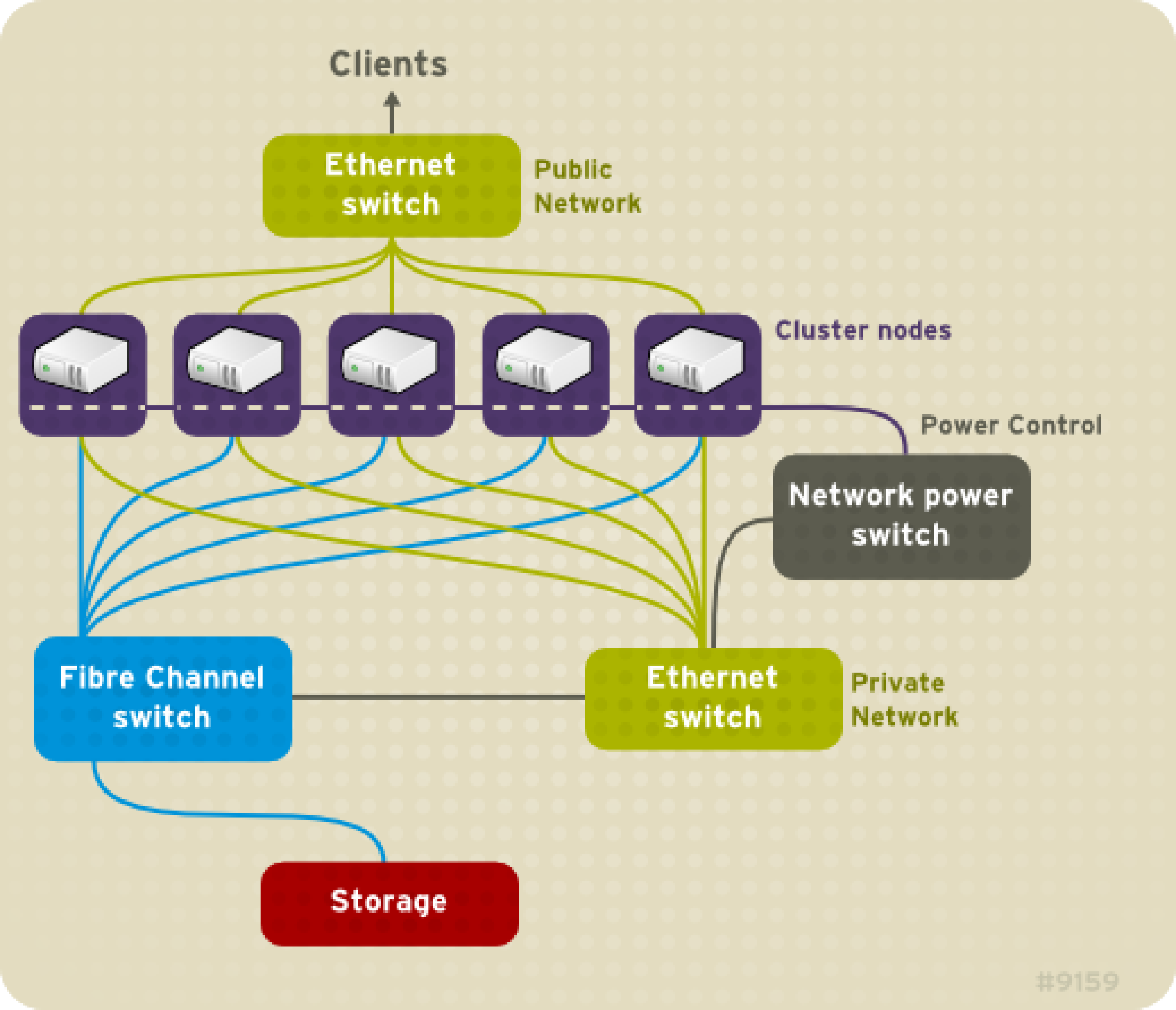


Figure 1.1. Red Hat High Availability Add-On Hardware Overview

1.3. Installing Red Hat High Availability Add-On software

To install Red Hat High Availability Add-On software, you must have entitlements for the software. If you are using the **Conga** configuration GUI, you can let it install the cluster software. If you are using other tools to configure the cluster, secure and install the software as you would with Red Hat Enterprise Linux software.

1.4. Configuring Red Hat High Availability Add-On Software

Configuring Red Hat High Availability Add-On software consists of using configuration tools to specify the relationship among the cluster components. The following cluster configuration tools are available with Red Hat High Availability Add-On:

- **Conga** — This is a comprehensive user interface for installing, configuring, and managing Red Hat High Availability Add-On. Refer to *Chapter 3, Configuring Red Hat High Availability Add-On With* ***Conga*** and *Chapter 4, Managing Red Hat High Availability Add-On With* ***Conga*** for information about configuring and managing High Availability Add-On with **Conga**.

- Command-line tools — This is a set of command-line tools for configuring and managing Red Hat High Availability Add-On. Refer to *Chapter 5, Configuring Red Hat High Availability Add-On With Command Line Tools* and *Chapter 6, Managing Red Hat High Availability Add-On With Command*

Line Tools for information about configuring and managing a cluster with command-line tools. Refer to *Appendix D, Command Line Tools Summary* for a summary of preferred command-line tools.

Note

`system-config-cluster` is not available in RHEL 6.

Before Configuring the Red Hat High Availability Add-On

This chapter describes tasks to perform and considerations to make before installing and configuring the Red Hat High Availability Add-On, and consists of the following sections.

Important

Make sure that your deployment of Red Hat High Availability Add-On meets your needs and can be supported. Consult with an authorized Red Hat representative to verify your configuration prior to deployment. In addition, allow time for a configuration burn-in period to test failure modes.

- *Section 2.1, "General Configuration Considerations"*
- *Section 2.2, "Compatible Hardware"*
- *Section 2.3, "Enabling IP Ports"*
- *Section 2.4, "Configuring ACPI For Use with Integrated Fence Devices"*
- *Section 2.5, "Considerations for Configuring HA Services"*
- *Section 2.6, "Configuration Validation"*
- *Section 2.7, "Considerations for **NetworkManager**"*
- *Section 2.8, "Considerations for Using Quorum Disk"*
- *Section 2.9, "Red Hat High Availability Add-On and SELinux"*
- *Section 2.10, "Multicast Addresses"*
- *Section 2.11, "Considerations for **`ricci`**"*
- *Section 2.12, "Considerations for Using **Conga**"*

2.1. General Configuration Considerations

You can configure the Red Hat High Availability Add-On in a variety of ways to suit your needs. Take into account the following general considerations when you plan, configure, and implement your deployment.

Number of cluster nodes supported

> The maximum number of cluster nodes supported by the High Availability Add-On is 16.

GFS2

> Although a GFS2 file system can be implemented in a standalone system or as part of a cluster configuration, Red Hat does not support the use of GFS2 as a single-node file system. Red Hat does support a number of high-performance single-node file systems that are optimized for single node, and thus have generally lower overhead than a cluster file system. Red Hat recommends using those file systems in preference to GFS2 in cases where only a single node needs to mount the file system. Red Hat will continue to support single-node GFS2 file systems for existing customers.

When you configure a GFS2 file system as a cluster file system, you must ensure that all nodes in the cluster have access to the shared file system. Asymmetric cluster configurations in which some nodes have access to the file system and others do not are not supported.This does not require that all nodes actually mount the GFS2 file system itself.

No-single-point-of-failure hardware configuration

Clusters can include a dual-controller RAID array, multiple bonded network channels, multiple paths between cluster members and storage, and redundant un-interruptible power supply (UPS) systems to ensure that no single failure results in application down time or loss of data.

Alternatively, a low-cost cluster can be set up to provide less availability than a no-single-point-of-failure cluster. For example, you can set up a cluster with a single-controller RAID array and only a single Ethernet channel.

Certain low-cost alternatives, such as host RAID controllers, software RAID without cluster support, and multi-initiator parallel SCSI configurations are not compatible or appropriate for use as shared cluster storage.

Data integrity assurance

To ensure data integrity, only one node can run a cluster service and access cluster-service data at a time. The use of power switches in the cluster hardware configuration enables a node to power-cycle another node before restarting that node's HA services during a failover process. This prevents two nodes from simultaneously accessing the same data and corrupting it. It is strongly recommended that *fence devices* (hardware or software solutions that remotely power, shutdown, and reboot cluster nodes) are used to guarantee data integrity under all failure conditions.

Ethernet channel bonding

Cluster quorum and node health is determined by communication of messages among cluster nodes via Ethernet. In addition, cluster nodes use Ethernet for a variety of other critical cluster functions (for example, fencing). With Ethernet channel bonding, multiple Ethernet interfaces are configured to behave as one, reducing the risk of a single-point-of-failure in the typical switched Ethernet connection among cluster nodes and other cluster hardware.

IPv4 and IPv6

The High Availability Add-On supports both IPv4 and IPv6 Internet Protocols. Support of IPv6 in the High Availability Add-On is new for Red Hat Enterprise Linux 6.

2.2. Compatible Hardware

Before configuring Red Hat High Availability Add-On software, make sure that your cluster uses appropriate hardware (for example, supported fence devices, storage devices, and Fibre Channel switches). Refer to the hardware configuration guidelines at *http://www.redhat.com/cluster_suite/hardware/* for the most current hardware compatibility information.

2.3. Enabling IP Ports

Before deploying the Red Hat High Availability Add-On, you must enable certain IP ports on the cluster nodes and on computers that run **luci** (the **Conga** user interface server). The following sections identify the IP ports to be enabled:

- *Section 2.3.1, “Enabling IP Ports on Cluster Nodes”*
- *Section 2.3.2, “Enabling IP Ports on Computers That Run **luci**”*

2.3.1. Enabling IP Ports on Cluster Nodes

To allow Red Hat High Availability Add-On nodes to communicate with each other, you must enable the IP ports assigned to certain Red Hat High Availability Add-On components. *Table 2.1, “Enabled IP Ports on Red Hat High Availability Add-On Nodes”* lists the IP port numbers, their respective protocols, and the components to which the port numbers are assigned. At each cluster node, enable IP ports according to *Table 2.1, “Enabled IP Ports on Red Hat High Availability Add-On Nodes”*. You can use `system-config-firewall` to enable the IP ports.

Table 2.1. Enabled IP Ports on Red Hat High Availability Add-On Nodes

IP Port Number	Protocol	Component
5404, 5405	UDP	`corosync/cman` (Cluster Manager)
11111	TCP	`ricci` (part of **Conga** remote agent)
21064	TCP	`dlm` (Distributed Lock Manager)
50006, 50008, 50009	TCP	`ccsd` (Cluster Configuration System daemon)
50007	UDP	`ccsd` (Cluster Configuration System daemon)

2.3.2. Enabling IP Ports on Computers That Run luci

To allow client computers to communicate with a computer that runs **luci** (the **Conga** user interface server), and to allow a computer that runs **luci** to communicate with **ricci** in the cluster nodes, you must enable the IP ports assigned to **luci** and **ricci**. *Table 2.2, “Enabled IP Ports on a Computer That Runs **luci**”* lists the IP port numbers, their respective protocols, and the components to which the port numbers are assigned. At each computer that runs **luci**, enable IP ports according to *Table 2.1, “Enabled IP Ports on Red Hat High Availability Add-On Nodes”*.

Note

If a cluster node is running **luci**, port 11111 should already have been enabled.

Table 2.2. Enabled IP Ports on a Computer That Runs **luci**

IP Port Number	Protocol	Component
8084	TCP	**luci** (**Conga** user interface server)
11111	TCP	`ricci` (**Conga** remote agent)

2.4. Configuring ACPI For Use with Integrated Fence Devices

If your cluster uses integrated fence devices, you must configure ACPI (Advanced Configuration and Power Interface) to ensure immediate and complete fencing.

Note

For the most current information about integrated fence devices supported by Red Hat High Availability Add-On, refer to *http://www.redhat.com/cluster_suite/hardware/*[1].

If a cluster node is configured to be fenced by an integrated fence device, disable ACPI Soft-Off for that node. Disabling ACPI Soft-Off allows an integrated fence device to turn off a node immediately and completely rather than attempting a clean shutdown (for example, **`shutdown -h now`**). Otherwise, if ACPI Soft-Off is enabled, an integrated fence device can take four or more seconds to turn off a node (refer to note that follows). In addition, if ACPI Soft-Off is enabled and a node panics or freezes during shutdown, an integrated fence device may not be able to turn off the node. Under those circumstances, fencing is delayed or unsuccessful. Consequently, when a node is fenced with an integrated fence device and ACPI Soft-Off is enabled, a cluster recovers slowly or requires administrative intervention to recover.

Note

The amount of time required to fence a node depends on the integrated fence device used. Some integrated fence devices perform the equivalent of pressing and holding the power button; therefore, the fence device turns off the node in four to five seconds. Other integrated fence devices perform the equivalent of pressing the power button momentarily, relying on the operating system to turn off the node; therefore, the fence device turns off the node in a time span much longer than four to five seconds.

To disable ACPI Soft-Off, use **`chkconfig`** management and verify that the node turns off immediately when fenced. The preferred way to disable ACPI Soft-Off is with **`chkconfig`** management: however, if that method is not satisfactory for your cluster, you can disable ACPI Soft-Off with one of the following alternate methods:

- Changing the BIOS setting to "instant-off" or an equivalent setting that turns off the node without delay

  ### Note

  Disabling ACPI Soft-Off with the BIOS may not be possible with some computers.

- Appending **`acpi=off`** to the kernel boot command line of the **`/boot/grub/grub.conf`** file

[1] http://www.redhat.com/cluster_suite/hardware/

Important

This method completely disables ACPI; some computers do not boot correctly if ACPI is completely disabled. Use this method *only* if the other methods are not effective for your cluster.

The following sections provide procedures for the preferred method and alternate methods of disabling ACPI Soft-Off:

- *Section 2.4.1, "Disabling ACPI Soft-Off with **chkconfig** Management"* — Preferred method
- *Section 2.4.2, "Disabling ACPI Soft-Off with the BIOS"* — First alternate method
- *Section 2.4.3, "Disabling ACPI Completely in the **grub.conf** File"* — Second alternate method

2.4.1. Disabling ACPI Soft-Off with chkconfig Management

You can use **chkconfig** management to disable ACPI Soft-Off either by removing the ACPI daemon (**acpid**) from **chkconfig** management or by turning off **acpid**.

Note

This is the preferred method of disabling ACPI Soft-Off.

Disable ACPI Soft-Off with **chkconfig** management at each cluster node as follows:

1. Run either of the following commands:
   - **chkconfig --del acpid** — This command removes **acpid** from **chkconfig** management.

     — OR —

   - **chkconfig --level 2345 acpid off** — This command turns off **acpid**.
2. Reboot the node.
3. When the cluster is configured and running, verify that the node turns off immediately when fenced.

   Note

   You can fence the node with the **fence_node** command or **Conga**.

2.4.2. Disabling ACPI Soft-Off with the BIOS

The preferred method of disabling ACPI Soft-Off is with **chkconfig** management (*Section 2.4.1, "Disabling ACPI Soft-Off with **chkconfig** Management"*). However, if the preferred method is not effective for your cluster, follow the procedure in this section.

Note

Disabling ACPI Soft-Off with the BIOS may not be possible with some computers.

You can disable ACPI Soft-Off by configuring the BIOS of each cluster node as follows:

1. Reboot the node and start the **`BIOS CMOS Setup Utility`** program.

2. Navigate to the **Power** menu (or equivalent power management menu).

3. At the **Power** menu, set the **Soft-Off by PWR-BTTN** function (or equivalent) to **Instant-Off** (or the equivalent setting that turns off the node via the power button without delay). *Example 2.1, “**`BIOS CMOS Setup Utility`**: **Soft-Off by PWR-BTTN** set to **Instant-Off**”* shows a **Power** menu with **ACPI Function** set to **Enabled** and **Soft-Off by PWR-BTTN** set to **Instant-Off**.

   Note

   The equivalents to **ACPI Function**, **Soft-Off by PWR-BTTN**, and **Instant-Off** may vary among computers. However, the objective of this procedure is to configure the BIOS so that the computer is turned off via the power button without delay.

4. Exit the **`BIOS CMOS Setup Utility`** program, saving the BIOS configuration.

5. When the cluster is configured and running, verify that the node turns off immediately when fenced.

   Note

   You can fence the node with the **`fence_node`** command or **Conga**.

Example 2.1. **`BIOS CMOS Setup Utility`**: **Soft-Off by PWR-BTTN** set to **Instant-Off**

```
+---------------------------------------------|-------------------+
|    ACPI Function             [Enabled]      |    Item Help      |
|    ACPI Suspend Type         [S1(POS)]      |-------------------|
|  x Run VGABIOS if S3 Resume   Auto          |   Menu Level   *  |
|    Suspend Mode              [Disabled]     |                   |
|    HDD Power Down            [Disabled]     |                   |
|    Soft-Off by PWR-BTTN      [Instant-Off   |                   |
|    CPU THRM-Throttling       [50.0%]        |                   |
|    Wake-Up by PCI card       [Enabled]      |                   |
|    Power On by Ring          [Enabled]      |                   |
|    Wake Up On LAN            [Enabled]      |                   |
|  x USB KB Wake-Up From S3     Disabled      |                   |
|    Resume by Alarm           [Disabled]     |                   |
|  x  Date(of Month) Alarm       0            |                   |
|  x  Time(hh:mm:ss) Alarm       0 :  0 :     |                   |
|    POWER ON Function         [BUTTON ONLY   |                   |
|  x KB Power ON Password       Enter         |                   |
|  x Hot Key Power ON           Ctrl-F1       |                   |
|                                             |                   |
|                                               |                    |
+-----------------------------------------------|--------------------+
```

This example shows **ACPI Function** set to **Enabled**, and **Soft-Off by PWR-BTTN** set to **Instant-Off**.

2.4.3. Disabling ACPI Completely in the `grub.conf` File

The preferred method of disabling ACPI Soft-Off is with **`chkconfig`** management (*Section 2.4.1, “Disabling ACPI Soft-Off with **`chkconfig`** Management”*). If the preferred method is not effective for your cluster, you can disable ACPI Soft-Off with the BIOS power management (*Section 2.4.2, “Disabling ACPI Soft-Off with the BIOS”*). If neither of those methods is effective for your cluster, you can disable ACPI completely by appending **`acpi=off`** to the kernel boot command line in the **`grub.conf`** file.

Important

This method completely disables ACPI; some computers do not boot correctly if ACPI is completely disabled. Use this method *only* if the other methods are not effective for your cluster.

You can disable ACPI completely by editing the **`grub.conf`** file of each cluster node as follows:

1. Open **`/boot/grub/grub.conf`** with a text editor.

2. Append **`acpi=off`** to the kernel boot command line in **`/boot/grub/grub.conf`** (refer to *Example 2.2, “Kernel Boot Command Line with **`acpi=off`** Appended to It”*).

3. Reboot the node.

4. When the cluster is configured and running, verify that the node turns off immediately when fenced.

   Note

   You can fence the node with the **`fence_node`** command or **Conga**.

Example 2.2. Kernel Boot Command Line with **`acpi=off`** Appended to It

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,0)
#          kernel /vmlinuz-version ro root=/dev/VolGroup00/LogVol00
#          initrd /initrd-version.img
#boot=/dev/hda
default=0
timeout=5
serial --unit=0 --speed=115200
terminal --timeout=5 serial console
title Red Hat Enterprise Linux Server (2.6.18-36.el5)
```

```
        root (hd0,0)
        kernel /vmlinuz-2.6.18-36.el5 ro root=/dev/VolGroup00/LogVol00
console=ttyS0,115200n8 acpi=off
        initrd /initrd-2.6.18-36.el5.img
```

> In this example, **`acpi=off`** has been appended to the kernel boot command line — the line starting with "kernel /vmlinuz-2.6.18-36.el5".

2.5. Considerations for Configuring HA Services

You can create a cluster to suit your needs for high availability by configuring HA (high-availability) services. The key component for HA service management in the Red Hat High Availability Add-On, **`rgmanager`**, implements cold failover for off-the-shelf applications. In the Red Hat High Availability Add-On, an application is configured with other cluster resources to form an HA service that can fail over from one cluster node to another with no apparent interruption to cluster clients. HA-service failover can occur if a cluster node fails or if a cluster system administrator moves the service from one cluster node to another (for example, for a planned outage of a cluster node).

To create an HA service, you must configure it in the cluster configuration file. An HA service comprises cluster *resources*. Cluster resources are building blocks that you create and manage in the cluster configuration file — for example, an IP address, an application initialization script, or a Red Hat GFS2 shared partition.

An HA service can run on only one cluster node at a time to maintain data integrity. You can specify failover priority in a failover domain. Specifying failover priority consists of assigning a priority level to each node in a failover domain. The priority level determines the failover order — determining which node that an HA service should fail over to. If you do not specify failover priority, an HA service can fail over to any node in its failover domain. Also, you can specify if an HA service is restricted to run only on nodes of its associated failover domain. (When associated with an unrestricted failover domain, an HA service can start on any cluster node in the event no member of the failover domain is available.)

Figure 2.1, “Web Server Cluster Service Example” shows an example of an HA service that is a web server named "content-webserver". It is running in cluster node B and is in a failover domain that consists of nodes A, B, and D. In addition, the failover domain is configured with a failover priority to fail over to node D before node A and to restrict failover to nodes only in that failover domain. The HA service comprises these cluster resources:

- IP address resource — IP address 10.10.10.201.
- An application resource named "httpd-content" — a web server application init script **`/etc/init.d/httpd`** (specifying **`httpd`**).
- A file system resource — Red Hat GFS2 named "gfs2-content-webserver".

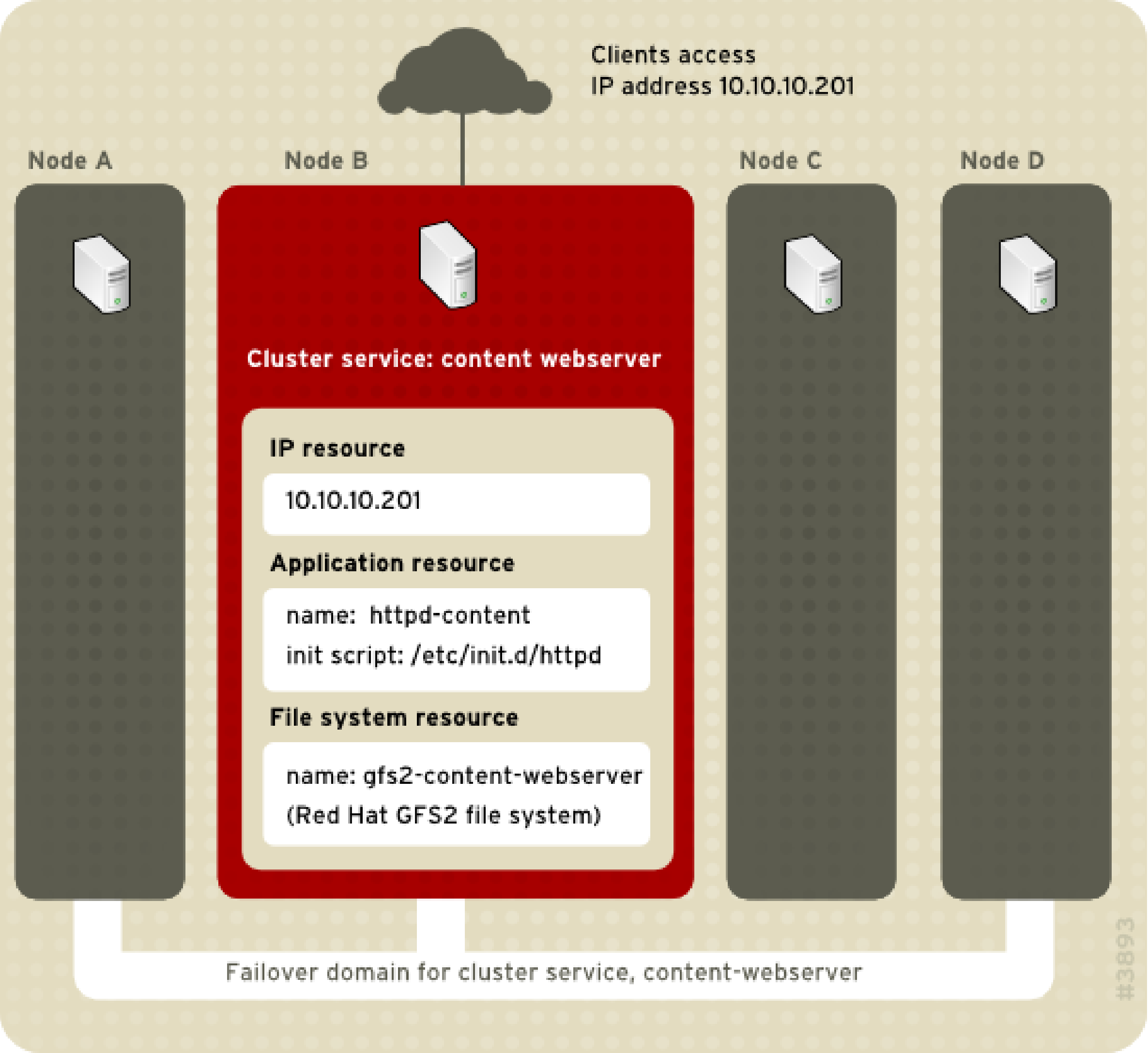

Clients access
IP address 10.10.10.201
Node A
Node B
Node C
Node D
Cluster service: content webserver
IP resource
10.10.10.201
Application resource
name: httpd-content
init script: /etc/init.d/httpd
File system resource
name: gfs2-content-webserver
(Red Hat GFS2 file system)
Failover domain for cluster service, content-webserver
#3893

Figure 2.1. Web Server Cluster Service Example

Clients access the HA service through the IP address 10.10.10.201, enabling interaction with the web server application, httpd-content. The httpd-content application uses the gfs2-content-webserver file system. If node B were to fail, the content-webserver HA service would fail over to node D. If node D were not available or also failed, the service would fail over to node A. Failover would occur with minimal service interruption to the cluster clients. For example, in an HTTP service, certain state information may be lost (like session data). The HA service would be accessible from another cluster node via the same IP address as it was before failover.

Note

For more information about HA services and failover domains, refer to *Red Hat Cluster Suite Overview*. For information about configuring failover domains, refer to *Chapter 3, Configuring Red Hat High Availability Add-On With **Conga***(using **Conga**) or *Chapter 5, Configuring Red Hat High Availability Add-On With Command Line Tools* (using command line utilities).

An HA service is a group of cluster resources configured into a coherent entity that provides specialized services to clients. An HA service is represented as a resource tree in the cluster configuration file, **`/etc/cluster/cluster.conf`** (in each cluster node). In the cluster configuration file, each resource tree is an XML representation that specifies each resource, its attributes, and its relationship among other resources in the resource tree (parent, child, and sibling relationships).

> **Note**
>
> Because an HA service consists of resources organized into a hierarchical tree, a service is sometimes referred to as a *resource tree* or *resource group*. Both phrases are synonymous with *HA service*.

At the root of each resource tree is a special type of resource — a *service resource*. Other types of resources comprise the rest of a service, determining its characteristics. Configuring an HA service consists of creating a service resource, creating subordinate cluster resources, and organizing them into a coherent entity that conforms to hierarchical restrictions of the service.

The High Availability Add-On supports the following HA services:

- Apache
- Application (Script)
- LVM (HA LVM)
- MySQL
- NFS
- Open LDAP
- Oracle
- PostgreSQL 8
- Samba
- SAP
- Tomcat 6

There are two major considerations to take into account when configuring an HA service:

- The types of resources needed to create a service
- Parent, child, and sibling relationships among resources

The types of resources and the hierarchy of resources depend on the type of service you are configuring.

The types of cluster resources are listed in *Appendix B, HA Resource Parameters*. Information about parent, child, and sibling relationships among resources is described in *Appendix C, HA Resource Behavior*.

2.6. Configuration Validation

The cluster configuration is automatically validated according to the cluster schema at **`/usr/share/cluster/cluster.rng`** during startup time and when a configuration is reloaded. Also, you can validate a cluster configuration any time by using the **`ccs_config_validate`** command.

An annotated schema is available for viewing at **`/usr/share/doc/cman-X.Y.ZZ/cluster_conf.html`** (for example **`/usr/share/doc/cman-3.0.12/cluster_conf.html`**).

Configuration validation checks for the following basic errors:

- XML validity — Checks that the configuration file is a valid XML file.
- Configuration options — Checks to make sure that options (XML elements and attributes) are valid.
- Option values — Checks that the options contain valid data (limited).

The following examples show a valid configuration and invalid configurations that illustrate the validation checks:

- Valid configuration — *Example 2.3, "**`cluster.conf`** Sample Configuration: Valid File"*
- Invalid XML — *Example 2.4, "**`cluster.conf`** Sample Configuration: Invalid XML"*
- Invalid option — *Example 2.5, "**`cluster.conf`** Sample Configuration: Invalid Option"*
- Invalid option value — *Example 2.6, "**`cluster.conf`** Sample Configuration: Invalid Option Value"*

Example 2.3. **`cluster.conf`** Sample Configuration: Valid File

```
<cluster name="mycluster" config_version="1">
  <logging debug="off"/>
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

Example 2.4. **`cluster.conf`** Sample Configuration: Invalid XML

```
<cluster name="mycluster" config_version="1">
  <logging debug="off"/>
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
   </fencedevices>
   <rm>
   </rm>
<cluster>         <----------------INVALID
```

In this example, the last line of the configuration (annotated as "INVALID" here) is missing a slash — it is **`<cluster>`** instead of **`</cluster>`**.

Example 2.5. `cluster.conf` Sample Configuration: Invalid Option

```
<cluster name="mycluster" config_version="1">
  <loging debug="off"/>         <----------------INVALID
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
   </fencedevices>
   <rm>
   </rm>
<cluster>
```

In this example, the second line of the configuration (annotated as "INVALID" here) contains an invalid XML element — it is **`loging`** instead of **`logging`**.

Example 2.6. `cluster.conf` Sample Configuration: Invalid Option Value

```
<cluster name="mycluster" config_version="1">
  <loging debug="off"/>
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="-1">  <--------INVALID
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
   </fencedevices>
   <rm>
   </rm>
<cluster>
```

In this example, the fourth line of the configuration (annotated as "INVALID" here) contains an invalid value for the XML attribute, **`nodeid`** in the **`clusternode`** line for **`node-01.example.com`**. The value is a negative value ("-1") instead of a positive value ("1"). For the **`nodeid`** attribute, the value must be a positive value.

2.7. Considerations for NetworkManager

The use of **NetworkManager** is not supported on cluster nodes. If you have installed **NetworkManager** on your cluster nodes, you should either remove it or disable it.

2.8. Considerations for Using Quorum Disk

Quorum Disk is a disk-based quorum daemon, **`qdiskd`**, that provides supplemental heuristics to determine node fitness. With heuristics you can determine factors that are important to the operation of the node in the event of a network partition. For example, in a four-node cluster with a 3:1 split, ordinarily, the three nodes automatically "win" because of the three-to-one majority. Under those circumstances, the one node is fenced. With **`qdiskd`** however, you can set up heuristics that allow the one node to win based on access to a critical resource (for example, a critical network path). If your cluster requires additional methods of determining node health, then you should configure **`qdiskd`** to meet those needs.

Note

Configuring **`qdiskd`** is not required unless you have special requirements for node health. An example of a special requirement is an "all-but-one" configuration. In an all-but-one configuration, **`qdiskd`** is configured to provide enough quorum votes to maintain quorum even though only one node is working.

Important

Overall, heuristics and other **`qdiskd`** parameters for your deployment depend on the site environment and special requirements needed. To understand the use of heuristics and other **`qdiskd`** parameters, refer to the qdisk(5) man page. If you require assistance understanding and using **`qdiskd`** for your site, contact an authorized Red Hat support representative.

If you need to use **`qdiskd`**, you should take into account the following considerations:

Cluster node votes

Each cluster node should have the same number of votes.

CMAN membership timeout value

The CMAN membership timeout value (the time a node needs to be unresponsive before CMAN considers that node to be dead, and not a member) should be at least two times that of the **`qdiskd`** membership timeout value. The reason is because the quorum daemon must detect failed nodes on its own, and can take much longer to do so than CMAN. The default value for CMAN membership timeout is 10 seconds. Other site-specific conditions may affect the relationship between the membership timeout values of CMAN and **`qdiskd`**. For assistance with adjusting the CMAN membership timeout value, contact an authorized Red Hat support representative.

Fencing

To ensure reliable fencing when using **`qdiskd`**, use power fencing. While other types of fencing can be reliable for clusters not configured with **`qdiskd`**, they are not reliable for a cluster configured with **`qdiskd`**.

Maximum nodes

A cluster configured with **`qdiskd`** supports a maximum of 16 nodes. The reason for the limit is because of scalability; increasing the node count increases the amount of synchronous I/O contention on the shared quorum disk device.

Quorum disk device

A quorum disk device should be a shared block device with concurrent read/write access by all nodes in a cluster. The minimum size of the block device is 10 Megabytes. Examples of shared block devices that can be used by **`qdiskd`** are a multi-port SCSI RAID array, a Fibre Channel RAID SAN, or a RAID-configured iSCSI target. You can create a quorum disk device with **`mkqdisk`**, the Cluster Quorum Disk Utility. For information about using the utility refer to the mkqdisk(8) man page.

> **Note**
>
> Using JBOD as a quorum disk is not recommended. A JBOD cannot provide dependable performance and therefore may not allow a node to write to it quickly enough. If a node is unable to write to a quorum disk device quickly enough, the node is falsely evicted from a cluster.

2.9. Red Hat High Availability Add-On and SELinux

The High Availability Add-On for Red Hat Enterprise Linux 6 supports SELinux in the **`enforcing`** state with the SELinux policy type set to **`targeted`**.

For more information about SELinux, refer to *Deployment Guide* for Red Hat Enterprise Linux 6.

2.10. Multicast Addresses

Red Hat High Availability Add-On nodes communicate among each other using multicast addresses. Therefore, each network switch and associated networking equipment in the Red Hat High Availability Add-On must be configured to enable multicast addresses and support IGMP (Internet Group Management Protocol). Ensure that each network switch and associated networking equipment in the Red Hat High Availability Add-On are capable of supporting multicast addresses and IGMP; if they are, ensure that multicast addressing and IGMP are enabled. Without multicast and IGMP, not all nodes can participate in a cluster, causing the cluster to fail.

Note

Procedures for configuring network switches and associated networking equipment vary according each product. Refer to the appropriate vendor documentation or other information about configuring network switches and associated networking equipment to enable multicast addresses and IGMP.

2.11. Considerations for `ricci`

For Red Hat Enterprise Linux 6, **`ricci`** (a component of **Conga**), replaces **`ccsd`**. Therefore, it is necessary that **`ricci`** is running in each cluster node to be able to propagate updated cluster configuration information via the **`cman_tool -r`** command. You can start **`ricci`** by using **`service ricci start`** or by enabling it to start at boot time via **`chkconfig`**.

2.12. Considerations for Using Conga

When using **Conga** to configure and manage the Red Hat High Availability Add-On, make sure that each computer running **luci** (the **Conga** user interface server) is running on the same network that the cluster is using for cluster communication. Otherwise, **luci** cannot configure the nodes to communicate on the right network. If the computer running **luci** is on another network (for example, a public network rather than a private network that the cluster is communicating on), contact an authorized Red Hat support representative to make sure that the appropriate host name is configured for each cluster node.

Configuring Red Hat High Availability Add-On With Conga

This chapter describes how to configure Red Hat High Availability Add-On software using **Conga**. For information on using **Conga** to manage a running cluster, see *Chapter 4, Managing Red Hat High Availability Add-On With **Conga***.

This chapter consists of the following sections:

- *Section 3.1, “Configuration Tasks”*
- *Section 3.2, “Starting **luci** and **ricci**”*
- *Section 3.3, “Creating A Cluster”*
- *Section 3.4, “Global Cluster Properties”*
- *Section 3.5, “Configuring Fence Devices”*
- *Section 3.6, “Configuring Fencing for Cluster Members”*
- *Section 3.7, “Configuring a Failover Domain”*
- *Section 3.8, “Configuring Global Cluster Resources”*
- *Section 3.9, “Adding a Cluster Service to the Cluster”*

3.1. Configuration Tasks

Configuring Red Hat High Availability Add-On software with **Conga** consists of the following steps:

1. Configuring and running the **Conga** configuration user interface — the **luci** server. Refer to *Section 3.2, “Starting **luci** and **ricci**”*.
2. Creating a cluster. Refer to *Section 3.3, “Creating A Cluster”*.
3. Configuring global cluster properties. Refer to *Section 3.4, “Global Cluster Properties”*.
4. Configuring fence devices. Refer to *Section 3.5, “Configuring Fence Devices”*.
5. Configuring fencing for cluster members. Refer to *Section 3.6, “Configuring Fencing for Cluster Members”*.
6. Creating failover domains. Refer to *Section 3.7, “Configuring a Failover Domain”*.
7. Creating resources. Refer to *Section 3.8, “Configuring Global Cluster Resources”*.
8. Creating cluster services. Refer to *Section 3.9, “Adding a Cluster Service to the Cluster”*.

3.2. Starting luci and ricci

> **Note**
>
> Before starting **luci** and **ricci**, ensure that the IP ports on your cluster nodes allow connections to port 11111 from the **luci** server on any nodes that **luci** will be communicating with. For information on enabling IP ports on cluster nodes, see *Section 2.3.1, “Enabling IP Ports on Cluster Nodes”*.

To administer Red Hat High Availability Add-On with **Conga**, install and run **luci** and **ricci** as follows:

1. At each node to be administered by **Conga**, install the **ricci** agent. For example:

   ```
   # yum install ricci
   ```

2. At each node to be administered by **Conga**, start **ricci**. For example:

   ```
   # service ricci start
   Starting ricci:                                            [  OK  ]
   ```

3. Select a computer to host **luci** and install the **luci** software on that computer. For example:

   ```
   # yum install luci
   ```

   > **Note**
   >
   > Typically, a computer in a server cage or a data center hosts **luci**; however, a cluster computer can host **luci**.

4. Start **luci** using **`service luci start`**. For example:

   ```
   # service luci start
   Starting luci: generating https SSL certificates...  done
                                                              [  OK  ]

   Please, point your web browser to https://nano-01:8084 to access luci
   ```

5. At a Web browser, place the URL of the **luci** server into the URL address box and click **Go** (or the equivalent). The URL syntax for the **luci** server is **`https://`*`luci_server_hostname`*`:8084`**. The first time you access **luci**, two SSL certificate dialog boxes are displayed. Upon acknowledging the dialog boxes, your Web browser displays the **luci** login page.

6. From the **luci** login page, enter the root ID and root password for the system that is hosting **luci**.

7. After you log on, **luci** displays the **Homebase** page, as shown in *Figure 3.1, “luci Homebase page”*.

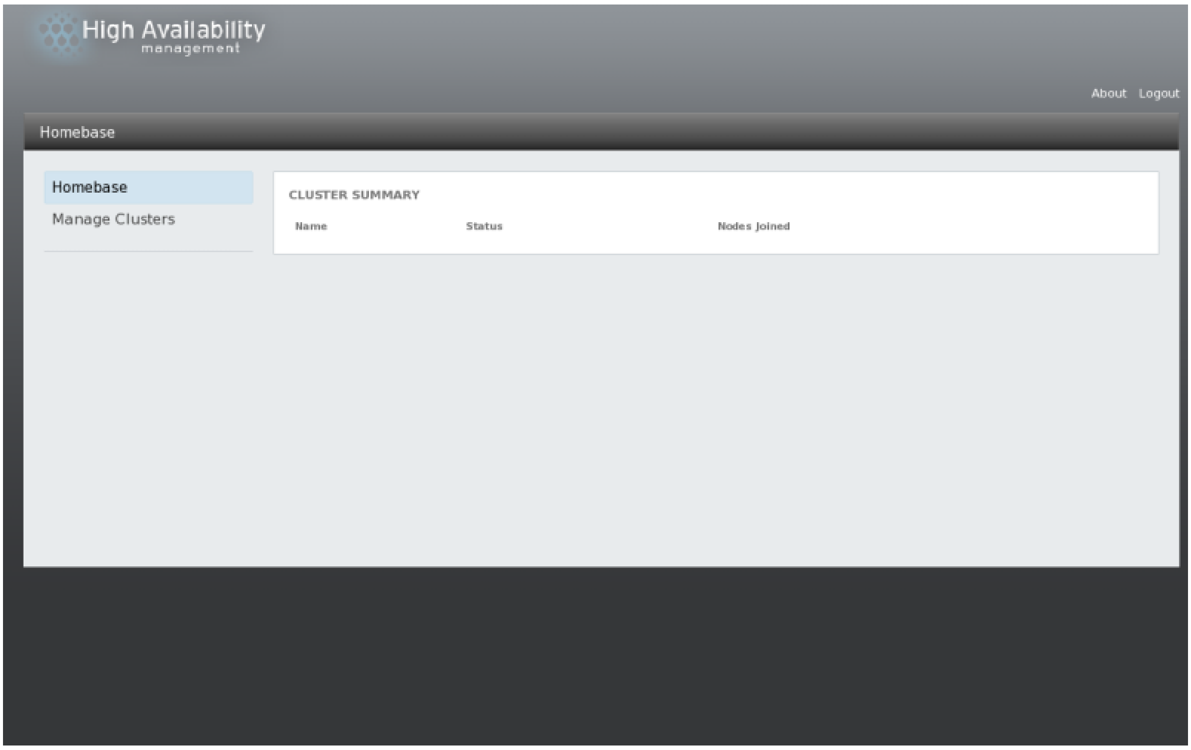


Figure 3.1. luci Homebase page

3.3. Creating A Cluster

Creating a cluster with **luci** consists of naming a cluster, adding cluster nodes to the cluster, entering their passwords, and submitting the request to create a cluster. If the node information and passwords are correct, **Conga** automatically installs software into the cluster nodes and starts the cluster. Create a cluster as follows:

1. Click **Manage Clusters** from the menu on the left side of the luci **Homebase** page. The Clusters screen appears, as shown in *Figure 3.2, "luci cluster management page"*.

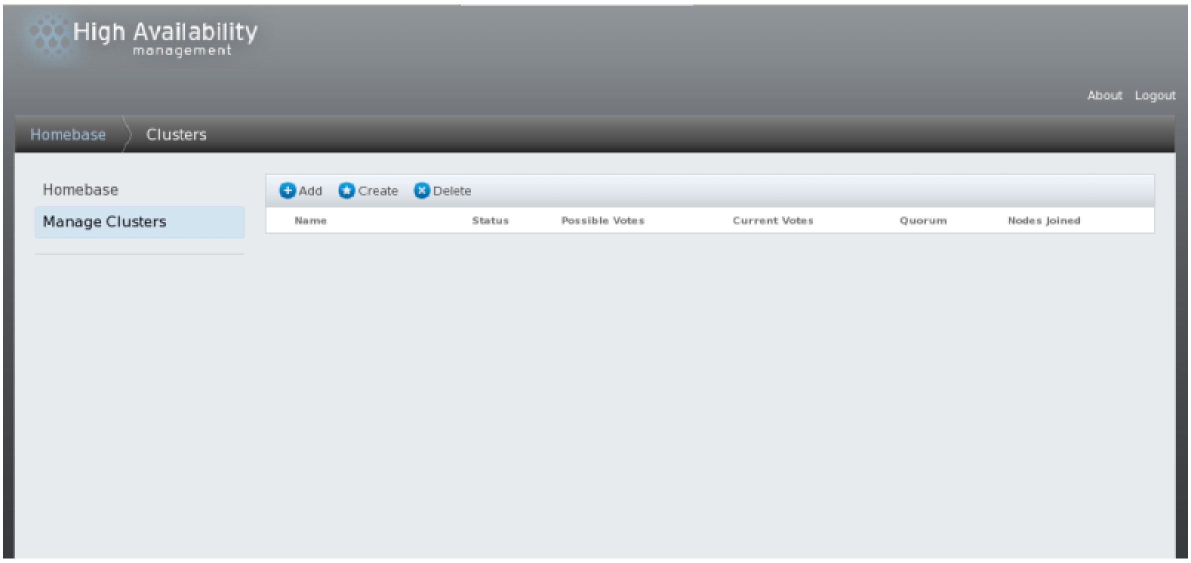
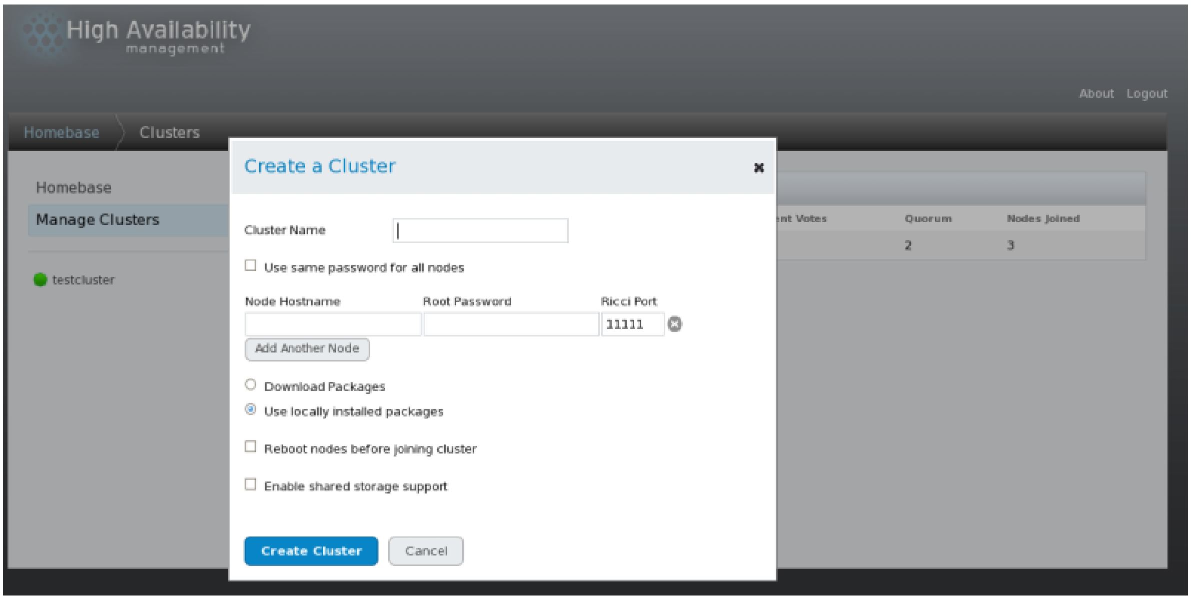


Figure 3.2. luci cluster management page

2. Click **Create**. The **Create a Cluster** screen appears, as shown in *Figure 3.3, “luci cluster creation screen”*.



Create a Cluster

Cluster Name

□ Use same password for all nodes

Node Hostname	Root Password	Ricci Port
		11111

○ Download Packages

● Use locally installed packages

□ Reboot nodes before joining cluster

□ Enable shared storage support

Figure 3.3. luci cluster creation screen

3. Enter the following parameters on the **Create a Cluster** screen, as necessary:

   - At the **Cluster Name** text box, enter a cluster name. The cluster name cannot exceed 15 characters.

   - If each node in the cluster has the same root password, you can check **Use same password for all nodes** to autofill the password field as you add nodes.

   - Enter the node name for a node in the cluster in the **Node Hostname** column and enter the root password for the node in the **Root Password** column. If you are using a different port for the **ricci** agent than the default of 11111, you can change that parameter.

   - Click on **Add Another Node** and enter the node name and root password for each additional node in the cluster.

   - If you do not want to upgrade the cluster software packages that are already installed on the nodes when you create the cluster, leave the **Use locally installed packages** option selected. If you want to upgrade all cluster software packages, select the **Download Packages** option.

     Note

     Whether you select the **Use locally installed packages** or the **Download Packages** option, if any of the base cluster components are missing (`cman`, `rgmanager`, `modcluster` and all their dependencies), they will be installed. If they cannot be installed, the node creation will fail.

   - Select **Enable shared storage support** if clustered storage is required; This downloads the packages that support clustered storage and enables clustered LVM. You should select this only when you have access to the Resilient Storage Add-On or the Scalable File System Add-On.

4. Click **Submit**. Clicking **Submit** causes the following actions:

   a. The cluster software packages are downloaded onto the added node.

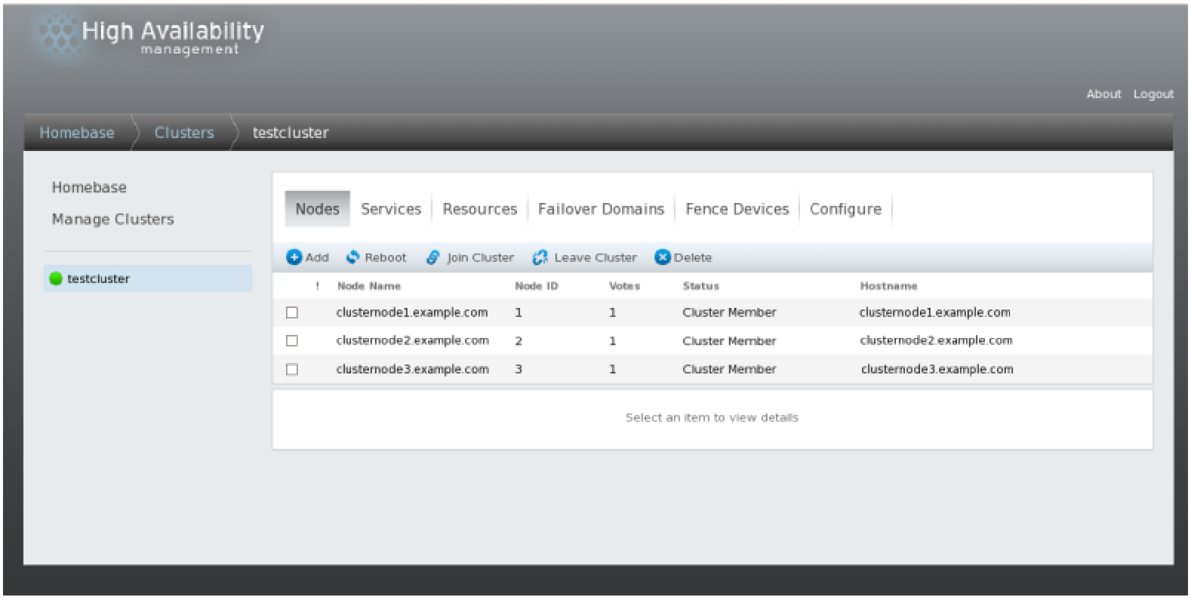
   b. Cluster software is installed onto the added node (or it is verified that the appropriate software packages are installed).

   c. The cluster configuration file is updated and propagated to each node in the cluster — including the added node.

   d. The added node joins the cluster.

   A message is displayed indicating that the cluster is being created. Refresh the page to see the current status of the cluster creation. When the cluster is ready, the display shows the status of the newly created cluster.

   Click on the cluster name to display the nodes that make up the cluster, as shown in *Figure 3.4, “Cluster node display”*.



	!	Node Name	Node ID	Votes	Status	Hostname
□		clusternode1.example.com	1	1	Cluster Member	clusternode1.example.com
□		clusternode2.example.com	2	1	Cluster Member	clusternode2.example.com
□		clusternode3.example.com	3	1	Cluster Member	clusternode3.example.com

Select an item to view details

Figure 3.4. Cluster node display

5. After clicking **Submit** to create the cluster, you can still add or delete nodes from the cluster by clicking the **Add** or **Delete** function from the menu at the top of the cluster node display page. For information on deleting a node from an existing cluster that is currently in operation, see *Section 4.2.4, “Deleting a Member from a Cluster”*.

3.4. Global Cluster Properties

When you select a cluster to configure, a cluster-specific page is displayed. The page provides an interface for configuring cluster-wide properties. You can configure cluster-wide properties by clicking on **Configure** along the top of the cluster display. This yields a tabbed interface which provides the following tabs: **General Properties**, **Fence Daemon Properties**, **Network Configuration**, and **QDisk Configuration**. To configure the parameters in those tabs, follow the steps in this section. If you do not need to configure parameters in a tab, skip the step for that tab.

1. **General Properties** tab — This tab displays the cluster name and provides an interface for modifying the configuration version.

- The **Cluster Name** text box displays the cluster name; it does not accept a cluster name change. The only way to change the name of a cluster is to create a new cluster configuration with the new name.

- The **Configuration Version** value is set to **1** by default and is automatically incremented each time you modify your cluster configuration. However, if you need to set it to another value, you can specify it at the **Configuration Version** text box.

If you have changed the **Configuration Version** value, click **Apply** for this change to take effect.

2. **Fence Daemon Properties** tab — This tab provides an interface for configuring these **Fence Daemon Properties** parameters: **Post-Fail Delay** and **Post-Join Delay**. The values you configure for these parameters are general fencing properties for the cluster. To configure specific fence devices for the nodes of the cluster, use the **Fence Devices** menu item of the cluster display, as described in *Section 3.5, "Configuring Fence Devices"*.

   The general fencing properties for the cluster you can configure with the **Fence Daemon Properties** tab are summarized as follows:

   - The **Post-Fail Delay** parameter is the number of seconds the fence daemon (`fenced`) waits before fencing a node (a member of the fence domain) after the node has failed. The **Post-Fail Delay** default value is **0**. Its value may be varied to suit cluster and network performance.

   - The **Post-Join Delay** parameter is the number of seconds the fence daemon (`fenced`) waits before fencing a node after the node joins the fence domain. The **Post-Join Delay** default value is **3**. A typical setting for **Post-Join Delay** is between 20 and 30 seconds, but can vary according to cluster and network performance.

   Enter the values required and click **Apply** for changes to take effect.

   > **Note**
   >
   > For more information about **Post-Join Delay** and **Post-Fail Delay**, refer to the fenced(8) man page.

3. **Network Configuration** tab — This tab provides an interface for configuring multicast parameters.

   You can use this tab to configure these multicast parameters: **Let cluster choose the multicast address** and **Specify the multicast address manually**. The default setting is **Let cluster choose the multicast address**. If you need to use a specific multicast address, click **Specify the multicast address manually** and enter a multicast address into the text box.

   You can enter advanced cluster properties by clicking **Show Advanced Properties**, which reveals a list of advanced properties you can reset from their default values. It is recommended that you leave these properties at their default values.

   Click **Apply** for changes to take effect.

   If you do not specify a multicast address, the Red Hat High Availability Add-On software creates one based on the cluster ID. It generates the lower 16 bits of the address and appends them to the upper portion of the address according to whether the IP protocol is IPV4 or IPV6:

- For IPV4 — The address formed is 239.192. plus the lower 16 bits generated by Red Hat High Availability Add-On software.
- For IPV6 — The address formed is FF15:: plus the lower 16 bits generated by Red Hat High Availability Add-On software.

If you do not specify a multicast address, the Red Hat High Availability Add-On software creates one by generating the lower 16 bits of the address and appending them to the upper portion of the address, 239.192. To ensure a unique multicast address, the Red Hat High Availability Add-On software generates the lower 16 bits based on the cluster ID.

> **Note**
>
> The cluster ID is a unique identifier that **`cman`** generates for each cluster. To view the cluster ID, run the **`cman_tool status`** command on a cluster node.

If you do specify a multicast address, you should use the 239.192.x.x series (or FF15:: for IPv6) that **`cman`** uses. Otherwise, using a multicast address outside that range may cause unpredictable results. For example, using 224.0.0.x (which is "All hosts on the network") may not be routed correctly, or even routed at all by some hardware.

> **Note**
>
> If you specify a multicast address, make sure that you check the configuration of routers that cluster packets pass through. Some routers may take a long time to learn addresses, seriously impacting cluster performance.

4. **QDisk Configuration** tab — This tab provides an interface for configuring these **Quorum Disk Configuration** parameters: **Do not use a Quorum Disk**, **Use a Quorum Disk**, **Interval**, **Votes**, **TKO**, and **Minimum Score**. This tab also provides an interface to specify a physical device to use, and to specify the heuristics to use. The **Do not use a Quorum Disk** parameter is enabled by default. *Table 3.1, “Quorum-Disk Parameters”* describes the parameters. If you need to use a quorum disk, click **Use a Quorum Disk**, enter quorum disk parameters, click **Apply**, and restart the cluster for the changes to take effect.

> **Important**
>
> Quorum-disk parameters and heuristics depend on the site environment and the special requirements needed. To understand the use of quorum-disk parameters and heuristics, refer to the qdisk(5) man page. If you require assistance understanding and using quorum disk, contact an authorized Red Hat support representative.

> **Note**
>
> Clicking **Apply** on the **QDisk Configuration** tab propagates changes to the cluster configuration file (**`/etc/cluster/cluster.conf`**) in each cluster node. However, for the quorum disk to operate, you must restart the cluster (refer to *Section 4.3, “Starting, Stopping, Restarting, and Deleting Clusters”*).

Table 3.1. Quorum-Disk Parameters

Parameter	Description
Do not use a Quorum Disk	Disables quorum disk. Disables quorum-disk parameters in the **Qdisk Configuration** tab.
Use a Quorum Disk	Enables quorum disk. Enables quorum-disk parameters in the **Qdisk Configuration** tab.
Interval	The frequency of read/write cycles, in seconds. The value of **Interval** is automatically determined if left blank in the **Quorum Disk Configuration** dialog box.
Votes	The number of votes the quorum daemon advertises to **`cman`** when it has a high enough score. The value of **Votes** is automatically determined if left blank in the **Quorum Disk Configuration** dialog box.
TKO	The number of cycles a node must miss to be declared dead. The value of **TKO** is automatically determined if left blank in the **Quorum Disk Configuration** dialog box.
Minimum Score	The minimum score for a node to be considered "alive". If omitted or set to 0, the default function, **`floor((`*`n`*`+1)/2)`**, is used, where *n* is the sum of the heuristics scores. The **Minimum Score** value must never exceed the sum of the heuristic scores; otherwise, the quorum disk cannot be available.
Specify physical device: By device label	Specifies the quorum disk label created by the **`mkqdisk`** utility. If this field is used, the quorum daemon reads the **`/proc/partitions`** and checks for qdisk signatures on every block device found, comparing the label against the specified label. This is useful in configurations where the quorum device name differs among nodes.
Device	The storage device the quorum daemon uses. The device must be the same on all nodes.
Label	Specifies the quorum disk label created by the **`mkqdisk`** utility. If this field contains an entry, the label overrides the **Device** field. If this field is used, the quorum daemon reads **`/proc/partitions`** and checks for qdisk signatures on every block device found, comparing the label against the specified label. This is useful in configurations where the quorum device name differs among nodes.
Heuristics	**Path to Program** — The program used to determine if this heuristic is available. This can be anything that can be executed by **`/bin/sh -c`**. A return value of 0 indicates success; anything else indicates failure. This field is required. **Interval** — The frequency (in seconds) at which the heuristic is polled. The default interval for every heuristic is 2 seconds.

Parameter	Description
	Score — The weight of this heuristic. Be careful when determining scores for heuristics. The default score for each heuristic is 1. **TKO** — The number of consecutive failures required before this heuristic is declared unavailable.
Apply	Propagates the changes to the cluster configuration file (`/etc/cluster/cluster.conf`) in each cluster node.

3.5. Configuring Fence Devices

Configuring fence devices consists of creating, updating, and deleting fence devices for the cluster. You must configure the fence devices in a cluster before you can configure fencing for the nodes in the cluster.

Creating a fence device consists of selecting a fence device type and entering parameters for that fence device (for example, name, IP address, login, and password). Updating a fence device consists of selecting an existing fence device and changing parameters for that fence device. Deleting a fence device consists of selecting an existing fence device and deleting it.

This section provides procedures for the following tasks:

- Creating fence devices — Refer to *Section 3.5.1, “Creating a Fence Device”*. Once you have created and named a fence device, you can configure the fence devices for each node in the cluster, as described in *Section 3.6, “Configuring Fencing for Cluster Members”*.
- Updating fence devices — Refer to *Section 3.5.2, “Modifying a Fence Device”*.
- Deleting fence devices — Refer to *Section 3.5.3, “Deleting a Fence Device”*.

From the cluster-specific page, you can configure fence devices for that cluster by clicking on **Fence Devices** along the top of the cluster display. This displays the fence devices for the cluster and displays the menu items for fence device configuration: **Add**, **Update**, and **Delete**. This is the starting point of each procedure described in the following sections.

Note

If this is an initial cluster configuration, no fence devices have been created, and therefore none are displayed.

Figure 3.5, “luci fence devices configuration page” shows the fence devices configuration screen before any fence devices have been created.

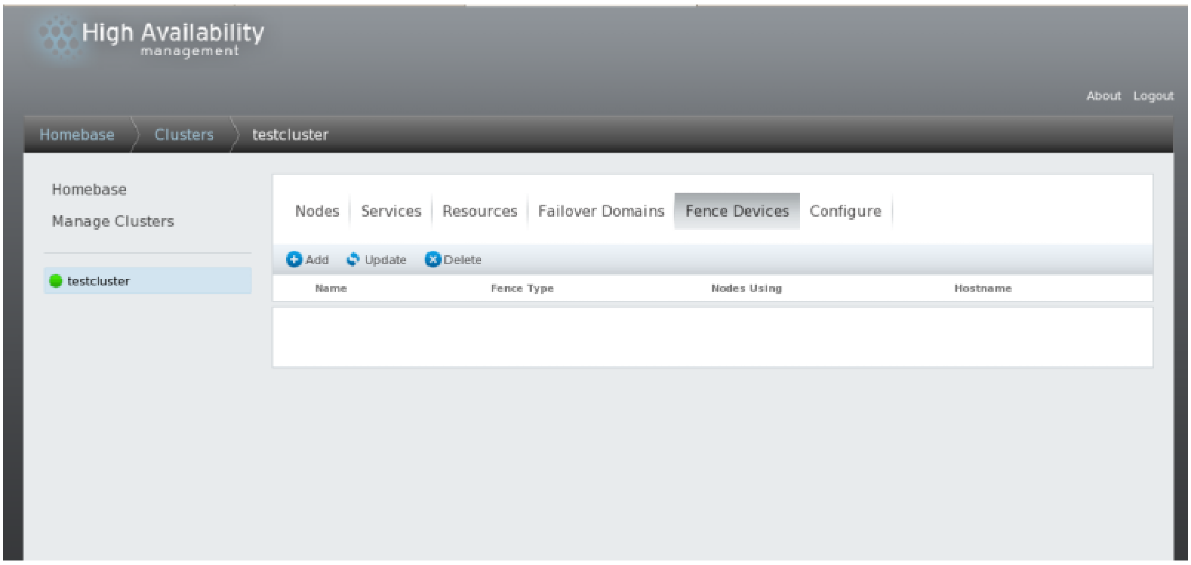


Figure 3.5. luci fence devices configuration page

3.5.1. Creating a Fence Device

To create a fence device, follow these steps:

1. From the **Fence Devices** configuration page, click **Add**. Clicking **Add** displays the **Add a Fence Instance** dialog box. From this drop-down box, select the type of fence device to configure.

2. Specify the information in the **Add a Fence Instance** dialog box according to the type of fence device. Refer to *Appendix A, Fence Device Parameters* for more information about fence device parameters. In some cases you will need to specify additional node-specific parameters for the fence device, as described in *Section 3.6, “Configuring Fencing for Cluster Members”*.

3. Click **Submit**.

4. After the fence device has been added, it appears on the **Fence Devices** configuration page.

3.5.2. Modifying a Fence Device

To modify a fence device, follow these steps:

1. From the **Fence Devices** configuration page, click on the name of the fence device to modify. This displays the dialog box for that fence device, with the values that have been configured for the device.

2. To modify the fence device, enter changes to the parameters displayed. Refer to *Appendix A, Fence Device Parameters* for more information.

3. Click **Apply** and wait for the configuration to be updated.

3.5.3. Deleting a Fence Device

To delete a fence device, follow these steps:

1. From the **Fence Devices** configuration page, click the box to the left of the fence device or devices to select the devices to delete.

2. Click **Delete** and wait for the configuration to be updated. A message appears indicating which devices are being deleted.

3. When the configuration has been updated, the deleted fence device no longer appears in the display.

3.6. Configuring Fencing for Cluster Members

Once you have completed the initial steps of creating a cluster and creating fence devices, you need to configure fencing for the cluster nodes. To configure fencing for the nodes after creating a new cluster and configuring the fencing devices for the cluster, follow the steps in this section. Note that you must configure fencing for each node in the cluster.

The following sections provide procedures for configuring a single fence device for a node, configuring a node with a backup fence device, and configuring a node with redundant power supplies:

- *Section 3.6.1, “Configuring a Single Fence Device for a Node”*
- *Section 3.6.2, “Configuring a Backup Fence Device”*
- *Section 3.6.3, “Configuring A Node with Redundant Power”*

3.6.1. Configuring a Single Fence Device for a Node

Use the following procedure to configure a node with a single fence device.

1. From the cluster-specific page, you can configure fencing for the nodes in the cluster by clicking on **Nodes** along the top of the cluster display. This displays the nodes that constitute the cluster. This is also the default page that appears when you click on the cluster name beneath **Manage Clusters** from the menu on the left side of the luci **Homebase** page.

2. Click on a node name. Clicking a link for a node causes a page to be displayed for that link showing how that node is configured.

   The node-specific page displays any services that are currently running on the node, as well as any failover domains of which this node is a member. You can modify an existing failover domain by clicking on its name. For information on configuring failover domains, see *Section 3.7, “Configuring a Failover Domain”*.

3. On the node-specific page, click **Add a fence method**.

4. Enter a name for the fencing method that you are configuring for this node.

5. Click **Submit**. This displays the node-specific screen that now displays the method you have just added under **Fence Devices**.

6. Configure a fence instance for this method by clicking **Add a Fence Instance**. This displays a drop-down menu from which you can select a fence device you have previously configured, as described in *Section 3.5.1, “Creating a Fence Device”*.

7. Select a fence device for this method. If this fence device requires that you configure node-specific parameters, the display shows the parameters to configure. For information on fencing parameters, refer to *Appendix A, Fence Device Parameters*.

   Click **Submit**. This returns you to the node-specific screen with the fence method and fence instance displayed.

3.6.2. Configuring a Backup Fence Device

You can define multiple fencing methods for a node. If fencing fails using the first method, the system will attempt to fence the node using the second method.

Use the following procedure to configure a backup fence device for a node.

1. Use the procedure provided in *Section 3.6.1, “Configuring a Single Fence Device for a Node”* to configure the primary fencing method for a node.

2. Beneath the display of the primary method you defined, click **Add a fence method**.

3. Enter a name for the backup fencing method that you are configuring for this node and click **Submit**. This displays the node-specific screen that now displays the method you have just added, below the primary fence method.

4. Configure a fence instance for this method by clicking **Add a Fence Instance**. This displays a drop-down menu from which you can select a fence device you have previously configured, as described in *Section 3.5.1, “Creating a Fence Device”*.

5. Select a fence device for this method. If this fence device requires that you configure node-specific parameters, the display shows the parameters to configure. For information on fencing parameters, refer to *Appendix A, Fence Device Parameters*.

   Click **Submit**. This returns you to the node-specific screen with the fence method and fence instance displayed.

You can continue to add fencing methods as needed. You can rearrange the order of fencing methods that will be used for this node by clicking on **Move Up** and **Move Down**.

3.6.3. Configuring A Node with Redundant Power

If your cluster is configured with redundant power supplies for your nodes, you must be sure to configure fencing so that your nodes fully shut down when they need to be fenced. If you configure each power supply as a separate fence method, each power supply will be fenced separately; the second power supply will allow the system to continue running when the first power supply is fenced and the system will not be fenced at all. To configure a system with dual power supplies, you must configure your fence devices so that both power supplies are shut off and the system is taken completely down. When configuring your system using **Conga**, this requires that you configure two instances within a single fencing method.

To configure fencing for a node with dual power supplies, follow the steps in this section.

1. Before you can configure fencing for a node with redundant power, you must configure each of the power switches as a fence device for the cluster. For information on configuring fence devices, see *Section 3.5, “Configuring Fence Devices”*.

2. From the cluster-specific page, click on **Nodes** along the top of the cluster display. This displays the nodes that constitute the cluster. This is also the default page that appears when you click on the cluster name beneath **Manage Clusters** from the menu on the left side of the luci **Homebase** page.

3. Click on a node name. Clicking a link for a node causes a page to be displayed for that link showing how that node is configured.

4. On the node-specific page, click **Add a fence method**.

5. Enter a name for the fencing method that you are configuring for this node.

6. Click **Submit**. This displays the node-specific screen that now displays the method you have just added under **Fence Devices**.

7. Configure the first power supply as a fence instance for this method by clicking **Add a Fence Instance**. This displays a drop-down menu from which you can select one of the power fencing devices you have previously configured, as described in *Section 3.5.1, “Creating a Fence Device”*.

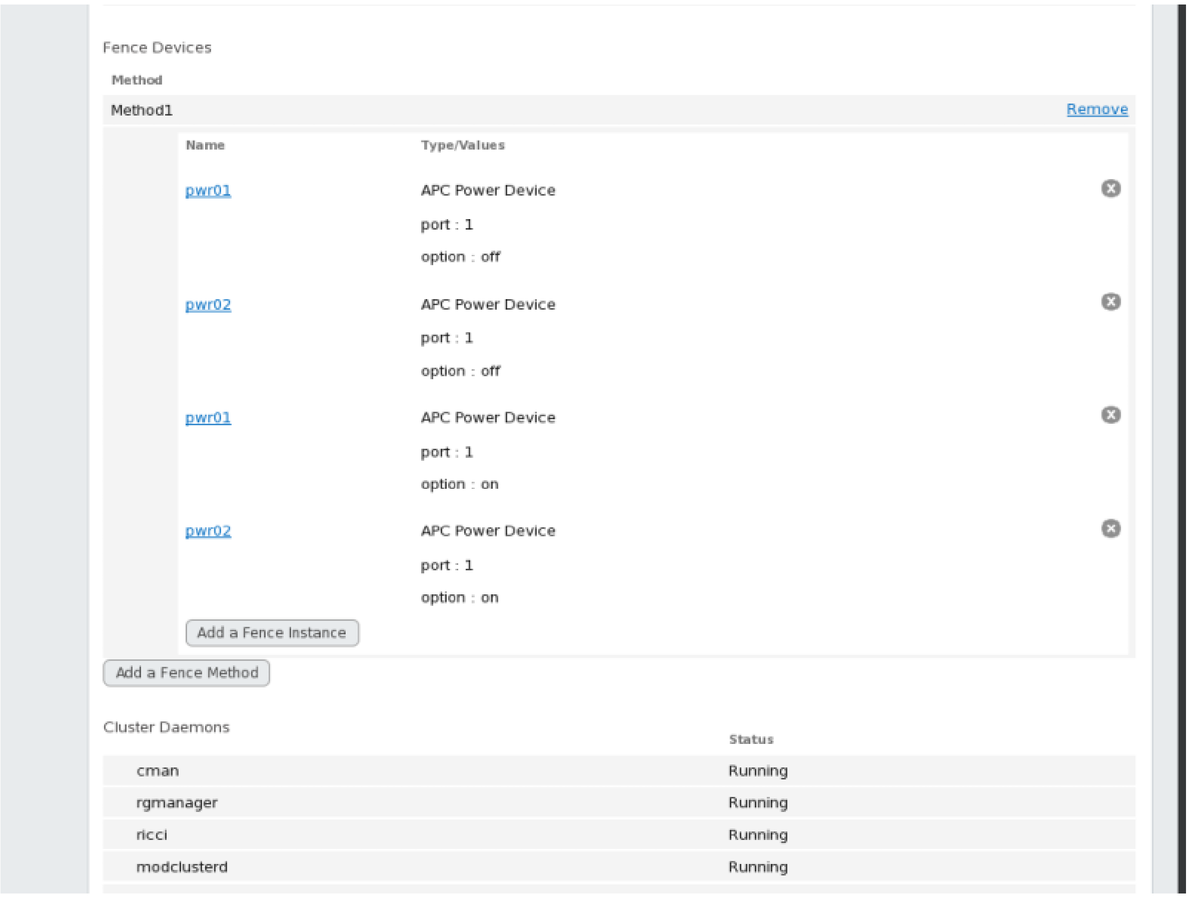
8. Select one of the power fence devices for this method and enter the appropriate parameters for this device.

9. Click **Submit**. This returns you to the node-specific screen with the fence method and fence instance displayed.

10. Under the same fence method for which you have configured the first power fencing device, click **Add a Fence Instance**. This displays a drop-down menu from which you can select the second power fencing devices you have previously configured, as described in *Section 3.5.1, “Creating a Fence Device”*.

11. Select the second of the power fence devices for this method and enter the appropriate parameters for this device.

12. Click **Submit**. This returns you to the node-specific screen with the fence methods and fence instances displayed, showing that each device will power the system off in sequence and power the system on in sequence. This is shown in *Figure 3.6, “Dual-Power Fencing Configuration”*.



Fence Devices

Method

Method1 Remove

Name	Type/Values
pwr01	APC Power Device port : 1 option : off
pwr02	APC Power Device port : 1 option : off
pwr01	APC Power Device port : 1 option : on
pwr02	APC Power Device port : 1 option : on

Cluster Daemons

	Status
cman	Running
rgmanager	Running
ricci	Running
modclusterd	Running

Figure 3.6. Dual-Power Fencing Configuration

3.7. Configuring a Failover Domain

A failover domain is a named subset of cluster nodes that are eligible to run a cluster service in the event of a node failure. A failover domain can have the following characteristics:

- Unrestricted — Allows you to specify that a subset of members are preferred, but that a cluster service assigned to this domain can run on any available member.
- Restricted — Allows you to restrict the members that can run a particular cluster service. If none of the members in a restricted failover domain are available, the cluster service cannot be started (either manually or by the cluster software).
- Unordered — When a cluster service is assigned to an unordered failover domain, the member on which the cluster service runs is chosen from the available failover domain members with no priority ordering.
- Ordered — Allows you to specify a preference order among the members of a failover domain. The member at the top of the list is the most preferred, followed by the second member in the list, and so on.
- Failback — Allows you to specify whether a service in the failover domain should fail back to the node that it was originally running on before that node failed. Configuring this characteristic is useful in circumstances where a node repeatedly fails and is part of an ordered failover domain. In that circumstance, if a node is the preferred node in a failover domain, it is possible for a service to fail over and fail back repeatedly between the preferred node and another node, causing severe impact on performance.

  Note

  The failback characteristic is applicable only if ordered failover is configured.

Note

Changing a failover domain configuration has no effect on currently running services.

Note

Failover domains are *not* required for operation.

By default, failover domains are unrestricted and unordered.

In a cluster with several members, using a restricted failover domain can minimize the work to set up the cluster to run a cluster service (such as **`httpd`**), which requires you to set up the configuration identically on all members that run the cluster service). Instead of setting up the entire cluster to run the cluster service, you must set up only the members in the restricted failover domain that you associate with the cluster service.

Note

To configure a preferred member, you can create an unrestricted failover domain comprising only one cluster member. Doing that causes a cluster service to run on that cluster member primarily (the preferred member), but allows the cluster service to fail over to any of the other members.

The following sections describe adding, modifying, and deleting a failover domain:

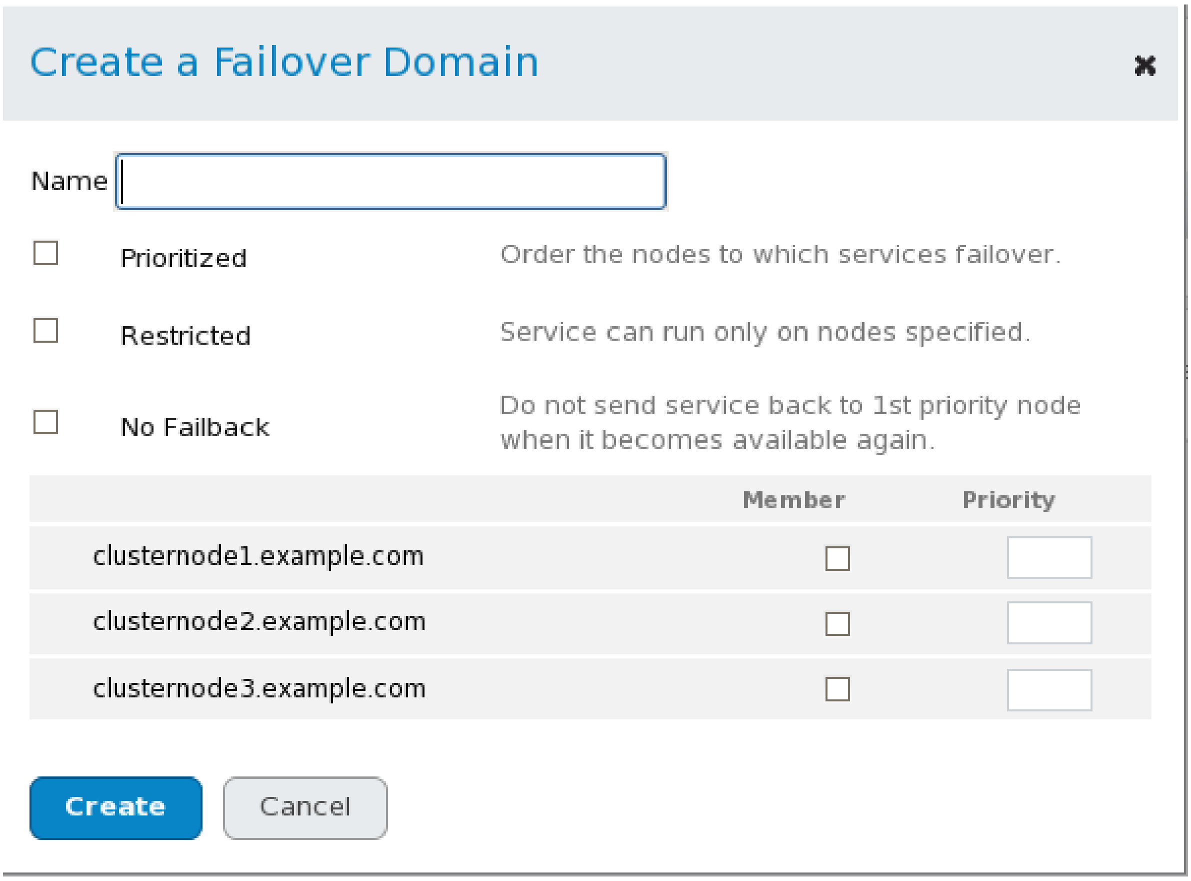
- *Section 3.7.1, “Adding a Failover Domain”*
- *Section 3.7.2, “Modifying a Failover Domain”*
- *Section 3.7.3, “Deleting a Failover Domain”*

3.7.1. Adding a Failover Domain

To add a failover domain, follow the steps in this section.

1. From the cluster-specific page, you can configure Failover Domains for that cluster by clicking on **Failover Domains** along the top of the cluster display. This displays the failover domains that have been configured for this cluster.

2. Click **Add**. Clicking **Add** causes the display of the **Create a Failover Domain** window, as shown in *Figure 3.7, “luci failover domain configuration page”*.



Create a Failover Domain

Name

□	Prioritized	Order the nodes to which services failover.
□	Restricted	Service can run only on nodes specified.
□	No Failback	Do not send service back to 1st priority node when it becomes available again.

	Member	Priority
clusternode1.example.com	□	
clusternode2.example.com	□	
clusternode3.example.com	□	

Figure 3.7. luci failover domain configuration page

3. In the **Create a Failover Domain** window, specify a failover domain name at the **Name** text box.

> **Note**
>
> The name should be descriptive enough to distinguish its purpose relative to other names used in your cluster.

4. To enable setting failover priority of the members in the failover domain, click the **Prioritized** checkbox. With **Prioritized** checked, you can set the priority value, **Priority**, for each node selected as members of the failover domain.

5. To restrict failover to members in this failover domain, click the **Restricted** checkbox. With **Restricted** checked, services assigned to this failover domain fail over only to nodes in this failover domain.

6. To specify that a node does not fail back in this failover domain, click the **No Failback** checkbox. With **No Failback** checked, if a service fails over from a preferred node, the service does not fail back to the original node once it has recovered.

7. Configure members for this failover domain. Click the **Member** checkbox for each node that is to be a member of the failover domain. If **Prioritized** is checked, set the priority in the **Priority** text box for each member of the failover domain.

8. Click **Create**. This displays the **Failover Domains** page with the newly-created failover domain displayed. A message indicates that the new domain is being created. Refresh the page for an updated status.

3.7.2. Modifying a Failover Domain

To modify a failover domain, follow the steps in this section.

1. From the cluster-specific page, you can configure Failover Domains for that cluster by clicking on **Failover Domains** along the top of the cluster display. This displays the failover domains that have been configured for this cluster.

2. Click on the name of a failover domain. This displays the configuration page for that failover domain.

3. To modify the **Prioritized**, **Restricted**, or **No Failback** properties for the failover domain, click or unclick the checkbox next to the property and click **Update Properties**.

4. To modify the failover domain membership, click or unclick the checkbox next to the cluster member. If the failover domain is prioritized, you can also modify the priority setting for the cluster member. Click **Update Settings**.

3.7.3. Deleting a Failover Domain

To delete a failover domain, follow the steps in this section.

1. From the cluster-specific page, you can configure Failover Domains for that cluster by clicking on **Failover Domains** along the top of the cluster display. This displays the failover domains that have been configured for this cluster.

2. Select the checkbox for the failover domain to delete.

3. Click on **Delete**.

3.8. Configuring Global Cluster Resources

You can configure global resources that can be used by any service running in the cluster, and you can configure resources that are available only to a specific service.

To add a global cluster resource, follow the steps in this section. You can add a resource that is local to a particular service when you configure the service, as described in *Section 3.9, “Adding a Cluster Service to the Cluster”*.

1. From the cluster-specific page, you can add resources to that cluster by clicking on **Resources** along the top of the cluster display. This displays the resources that have been configured for that cluster.
2. Click **Add**. This displays the **Add a Resource** drop-down menu.
3. Click the drop-down box under **Add a Resource** and select the type of resource to configure.
4. Enter the resource parameters for the resource you are adding. *Appendix B, HA Resource Parameters* describes resource parameters.
5. Click **Submit**. Clicking **Submit** returns to the resources page that displays the display of **Resources**, which displays the added resource (and other resources).

To modify an existing resource, perform the following steps.

1. From the **luci Resources** page, click on the name of the resource to modify. This displays the parameters for that resource.
2. Edit the resource parameters.
3. Click **Apply**.

To delete an existing resource, perform the following steps.

1. From the **luci Resources** page, click the checkbox for any resources to delete.
2. Click **Delete**.

3.9. Adding a Cluster Service to the Cluster

To add a cluster service to the cluster, follow the steps in this section.

1. From the cluster-specific page, you can add services to that cluster by clicking on **Services** along the top of the cluster display. This displays the services that have been configured for that cluster. (From the **Services** page, you can also start, restart, and disable a service, as described in *Section 4.4, “Managing High-Availability Services”*.)
2. Click **Add**. This displays the **Add a service** window.
3. On the **Add a Service** window, at the **Service name** text box, type the name of the service.

> **Note**
>
> Use a descriptive name that clearly distinguishes the service from other services in the cluster.

4. Check the **Automatically start this service** checkbox if you want the service to start automatically when a cluster is started and running. If the checkbox is *not* checked, the service must be started manually any time the cluster comes up from the stopped state.

5. Check the **Run exclusive** checkbox to set a policy wherein the service only runs on nodes that have no other services running on them.

6. If you have configured failover domains for the cluster, you can use the dropdown menu of the **Failover domain** parameter to select a failover domain for this service. For information on configuring failover domains, see *Section 3.7, “Configuring a Failover Domain”*.

7. Use the **Recovery policy** drop-down box to select a recovery policy for the service. The options are to relocate, restart, or disable the service.

   If you select **Restart** as the recovery policy for the service, you can specify the maximum number of restart failures before relocating and the length of time in seconds after which to forget a restart.

8. To add a resource to the service, click **Add a resource**. Clicking **Add a resource** causes the display of a drop-down box that allows you to add an existing Global resource or to add a new resource that is available *only* to this service.

9. To add an existing Global resource, click on the name of the existing resource from the **Add a resource to this service** drop-down box. This displays the resource and its parameters on the **Services** page for the service you are configuring. You cannot edit the parameters of a Global resource from this screen. For information on adding or modifying Global resources, see *Section 3.8, “Configuring Global Cluster Resources”*).

10. To add a new resource that is available only to this service, select the type of resource to configure from the **Add a resource** drop-down box and enter the resource parameters for the resource you are adding. *Appendix B, HA Resource Parameters* describes resource parameters.

    > **Note**
    >
    > If you are adding a Samba-service resource, connect a Samba-service resource directly to the service, *not* to a resource within a service.

11. If you want to add child resources to the resource you are defining, click **Add a child resource**. Clicking **Add a child resource** causes the display of the **Add a resource to this service** drop-down box, from which you can add an existing Global resource or add a new resource that is available only to this service. You can continue adding children resources to the resource to suit your requirements.

12. When you have completed adding resources to the service, and have completed adding children resources to resources, click **Submit**. Clicking **Submit** returns to the **Services** page displaying the added service (and other services).

Note

To verify the existence of the IP service resource used in a cluster service, you must use the **`/sbin/ip addr list`** command on a cluster node. The following output shows the **`/sbin/ip addr list`** command executed on a node running a cluster service:

```
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1356 qdisc pfifo_fast qlen 1000
    link/ether 00:05:5d:9a:d8:91 brd ff:ff:ff:ff:ff:ff
    inet 10.11.4.31/22 brd 10.11.7.255 scope global eth0
    inet6 fe80::205:5dff:fe9a:d891/64 scope link
    inet 10.11.4.240/22 scope global secondary eth0
       valid_lft forever preferred_lft forever
```

To modify an existing service, perform the following steps.

1. From the **luci Services** page, click on the name of the service to modify. This displays the parameters and resources that have been configured for that service.

2. Edit the service parameters.

3. Click **Submit**.

To delete an existing resource, perform the following steps.

1. From the **luci Services** page, click the checkbox for any services to delete.

2. Click **Delete**.

Managing Red Hat High Availability Add-On With Conga

This chapter describes various administrative tasks for managing Red Hat High Availability Add-On and consists of the following sections:

- *Section 4.1, “Adding an Existing Cluster to the luci Interface”*
- *Section 4.2, “Managing Cluster Nodes”*
- *Section 4.3, “Starting, Stopping, Restarting, and Deleting Clusters”*
- *Section 4.4, “Managing High-Availability Services”*
- *Section 4.5, “Diagnosing and Correcting Problems in a Cluster”*

4.1. Adding an Existing Cluster to the luci Interface

If you have previously created a High Availability Add-On cluster you can easily add the cluster to the **luci** interface so that you can manage the cluster with **Conga**.

To add an existing cluster to the **luci** interface, follow these steps:

1. Click **Manage Clusters** from the menu on the left side of the luci **Homebase** page. The **Clusters** screen appears.
2. Click **Add**. The **Add an Existing Cluster** screen appears.
3. Enter the node hostname and root password of any of the nodes in the existing cluster. Since each node in the cluster contains all of the configuration information for the cluster, this should provide enough information to add the cluster to the **luci** interface.
4. Click **Connect**. The **Add an Existing Cluster** screen then displays the cluster name and the remaining nodes in the cluster.
5. Enter the individual passwords for each node in the cluster, or enter one password and select **Use the same password for all nodes**.
6. Click **Add Cluster**. The previously-configured cluster now displays on the **Manage Clusters** screen.

4.2. Managing Cluster Nodes

This section documents how to perform the following node-management functions through the **luci** server component of **Conga**:

- *Section 4.2.1, “Rebooting a Cluster Node”*
- *Section 4.2.2, “Causing a Node to Leave or Join a Cluster”*
- *Section 4.2.3, “Adding a Member to a Running Cluster”*
- *Section 4.2.4, “Deleting a Member from a Cluster”*

4.2.1. Rebooting a Cluster Node

To reboot a node in a cluster, perform the following steps:

1. From the cluster-specific page, click on **Nodes** along the top of the cluster display. This displays the nodes that constitute the cluster. This is also the default page that appears when you click on the cluster name beneath **Manage Clusters** from the menu on the left side of the luci **Homebase** page.

2. Select the node to reboot by clicking the checkbox for that node.

3. Select the **Reboot** function from the menu at the top of the page. This causes the selected node to reboot and a message appears at the top of the page indicating that the node is being rebooted.

4. Refresh the page to see the updated status of the node.

It is also possible to reboot more than one node at a time by selecting all of the nodes to reboot before clicking on **Reboot**.

4.2.2. Causing a Node to Leave or Join a Cluster

You can use the **luci** server component of **Conga** to cause a node to leave an active cluster by stopping all cluster services on the node. You can also use the **luci** server component of **Conga** to cause a node that has left a cluster to rejoin the cluster.

Causing a node to leave a cluster does not remove the cluster configuration information from that node, and the node still appears in the cluster node display with a status of **`Not a cluster member`**. For information on deleting the node entirely from the cluster configuration, see *Section 4.2.4, “Deleting a Member from a Cluster”*.

To cause a node to leave a cluster, perform the following steps. This shuts down the cluster software in the node. Making a node leave a cluster prevents the node from automatically joining the cluster when it is rebooted.

1. From the cluster-specific page, click on **Nodes** along the top of the cluster display. This displays the nodes that constitute the cluster. This is also the default page that appears when you click on the cluster name beneath **Manage Clusters** from the menu on the left side of the luci **Homebase** page.

2. Select the node you want to leave the cluster by clicking the checkbox for that node.

3. Select the **Leave Cluster** function from the menu at the top of the page. This causes a message to appear at the top of the page indicating that the node is being stopped.

4. Refresh the page to see the updated status of the node.

It is also possible to cause more than one node at a time to leave the cluster by selecting all of the nodes to leave the cluster before clicking on **Leave Cluster**.

To cause a node to rejoin a cluster, select any nodes you want to have rejoin the cluster by clicking the checkbox for those nodes and selecting **Join Cluster**. This makes the selected nodes join the cluster, and allows the selected nodes to join the cluster when they are rebooted.

4.2.3. Adding a Member to a Running Cluster

To add a member to a running cluster, follow the steps in this section.

1. From the cluster-specific page, click **Nodes** along the top of the cluster display. This displays the nodes that constitute the cluster. This is also the default page that appears when you click on the cluster name beneath **Manage Clusters** from the menu on the left side of the luci **Homebase** page.

2. Click **Add**. Clicking **Add a Node** causes the display of the **Add nodes to this cluster** window.

3. Enter the node name in the **Node Hostname** text box; enter the root password in the **Root Password** text box. Check the **Enable Shared Storage Support** checkbox if clustered storage is required. If you want to add more nodes, click **Add Another Node** and enter the node name and password for the each additional node.

4. Click **Add Nodes**. Clicking **Add Nodes** causes the following actions:

   a. The cluster software packages are downloaded onto the added node.

   b. Cluster software is installed onto the added node (or it is verified that the appropriate software packages are installed)

   c. The cluster configuration file is updated and propagated to each node in the cluster — including the added node.

   d. The added node joins the cluster.

   The **Nodes** page appears with a message indicating that the node is being added to the cluster. Refresh the page to update the status.

5. When the process of adding a node is complete, click on the node name for the newly-added node to configure fencing for this node, as described in *Section 3.5, “Configuring Fence Devices”*.

4.2.4. Deleting a Member from a Cluster

To delete a member from an existing cluster that is currently in operation, follow the steps in this section.

1. From the cluster-specific page, click **Nodes** along the top of the cluster display. This displays the nodes that constitute the cluster. This is also the default page that appears when you click on the cluster name beneath **Manage Clusters** from the menu on the left side of the luci **Homebase** page.

   > **Note**
   >
   > To allow services running on a node to fail over when the node is deleted, skip the next step.

2. Disable or relocate each service that is running on the node to be deleted. For information on disabling and relocating services, see *Section 4.4, “Managing High-Availability Services”*.

3. Select the node or nodes to delete.

4. Click **Delete**. The **Nodes** page indicates that the node is being removed. Refresh the page to see the current status.

4.3. Starting, Stopping, Restarting, and Deleting Clusters

You can start, stop, and restart a cluster by performing these actions on the individual nodes in the cluster. From the cluster-specific page, click on **Nodes** along the top of the cluster display. This displays the nodes that constitute the cluster.

To stop a cluster, perform the following steps. This shuts down the cluster software in the nodes, but does not remove the cluster configuration information from the nodes and the nodes still appear in the cluster node display with a status of **`Not a cluster member`**.

1. Select all of the nodes in the cluster by clicking on the checkbox next to each node.

2. Select the **Leave Cluster** function from the menu at the top of the page. This causes a message to appear at the top of the page indicating that each node is being stopped.

3. Refresh the page to see the updated status of the nodes.

To start a cluster, perform the following steps:

1. Select all of the nodes in the cluster by clicking on the checkbox next to each node.

2. Select the **Join Cluster** function from the menu at the top of the page.

3. Refresh the page to see the updated status of the nodes.

To restart a running cluster, first stop all of the nodes in cluster, then start all of the nodes in the cluster, as described above.

To delete a cluster entirely from the **luci** interface, perform the following steps:

1. Select all of the nodes in the cluster by clicking on the checkbox next to each node.

2. Select the **Delete** function from the menu at the top of the page. This removes the cluster configuration information from the nodes and removes them from the cluster display.

3. Click on **Manage Clusters** on the left side of the screen to display the existing clusters.

4. Select the cluster to delete and click **Delete**.

4.4. Managing High-Availability Services

In addition to adding and modifying a service, as described in *Section 3.9, “Adding a Cluster Service to the Cluster”*, you can perform the following management functions for high-availability services through the **luci** server component of **Conga**:

- Start a service.

- Restart a service.

- Disable a service

- Delete a service

- Relocate a service

From the cluster-specific page, you can manage services for that cluster by clicking on **Services** along the top of the cluster display. This displays the services that have been configured for that cluster.

- Starting a service — To start any services that are not currently running, select any services you want to start by clicking the checkbox for that service and clicking **Start**.

You can also start an individual service by clicking on the name of the service on the **Services** page. This displays the service configuration page. At the top right corner of the service configuration page are the same icons for **Start**, **Restart**, **Disable**, and **Delete**. Clicking on the **Start** icon starts the service you have displayed.

- Restarting a service — To restart any services that are currently running. select any services you want to restart by clicking the checkbox for that service and clicking **Restart**.

  You can also restart an individual service by clicking on the name of the service on the **Services** page. This displays the service configuration page. At the top right corner of the service configuration page are the same icons for **Start**, **Restart**, **Disable**, and **Delete**. Clicking on the **Restart** icon starts the service you have displayed.

- **Disabling a service** — To disable any service that is currently running, select any services you want to disable by clicking the checkbox for that service and clicking **Disable**.

  You can also disable an individual service by clicking on the name of the service on the **Services** page. This displays the service configuration page. At the top right corner of the service configuration page are the same icons for **Start**, **Restart**, **Disable**, and **Delete**. Clicking on the **Disable** icon starts the service you have displayed.

- **Deleting a service** — To delete any services that are not currently running, select any services you want to disable by clicking the checkbox for that service and clicking **Delete**.

  You can also delete an individual service by clicking on the name of the service on the **Services** page. This displays the service configuration page. At the top right corner of the service configuration page are the same icons for **Start**, **Restart**, **Disable**, and **Delete**. Clicking on the **Delete** icon starts the service you have displayed.

- **Relocating a service** — To relocate a running service, click on the name of the service in the services display. This causes the services configuration page for the service to be displayed, with a display indicating on which node the service is currently running.

  From the **Start on node...** drop-down box, select the node on which you want to relocate the service, and click on the **Start** icon. A message appears at the top of the screen indicating that the service is being started. You may need to refresh the screen to see the new display indicating that the service is running on the node you have selected.

4.5. Diagnosing and Correcting Problems in a Cluster

For information about diagnosing and correcting problems in a cluster, contact an authorized Red Hat support representative.

Configuring Red Hat High Availability Add-On With Command Line Tools

This chapter describes how to configure Red Hat High Availability Add-On software by directly editing the cluster configuration file (**`/etc/cluster/cluster.conf`**) and using command-line tools. The chapter provides procedures about building a configuration file one section at a time, starting with a sample file provided in the chapter. As an alternative to starting with a sample file provided here, you could copy a skeleton configuration file from the **`cluster.conf`** man page. However, doing so would not necessarily align with information provided in subsequent procedures in this chapter. There are other ways to create and configure a cluster configuration file; this chapter provides procedures about building a configuration file one section at a time. Also, keep in mind that this is just a starting point for developing a configuration file to suit your clustering needs.

This chapter consists of the following sections:

- *Section 5.1, "Configuration Tasks"*
- *Section 5.2, "Creating a Basic Cluster Configuration File"*
- *Section 5.3, "Configuring Fencing"*
- *Section 5.4, "Configuring Failover Domains"*
- *Section 5.5, "Configuring HA Services"*
- *Section 5.6, "Verifying a Configuration"*

Important

Make sure that your deployment of High Availability Add-On meets your needs and can be supported. Consult with an authorized Red Hat representative to verify your configuration prior to deployment. In addition, allow time for a configuration burn-in period to test failure modes.

Important

This chapter references commonly used **`cluster.conf`** elements and attributes. For a comprehensive list and description of **`cluster.conf`** elements and attributes, refer to the cluster schema at **`/usr/share/cluster/cluster.rng`**, and the annotated schema at **`/usr/share/doc/cman-X.Y.ZZ/cluster_conf.html`** (for example **`/usr/share/doc/cman-3.0.12/cluster_conf.html`**).

Important

Certain procedure in this chapter call for using the **`cman_tool -r`** command to propagate a cluster configuration throughout a cluster. Using that command requires that **`ricci`** is running.

Note

Procedures in this chapter, may include specific commands for some of the command-line tools listed in *Appendix D, Command Line Tools Summary*. For more information about all commands and variables, refer to the man page for each command-line tool.

5.1. Configuration Tasks

Configuring Red Hat High Availability Add-On software with command-line tools consists of the following steps:

1. Creating a cluster. Refer to *Section 5.2, "Creating a Basic Cluster Configuration File"*.
2. Configuring fencing. Refer to *Section 5.3, "Configuring Fencing"*.
3. Configuring failover domains. Refer to *Section 5.4, "Configuring Failover Domains"*.
4. Configuring HA services. Refer to *Section 5.5, "Configuring HA Services"*.
5. Verifying a configuration. Refer to *Section 5.6, "Verifying a Configuration"*.

5.2. Creating a Basic Cluster Configuration File

Provided that cluster hardware, Red Hat Enterprise Linux, and High Availability Add-On software are installed, you can create a cluster configuration file (**`/etc/cluster/cluster.conf`**) and start running the High Availability Add-On. As a starting point only, this section describes how to create a skeleton cluster configuration file without fencing, failover domains, and HA services. Subsequent sections describe how to configure those parts of the configuration file.

Important

This is just an interim step to create a cluster configuration file; the resultant file does not have any fencing and is not considered to be a supported configuration.

The following steps describe how to create and configure a skeleton cluster configuration file. Ultimately, the configuration file for your cluster will vary according to the number of nodes, the type of fencing, the type and number of HA services, and other site-specific requirements.

1. At any node in the cluster, create **`/etc/cluster/cluster.conf`**, using the template of the example in *Example 5.1, "`cluster.conf` Sample: Basic Configuration"*.
2. **(Optional)** If you are configuring a two-node cluster, you can add the following line to the configuration file to allow a single node to maintain quorum (for example, if one node fails):

   ```
   <cman two_node="1" expected_votes="1"/>
   ```

   Refer to *Example 5.2, "`cluster.conf` Sample: Basic Two-Node Configuration"*.
3. Specify the cluster name and the configuration version number using the **`cluster`** attributes: **`name`** and **`config_version`** (refer to *Example 5.1, "`cluster.conf` Sample: Basic Configuration"* or *Example 5.2, "`cluster.conf` Sample: Basic Two-Node Configuration"*).
4. In the **`clusternodes`** section, specify the node name and the node ID of each node using the **`clusternode`** attributes: **`name`** and **`nodeid`**.

5. Save **/etc/cluster/cluster.conf**.

6. Validate the file with against the cluster schema (**cluster.rng**) by running the **ccs_config_validate** command. For example:

```
[root@example-01 ~]# ccs_config_validate
Configuration validates
```

7. Propagate the configuration file to **/etc/cluster/** in each cluster node. For example, you could propagate the file to other cluster nodes using the **scp** command.

> **Note**
>
> Propagating the cluster configuration file this way is necessary the first time a cluster is created. Once a cluster is installed and running, the cluster configuration file can be propagated using the **cman_tool version -r**. It is possible to use the **scp** command to propagate an updated configuration file; however, the cluster software must be stopped on all nodes while using the **scp** command.In addition, you should run the **ccs_config_validate** if you propagate an updated configuration file via the **scp**.

> **Note**
>
> While there are other elements and attributes present in the sample configuration file (for example, **fence** and **fencedevices**, there is no need to populate them now. Subsequent procedures in this chapter provide information about specifying other elements and attributes.

8. Start the cluster. At each cluster node run the following command:

   service cman start

   For example:

```
[root@example-01 ~]# service cman start
Starting cluster:
   Checking Network Manager...                           [  OK  ]
   Global setup...                                       [  OK  ]
   Loading kernel modules...                             [  OK  ]
   Mounting configfs...                                  [  OK  ]
   Starting cman...                                      [  OK  ]
   Waiting for quorum...                                 [  OK  ]
   Starting fenced...                                    [  OK  ]
   Starting dlm_controld...                              [  OK  ]
   Starting gfs_controld...                              [  OK  ]
   Unfencing self...                                     [  OK  ]
   Joining fence domain...                               [  OK  ]
```

9. At any cluster node, run **cman_tools nodes** to verify that the nodes are functioning as members in the cluster (signified as "M" in the status column, "Sts"). For example:

```
[root@example-01 ~]# cman_tool nodes
Node  Sts   Inc   Joined               Name
   1   M    548   2010-09-28 10:52:21  node-01.example.com
   2   M    548   2010-09-28 10:52:21  node-02.example.com
   3   M    544   2010-09-28 10:52:21  node-03.example.com
```

10. If the cluster is running, proceed to *Section 5.3, “Configuring Fencing”*.

Basic Configuration Examples

Example 5.1, “`cluster.conf` Sample: Basic Configuration” and *Example 5.2, “`cluster.conf` Sample: Basic Two-Node Configuration”* (for a two-node cluster) each provide a very basic sample cluster configuration file as a starting point. Subsequent procedures in this chapter provide information about configuring fencing and HA services.

Example 5.1. `cluster.conf` Sample: Basic Configuration

```
<cluster name="mycluster" config_version="2">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

Example 5.2. `cluster.conf` Sample: Basic Two-Node Configuration

```
<cluster name="mycluster" config_version="2">
   <cman two_node="1" expected_votes="1"/>
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
   </fencedevices>
   <rm>
   </rm>
```

```
</cluster>
```

5.3. Configuring Fencing

Configuring fencing consists of (a) specifying one or more fence devices in a cluster and (b) specifying one or more fence methods for each node (using a fence device or fence devices specified).

Based on the type of fence devices and fence methods required for your configuration, configure **`cluster.conf`** as follows:

1. In the **`fencedevices`** section, specify each fence device, using a **`fencedevice`** element and fence-device dependent attributes. *Example 5.3, "APC Fence Device Added to **`cluster.conf`** "* shows an example of a configuration file with an APC fence device added to it.

2. At the **`clusternodes`** section, within the **`fence`** element of each **`clusternode`** section, specify each fence method of the node. Specify the fence method name, using the **`method`** attribute, **`name`**. Specify the fence device for each fence method, using the **`device`** element and its attributes, **`name`** and fence-device-specific parameters. *Example 5.4, "Fence Methods Added to **`cluster.conf`** "* shows an example of a fence method with one fence device for each node in the cluster.

3. For non-power fence methods (that is, SAN/storage fencing), at the **`clusternodes`** section, add an **`unfence`** section. The **`unfence`** section does not contain **`method`** sections like the **`fence`** section does. It contains **`device`** references directly, which mirror the corresponding device sections for **`fence`**, with the notable addition of the explicit action (**`action`**) of "on" or "enable". The same **`fencedevice`** is referenced by both **`fence`** and **`unfence device`** lines, and the same per-node arguments should be repeated.

   Specifying the **`action`** attribute as "on" or "enable" enables the node when rebooted. *Example 5.4, "Fence Methods Added to **`cluster.conf`** "* and *Example 5.5, "**`cluster.conf`**: Multiple Fence Methods per Node"* include examples of the **`unfence`** elements and attributed.

   For more information about **`unfence`** refer to the **`fence_node`** man page.

4. Update the **`config_version`** attribute by incrementing its value (for example, changing from **`config_version="2"`** to **`config_version="3">`**).

5. Save **`/etc/cluster/cluster.conf`**.

6. **(Optional)** Validate the updated file against the cluster schema (**`cluster.rng`**) by running the **`ccs_config_validate`** command. For example:

   ```
   [root@example-01 ~]# ccs_config_validate
   Configuration validates
   ```

7. Run the **`cman_tool version -r`** command to propagate the configuration to the rest of the cluster nodes.

8. Verify that the updated configuration file has been propagated.

9. Proceed to *Section 5.4, "Configuring Failover Domains"*.

If required, you can configure complex configurations with multiple fence methods per node and with multiple fence devices per fence method. When specifying multiple fence methods per node, if fencing fails using the first method, **fenced**, the fence daemon, tries the next method, and continues to cycle through methods until one succeeds.

Sometimes, fencing a node requires disabling two I/O paths or two power ports. This is done by specifying two or more devices within a fence method. **fenced** runs the fence agent once for each fence-device line; all must succeed for fencing to be considered successful.

More complex configurations are shown in *the section called “Fencing Configuration Examples”* that follow.

You can find more information about configuring specific fence devices from a fence-device agent man page (for example, the man page for **fence_apc**). In addition, you can get more information about fencing parameters from *Appendix A, Fence Device Parameters*, the fence agents in **/usr/sbin/**, the cluster schema at **/usr/share/cluster/cluster.rng**, and the annotated schema at **/usr/share/doc/cman-X.Y.ZZ/cluster_conf.html** (for example, **/usr/share/doc/cman-3.0.12/cluster_conf.html**).

Fencing Configuration Examples

The following examples show a simple configuration with one fence method per node and one fence device per fence method:

- *Example 5.3, “APC Fence Device Added to **cluster.conf** ”*
- *Example 5.4, “Fence Methods Added to **cluster.conf** ”*

The following examples show more complex configurations:

- *Example 5.5, “**cluster.conf**: Multiple Fence Methods per Node”*
- *Example 5.6, “**cluster.conf**: Fencing, Multipath Multiple Ports”*
- *Example 5.7, “**cluster.conf**: Fencing Nodes with Dual Power Supplies”*

Note

The examples in this section are not exhaustive; that is, there may be other ways to configure fencing depending on your requirements.

Example 5.3. APC Fence Device Added to **cluster.conf**

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
         <fencedevice agent="fence_apc" ipaddr="apc_ip_example" login="login_example"
 name="apc" passwd="password_example"/>
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

In this example, a fence device (**fencedevice**) has been added to the **fencedevices** element, specifying the fence agent (**agent**) as **fence_apc**, the IP address (**ipaddr**) as **apc_ip_example**, the login (**login**) as **login_example**, the name of the fence device (**name**) as **apc**, and the password (**passwd**) as **password_example**.

Example 5.4. Fence Methods Added to `cluster.conf`

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC">
              <device name="apc" port="1"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="APC">
              <device name="apc" port="2"/>
            </method>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
            <method name="APC">
              <device name="apc" port="3"/>
            </method>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
         <fencedevice agent="fence_apc" ipaddr="apc_ip_example" login="login_example"
 name="apc" passwd="password_example"/>
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

In this example, a fence method (**method**) has been added to each node. The name of the fence method (**name**) for each node is **APC**. The device (**device**) for the fence method in each node specifies the name (**name**) as **apc** and a unique APC switch power port number (**port**) for each node. For example, the port number for node-01.example.com is **1** (**port="1"**). The device name for each node (**device name="apc"**) points to the fence device by the name (**name**) of **apc** in this line of the **fencedevices** element: **fencedevice agent="fence_apc" ipaddr="apc_ip_example" login="login_example" name="apc" passwd="password_example"/**.

Example 5.5. `cluster.conf`: Multiple Fence Methods per Node

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC">
              <device name="apc" port="1"/>
             </method>
            <method name="SAN">
       <device name="sanswitch1" port="11"/>
             </method>
         </fence>
         <unfence>
             <device name="sanswitch1" port="11" action="on"/>
         </unfence
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="APC">
              <device name="apc" port="2"/>
            </method>
            <method name="SAN">
       <device name="sanswitch1" port="12"/>
             </method>
         </fence>
         <unfence>
             <device name="sanswitch1" port="12" action="on"/>
         </unfence
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
            <method name="APC">
              <device name="apc" port="3"/>
            </method>
            <method name="SAN">
       <device name="sanswitch1" port="13"/>
             </method>
         </fence>
         <unfence>
             <device name="sanswitch1" port="13" action="on"/>
         </unfence
     </clusternode>
   </clusternodes>
   <fencedevices>
        <fencedevice agent="fence_apc" ipaddr="apc_ip_example" login="login_example"
 name="apc" passwd="password_example"/>
        <fencedevice agent="fence_sanbox2" ipaddr="san_ip_example"
login="login_example" name="sanswitch1 passwd="password_example"
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

Example 5.6. `cluster.conf`: Fencing, Multipath Multiple Ports

```
<cluster name="mycluster" config_version="3">
```

```
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="SAN-multi">
       <device name="sanswitch1" port="11"/>
       <device name="sanswitch2" port="11"/>
     </method>
         </fence>
         <unfence>
             <device name="sanswitch1" port="11" action="on"/>
             <device name="sanswitch2" port="11" action="on"/>
         </unfence
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="SAN-multi">
       <device name="sanswitch1" port="12"/>
       <device name="sanswitch2" port="12"/>
            </method>
         </fence>
         <unfence>
             <device name="sanswitch1" port="12" action="on"/>
             <device name="sanswitch2" port="12" action="on"/>
         </unfence
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
            <method name="SAN-multi">
       <device name="sanswitch1" port="13"/>
       <device name="sanswitch2" port="13"/>
            </method>
         </fence>
         <unfence>
             <device name="sanswitch1" port="13" action="on"/>
             <device name="sanswitch2" port="13" action="on"/>
         </unfence
     </clusternode>
   </clusternodes>
   <fencedevices>
        <fencedevice agent="fence_sanbox2" ipaddr="san_ip_example"
login="login_example" name="sanswitch1 passwd="password_example" "
        <fencedevice agent="fence_sanbox2" ipaddr="san_ip_example"
login="login_example" name="sanswitch2 passwd="password_example"
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

Example 5.7. `cluster.conf`: Fencing Nodes with Dual Power Supplies

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC-dual">
              <device name="apc1" port="1"action="off"/>
              <device name="apc2" port="1"action="off"/>
              <device name="apc1" port="1"action="on"/>
              <device name="apc2" port="1"action="on"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="APC-dual">
              <device name="apc1" port="2"action="off"/>
              <device name="apc2" port="2"action="off"/>
              <device name="apc1" port="2"action="on"/>
              <device name="apc2" port="2"action="on"/>
            </method>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
            <method name="APC-dual">
              <device name="apc1" port="3"action="off"/>
              <device name="apc2" port="3"action="off"/>
              <device name="apc1" port="3"action="on"/>
              <device name="apc2" port="3"action="on"/>
            </method>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
       <fencedevice agent="fence_apc" ipaddr="apc_ip_example" login="login_example"
 name="apc1" passwd="password_example"/>
       <fencedevice agent="fence_apc" ipaddr="apc_ip_example" login="login_example"
 name="apc2" passwd="password_example"/>
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

When using power switches to fence nodes with dual power supplies, the agents must be told to turn off both power ports before restoring power to either port. The default off-on behavior of the agent could result in the power never being fully disabled to the node.

5.4. Configuring Failover Domains

A failover domain is a named subset of cluster nodes that are eligible to run a cluster service in the event of a node failure. A failover domain can have the following characteristics:

- Unrestricted — Allows you to specify that a subset of members are preferred, but that a cluster service assigned to this domain can run on any available member.

- Restricted — Allows you to restrict the members that can run a particular cluster service. If none of the members in a restricted failover domain are available, the cluster service cannot be started (either manually or by the cluster software).

- Unordered — When a cluster service is assigned to an unordered failover domain, the member on which the cluster service runs is chosen from the available failover domain members with no priority ordering.

- Ordered — Allows you to specify a preference order among the members of a failover domain. Ordered failover domains select the node with the lowest priority number first. That is, the node in a failover domain with a priority number of "1" specifies the highest priority, and therefore is the most preferred node in a failover domain. After that node, the next preferred node would be the node with the next highest priority number, and so on.

- Failback — Allows you to specify whether a service in the failover domain should fail back to the node that it was originally running on before that node failed. Configuring this characteristic is useful in circumstances where a node repeatedly fails and is part of an ordered failover domain. In that circumstance, if a node is the preferred node in a failover domain, it is possible for a service to fail over and fail back repeatedly between the preferred node and another node, causing severe impact on performance.

  > **Note**
  >
  > The failback characteristic is applicable only if ordered failover is configured.

> **Note**
>
> Changing a failover domain configuration has no effect on currently running services.

> **Note**
>
> Failover domains are *not* required for operation.

By default, failover domains are unrestricted and unordered.

In a cluster with several members, using a restricted failover domain can minimize the work to set up the cluster to run a cluster service (such as **`httpd`**), which requires you to set up the configuration identically on all members that run the cluster service). Instead of setting up the entire cluster to run the cluster service, you must set up only the members in the restricted failover domain that you associate with the cluster service.

> **Note**
>
> To configure a preferred member, you can create an unrestricted failover domain comprising only one cluster member. Doing that causes a cluster service to run on that cluster member primarily (the preferred member), but allows the cluster service to fail over to any of the other members.

To configure a failover domain, use the following procedures:

1. Open **`/etc/cluster/cluster.conf`** at any node in the cluster.

2. Add the following skeleton section within the **`rm`** element for each failover domain to be used:

```
<failoverdomains>
    <failoverdomain name="" nofailback="" ordered="" restricted="">
        <failoverdomainnode name="" priority=""/>
        <failoverdomainnode name="" priority=""/>
        <failoverdomainnode name="" priority=""/>
    </failoverdomain>
</failoverdomains>
```

Note

The number of **failoverdomainnode** attributes depends on the number of nodes in the failover domain. The skeleton **failoverdomain** section in preceding text shows three **failoverdomainnode** elements (with no node names specified), signifying that there are three nodes in the failover domain.

3. In the **failoverdomain** section, provide the values for the elements and attributes. For descriptions of the elements and attributes, refer to the *failoverdomain* section of the annotated cluster schema. The annotated cluster schema is available at **/usr/share/doc/cman-X.Y.ZZ/cluster_conf.html** (for example **/usr/share/doc/cman-3.0.12/cluster_conf.html**) in any of the cluster nodes. For an example of a **failoverdomains** section, refer to *Example 5.8, “A Failover Domain Added to **cluster.conf** ”*.

4. Update the **config_version** attribute by incrementing its value (for example, changing from **config_version="2"** to **config_version="3">**).

5. Save **/etc/cluster/cluster.conf**.

6. **(Optional)** Validate the file with against the cluster schema (**cluster.rng**) by running the **ccs_config_validate** command. For example:

```
[root@example-01 ~]# ccs_config_validate
Configuration validates
```

7. Run the **cman_tool version -r** command to propagate the configuration to the rest of the cluster nodes.

8. Proceed to *Section 5.5, “Configuring HA Services”*.

*Example 5.8, “A Failover Domain Added to **cluster.conf** ”* shows an example of a configuration with an ordered, unrestricted failover domain.

Example 5.8. A Failover Domain Added to cluster.conf

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC">
              <device name="apc" port="1"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="APC">
              <device name="apc" port="2"/>
            </method>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
            <method name="APC">
              <device name="apc" port="3"/>
            </method>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
         <fencedevice agent="fence_apc" ipaddr="apc_ip_example" login="login_example"
 name="apc" passwd="password_example"/>
   </fencedevices>
   <rm>
       <failoverdomains>
           <failoverdomain name="example_pri" nofailback="0" ordered="1" restricted="0">
               <failoverdomainnode name="node-01.example.com" priority="1"/>
               <failoverdomainnode name="node-02.example.com" priority="2"/>
               <failoverdomainnode name="node-03.example.com" priority="3"/>
           </failoverdomain>
       </failoverdomains>
   </rm>
</cluster>
```

The **failoverdomains** section contains a **failoverdomain** section for each failover domain in the cluster. This example has one failover domain. In the **failoverdomain** line, the name (**name**) is specified as **example_pri**.In addition, it specifies no failback (**failback="0"**), that failover is ordered (**ordered="1"**), and that the failover domain is unrestricted (**restricted="0"**).

5.5. Configuring HA Services

Configuring HA (High Availability) services consists of configuring resources and assigning them to services.

The following sections describe how to edit **/etc/cluster/cluster.conf** to add resources and services.

- *Section 5.5.1, "Adding Cluster Resources"*
- *Section 5.5.2, "Adding a Cluster Service to the Cluster"*

Important

There can be a wide range of configurations possible with High Availability resources and services. For a better understanding about resource parameters and resource behavior, refer to *Appendix B, HA Resource Parameters* and *Appendix C, HA Resource Behavior*. For optimal performance and to ensure that your configuration can be supported, contact an authorized Red Hat support representative.

5.5.1. Adding Cluster Resources

You can configure two types of resources:

- Global — Resources that are available to any service in the cluster. These are configured in the **resources** section of the configuration file (within the **rm** element).
- Service-specific — Resources that are available to only one service. These are configured in each **service** section of the configuration file (within the **rm** element).

This section describes how to add a global resource. For procedures about configuring service-specific resources, refer to *Section 5.5.2, “Adding a Cluster Service to the Cluster”*.

To add a global cluster resource, follow the steps in this section.

1. Open **`/etc/cluster/cluster.conf`** at any node in the cluster.

2. Add a **`resources`** section within the **`rm`** element. For example:

```
    <rm>
        <resources>

        </resources>
    </rm>
```

3. Populate it with resources according to the services you want to create. For example, here are resources that are to be used in an Apache service. They consist of a file system (**`fs`**) resource, an IP (**`ip`**) resource, and an Apache (**`apache`**) resource.

```
    <rm>
        <resources>
           <fs name="web_fs" device="/dev/sdd2" mountpoint="/var/www" fstype="ext3"/>
           <ip address="127.143.131.100" monitor_link="on" sleeptime="10"/>
           <apache config_file="conf/httpd.conf" name="example_server" server_root="/etc/
httpd" shutdown_wait="0"/>
        </resources>
    </rm>
```

   *Example 5.9, “**`cluster.conf`** File with Resources Added ”* shows an example of a **`cluster.conf`** file with the **`resources`** section added.

4. Update the **`config_version`** attribute by incrementing its value (for example, changing from **`config_version="2"`** to **`config_version="3"`**).

5. Save **`/etc/cluster/cluster.conf`**.

6. **(Optional)** Validate the file with against the cluster schema (**`cluster.rng`**) by running the **`ccs_config_validate`** command. For example:

```
[root@example-01 ~]# ccs_config_validate
Configuration validates
```

7. Run the **`cman_tool version -r`** command to propagate the configuration to the rest of the cluster nodes.

8. Verify that the updated configuration file has been propagated.

9. Proceed to *Section 5.5.2, “Adding a Cluster Service to the Cluster”*.

Example 5.9. **`cluster.conf`** File with Resources Added

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC">
              <device name="apc" port="1"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="APC">
              <device name="apc" port="2"/>
            </method>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
            <method name="APC">
              <device name="apc" port="3"/>
            </method>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
         <fencedevice agent="fence_apc" ipaddr="apc_ip_example" login="login_example"
 name="apc" passwd="password_example"/>
   </fencedevices>
   <rm>
       <failoverdomains>
           <failoverdomain name="example_pri" nofailback="0" ordered="1" restricted="0">
               <failoverdomainnode name="node-01.example.com" priority="1"/>
               <failoverdomainnode name="node-02.example.com" priority="2"/>
               <failoverdomainnode name="node-03.example.com" priority="3"/>
           </failoverdomain>
       </failoverdomains>
       <resources>
           <fs name="web_fs" device="/dev/sdd2" mountpoint="/var/www" fstype="ext3"/>
           <ip address="127.143.131.100" monitor_link="on" sleeptime="10"/>
           <apache config_file="conf/httpd.conf" name="example_server" server_root="/etc/
httpd" shutdown_wait="0"/>
        </resources>

   </rm>
</cluster>
```

5.5.2. Adding a Cluster Service to the Cluster

To add a cluster service to the cluster, follow the steps in this section.

1. Open **`/etc/cluster/cluster.conf`** at any node in the cluster.

2. Add a **`service`** section within the **`rm`** element for each service. For example:

```
    <rm>
```

```
        <service autostart="1" domain="" exclusive="0" name="" recovery="restart">

        </service>
    </rm>
```

3. Configure the following parameters (attributes) in the **`service`** element:

   - **`autostart`** — Specifies whether to autostart the service when the cluster starts.
   - **`domain`** — Specifies a failover domain (if required).
   - **`exclusive`** — Specifies a policy wherein the service only runs on nodes that have no other services running on them.
   - **`recovery`** — Specifies a recovery policy for the service. The options are to relocate, restart, or disable the service.

4. Depending on the type of resources you want to use, populate the service with global or service-specific resources

   For example, here is an Apache service that uses global resources:

```


    <rm>
        <resources>
                <fs name="web_fs" device="/dev/sdd2" mountpoint="/var/www" fstype="ext3"/
>
                <ip address="127.143.131.100" monitor_link="on" sleeptime="10"/>
                <apache config_file="conf/httpd.conf" name="example_server"
 server_root="/etc/httpd" shutdown_wait="0"/>
        </resources>
        <service autostart="1" domain="example_pri" exclusive="0" name="example_apache"
 recovery="relocate">
                <fs ref="web_fs"/>
                <ip ref="127.143.131.100"/>
                <apache ref="example_server"/>
        </service>
    </rm>
```

   For example, here is an Apache service that uses service-specific resources:

```


    <rm>
        <service autostart="0" domain="example_pri" exclusive="0" name="example_apache2"
 recovery="relocate">
                <fs name="web_fs2" device="/dev/sdd3" mountpoint="/var/www2"
 fstype="ext3"/>
                <ip address="127.143.131.101" monitor_link="on" sleeptime="10"/>
                <apache config_file="conf/httpd.conf" name="example_server2"
 server_root="/etc/httpd" shutdown_wait="0"/>
        </service>
    </rm>
```

   *Example 5.10, “**`cluster.conf`** with Services Added: One Using Global Resources and One Using Service-Specific Resources ”* shows an example of a **`cluster.conf`** file with two services:

- **example_apache** — This service uses global resources **web_fs**, **127.143.131.100**, and **example_server**.
- **example_apache2** — This service uses service-specific resources **web_fs2**, **127.143.131.101**, and **example_server2**.

5. Update the **config_version** attribute by incrementing its value (for example, changing from **config_version="2"** to **config_version="3">**).
6. Save **/etc/cluster/cluster.conf**.
7. **(Optional)** Validate the updated file against the cluster schema (**cluster.rng**) by running the **ccs_config_validate** command. For example:

```
[root@example-01 ~]# ccs_config_validate
Configuration validates
```

8. Run the **cman_tool version -r** command to propagate the configuration to the rest of the cluster nodes.
9. Verify that the updated configuration file has been propagated.
10. Proceed to *Section 5.6, “Verifying a Configuration”*.

Example 5.10. **cluster.conf** with Services Added: One Using Global Resources and One Using Service-Specific Resources

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC">
              <device name="apc" port="1"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="APC">
              <device name="apc" port="2"/>
            </method>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
            <method name="APC">
              <device name="apc" port="3"/>
            </method>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
         <fencedevice agent="fence_apc" ipaddr="apc_ip_example" login="login_example"
 name="apc" passwd="password_example"/>
   </fencedevices>
   <rm>
       <failoverdomains>
           <failoverdomain name="example_pri" nofailback="0" ordered="1" restricted="0">
```

```
               <failoverdomainnode name="node-01.example.com" priority="1"/>
               <failoverdomainnode name="node-02.example.com" priority="2"/>
               <failoverdomainnode name="node-03.example.com" priority="3"/>
           </failoverdomain>
       </failoverdomains>
       <resources>
           <fs name="web_fs" device="/dev/sdd2" mountpoint="/var/www" fstype="ext3"/>
           <ip address="127.143.131.100" monitor_link="on" sleeptime="10"/>
           <apache config_file="conf/httpd.conf" name="example_server" server_root="/etc/
httpd" shutdown_wait="0"/>
       </resources>
       <service autostart="1" domain="example_pri" exclusive="0" name="example_apache"
 recovery="relocate">
           <fs ref="web_fs"/>
           <ip ref="127.143.131.100"/>
           <apache ref="example_server"/>
       </service>
       <service autostart="0" domain="example_pri" exclusive="0" name="example_apache2"
 recovery="relocate">
           <fs name="web_fs2" device="/dev/sdd3" mountpoint="/var/www2" fstype="ext3"/>
           <ip address="127.143.131.101" monitor_link="on" sleeptime="10"/>
           <apache config_file="conf/httpd.conf" name="example_server2" server_root="/etc/
httpd" shutdown_wait="0"/>
       </service>
   </rm>
</cluster>
```

5.6. Verifying a Configuration

Once you have created your cluster configuration file, verify that it is running correctly by performing the following steps:

1. At each node, restart the cluster software. That action ensures that any configuration additions that are checked only at startup time are included in the running configuration. You can restart the cluster software by running **`service cman restart`**. For example:

```
[root@example-01 ~]# service cman restart
Stopping cluster:
   Leaving fence domain...                                 [  OK  ]
   Stopping gfs_controld...                                [  OK  ]
   Stopping dlm_controld...                                [  OK  ]
   Stopping fenced...                                      [  OK  ]
   Stopping cman...                                        [  OK  ]
   Waiting for corosync to shutdown:                       [  OK  ]
   Unloading kernel modules...                             [  OK  ]
   Unmounting configfs...                                  [  OK  ]
Starting cluster:
   Checking Network Manager...                             [  OK  ]
   Global setup...                                         [  OK  ]
   Loading kernel modules...                               [  OK  ]
   Mounting configfs...                                    [  OK  ]
   Starting cman...                                        [  OK  ]
   Waiting for quorum...                                   [  OK  ]
   Starting fenced...                                      [  OK  ]
   Starting dlm_controld...                                [  OK  ]
   Starting gfs_controld...                                [  OK  ]
   Unfencing self...                                       [  OK  ]
   Joining fence domain...                                 [  OK  ]
```

2. Run **`service clvmd start`**, if CLVM is being used to create clustered volumes. For example:

```
[root@example-01 ~]# service clvmd start
Activating VGs:                                            [  OK  ]
```

3. Run **`service gfs2 start`**, if you are using Red Hat GFS2. For example:

```
[root@example-01 ~]# service gfs2 start
Mounting GFS2 filesystem (/mnt/gfsA):                      [  OK  ]
Mounting GFS2 filesystem (/mnt/gfsB):                      [  OK  ]
```

4. Run **`service rgmanager start`**, if you using high-availability (HA) services. For example:

```
[root@example-01 ~]# service rgmanager start
Starting Cluster Service Manager:                          [  OK  ]
```

5. At any cluster node, run **`cman_tools nodes`** to verify that the nodes are functioning as members in the cluster (signified as "M" in the status column, "Sts"). For example:

```
[root@example-01 ~]# cman_tool nodes
Node  Sts   Inc   Joined               Name
   1   M    548   2010-09-28 10:52:21  node-01.example.com
   2   M    548   2010-09-28 10:52:21  node-02.example.com
   3   M    544   2010-09-28 10:52:21  node-03.example.com
```

6. At any node, using the **`clustat`** utility, verify that the HA services are running as expected. In addition, **`clustat`** displays status of the cluster nodes. For example:

```
[root@example-01 ~]#clustat
Cluster Status for mycluster @ Wed Nov 17 05:40:00 2010
Member Status: Quorate

 Member Name                             ID   Status
 ------ ----                             ---- ------
 node-03.example.com                         3 Online, rgmanager
 node-02.example.com                         2 Online, rgmanager
 node-01.example.com                         1 Online, Local, rgmanager

 Service Name                   Owner (Last)                   State
 ------- ----                   ----- ------                   -----
 service:example_apache         node-01.example.com            started
 service:example_apache2        (none)                         disabled
```

7. If the cluster is running as expected, you are done with creating a configuration file. You can manage the cluster with command-line tools described in *Chapter 6, Managing Red Hat High Availability Add-On With Command Line Tools*.

Managing Red Hat High Availability Add-On With Command Line Tools

This chapter describes various administrative tasks for managing Red Hat High Availability Add-On and consists of the following sections:

- *Section 6.1, "Starting and Stopping the Cluster Software"*
- *Section 6.2, "Deleting or Adding a Node"*
- *Section 6.3, "Managing High-Availability Services"*
- *Section 6.4, "Updating a Configuration"*
- *Section 6.5, "Diagnosing and Correcting Problems in a Cluster"*

Important

Make sure that your deployment of Red Hat High Availability Add-On meets your needs and can be supported. Consult with an authorized Red Hat representative to verify your configuration prior to deployment. In addition, allow time for a configuration burn-in period to test failure modes.

Important

This chapter references commonly used **`cluster.conf`** elements and attributes. For a comprehensive list and description of **`cluster.conf`** elements and attributes, refer to the cluster schema at **`/usr/share/cluster/cluster.rng`**, and the annotated schema at **`/usr/share/doc/cman-X.Y.ZZ/cluster_conf.html`** (for example **`/usr/share/doc/cman-3.0.12/cluster_conf.html`**).

Important

Certain procedure in this chapter call for using the **`cman_tool -r`** command to propagate a cluster configuration throughout a cluster. Using that command requires that **`ricci`** is running.

Note

Procedures in this chapter, may include specific commands for some of the command-line tools listed in *Appendix D, Command Line Tools Summary*. For more information about all commands and variables, refer to the man page for each command-line tool.

6.1. Starting and Stopping the Cluster Software

You can start or stop cluster software on a node according to *Section 6.1.1, "Starting Cluster Software"* and *Section 6.1.2, "Stopping Cluster Software"*. Starting cluster software on a node causes it to join the cluster; stopping the cluster software on a node causes it to leave the cluster.

6.1.1. Starting Cluster Software

To start the cluster software on a node, type the following commands in this order:

1. **`service cman start`**

2. **`service clvmd start`**, if CLVM has been used to create clustered volumes

3. **`service gfs2 start`**, if you are using Red Hat GFS2

4. **`service rgmanager start`**, if you using high-availability (HA) services (**`rgmanager`**).

For example:

```
[root@example-01 ~]# service cman start
Starting cluster:
   Checking Network Manager...                             [  OK  ]
   Global setup...                                         [  OK  ]
   Loading kernel modules...                               [  OK  ]
   Mounting configfs...                                    [  OK  ]
   Starting cman...                                        [  OK  ]
   Waiting for quorum...                                   [  OK  ]
   Starting fenced...                                      [  OK  ]
   Starting dlm_controld...                                [  OK  ]
   Starting gfs_controld...                                [  OK  ]
   Unfencing self...                                       [  OK  ]
   Joining fence domain...                                 [  OK  ]
[root@example-01 ~]# service clvmd start
Starting clvmd:                                            [  OK  ]
Activating VG(s):   2 logical volume(s) in volume group "vg_example" now active
                                                           [  OK  ]
[root@example-01 ~]# service gfs2 start
Mounting GFS2 filesystem (/mnt/gfsA):                      [  OK  ]
Mounting GFS2 filesystem (/mnt/gfsB):                      [  OK  ]
[root@example-01 ~]# service rgmanager start
Starting Cluster Service Manager:                          [  OK  ]
[root@example-01 ~]#
```

6.1.2. Stopping Cluster Software

To stop the cluster software on a node, type the following commands in this order:

1. **`service rgmanager stop`**, if you using high-availability (HA) services (**`rgmanager`**).

2. **`service gfs2 stop`**, if you are using Red Hat GFS2

3. **`service clvmd stop`**, if CLVM has been used to create clustered volumes

4. **`service cman stop`**

For example:

```
[root@example-01 ~]# service rgmanager stop
Stopping Cluster Service Manager:                          [  OK  ]
[root@example-01 ~]# service gfs2 stop
Unmounting GFS2 filesystem (/mnt/gfsA):                    [  OK  ]
Unmounting GFS2 filesystem (/mnt/gfsB):                    [  OK  ]
[root@example-01 ~]# service clvmd stop
Signaling clvmd to exit                                    [  OK  ]
clvmd terminated                                           [  OK  ]
[root@example-01 ~]# service cman stop
```

```
Stopping cluster:
   Leaving fence domain...                                  [  OK  ]
   Stopping gfs_controld...                                 [  OK  ]
   Stopping dlm_controld...                                 [  OK  ]
   Stopping fenced...                                       [  OK  ]
   Stopping cman...                                         [  OK  ]
   Waiting for corosync to shutdown:                        [  OK  ]
   Unloading kernel modules...                              [  OK  ]
   Unmounting configfs...                                   [  OK  ]
[root@example-01 ~]#
```

Note

Stopping cluster software on a node causes its HA services to fail over to another node. As an alternative to that, consider relocating or migrating HA services to another node before stopping cluster software. For information about managing HA services, refer to *Section 6.3, “Managing High-Availability Services”*.

6.2. Deleting or Adding a Node

This section describes how to delete a node from a cluster and add a node to a cluster. You can delete a node from a cluster according to *Section 6.2.1, “Deleting a Node from a Cluster”*; you can add a node to a cluster according to *Section 6.2.2, “Adding a Node to a Cluster”*.

6.2.1. Deleting a Node from a Cluster

Deleting a node from a cluster consists of shutting down the cluster software on the node to be deleted and updating the cluster configuration to reflect the change.

Important

If deleting a node from the cluster causes a transition from greater than two nodes to two nodes, you must restart the cluster software at each node after updating the cluster configuration file.

To delete a node from a cluster, perform the following steps:

1. At any node, use the **`clusvcadm`** utility to relocate, migrate, or stop each HA service running on the node that is being deleted from the cluster. For information about using **`clusvcadm`**, refer to *Section 6.3, “Managing High-Availability Services”*.

2. At the node to be deleted from the cluster, stop the cluster software according to *Section 6.1.2, “Stopping Cluster Software”*. For example:

```
[root@example-01 ~]# service rgmanager stop
Stopping Cluster Service Manager:                           [  OK  ]
[root@example-01 ~]# service gfs2 stop
Unmounting GFS2 filesystem (/mnt/gfsA):                     [  OK  ]
Unmounting GFS2 filesystem (/mnt/gfsB):                     [  OK  ]
[root@example-01 ~]# service clvmd stop
Signaling clvmd to exit                                     [  OK  ]
clvmd terminated                                            [  OK  ]
[root@example-01 ~]# service cman stop
Stopping cluster:
   Leaving fence domain...                                  [  OK  ]
   Stopping gfs_controld...                                 [  OK  ]
   Stopping dlm_controld...                                 [  OK  ]
   Stopping fenced...                                       [  OK  ]
   Stopping cman...                                         [  OK  ]
   Waiting for corosync to shutdown:                        [  OK  ]
   Unloading kernel modules...                              [  OK  ]
   Unmounting configfs...                                   [  OK  ]
[root@example-01 ~]#
```

3. At any node in the cluster, edit the **`/etc/cluster/cluster.conf`** to remove the **`clusternode`** section of the node that is to be deleted. For example, in *Example 6.1, “Three-node Cluster Configuration”*, if node-03.example.com is supposed to be removed, then delete the **`clusternode`** section for that node. If removing a node (or nodes) causes the cluster to be a two-node cluster, you can add the following line to the configuration file to allow a single node to maintain quorum (for example, if one node fails):

   `<cman two_node="1" expected_votes="1"/>`

   Refer to *Section 6.2.3, “Examples of Three-Node and Two-Node Configurations”* for comparison between a three-node and a two-node configuration.

4. Update the **`config_version`** attribute by incrementing its value (for example, changing from **`config_version="2"`** to **`config_version="3">`**).

5. Save **`/etc/cluster/cluster.conf`**.

6. **(Optional)** Validate the updated file against the cluster schema (**`cluster.rng`**) by running the **`ccs_config_validate`** command. For example:

```
[root@example-01 ~]# ccs_config_validate
Configuration validates
```

7. Run the **`cman_tool version -r`** command to propagate the configuration to the rest of the cluster nodes.

8. Verify that the updated configuration file has been propagated.

9. If the node count of the cluster has transitioned from greater than two nodes to two nodes, you must restart the cluster software as follows:

   a. At each node, stop the cluster software according to *Section 6.1.2, “Stopping Cluster Software”*. For example:

```
[root@example-01 ~]# service rgmanager stop
Stopping Cluster Service Manager:                           [  OK  ]
[root@example-01 ~]# service gfs2 stop
Unmounting GFS2 filesystem (/mnt/gfsA):                     [  OK  ]
Unmounting GFS2 filesystem (/mnt/gfsB):                     [  OK  ]
[root@example-01 ~]# service clvmd stop
Signaling clvmd to exit                                     [  OK  ]
clvmd terminated                                            [  OK  ]
[root@example-01 ~]# service cman stop
Stopping cluster:
   Leaving fence domain...                                  [  OK  ]
   Stopping gfs_controld...                                 [  OK  ]
   Stopping dlm_controld...                                 [  OK  ]
   Stopping fenced...                                       [  OK  ]
   Stopping cman...                                        [  OK  ]
   Waiting for corosync to shutdown:                       [  OK  ]
   Unloading kernel modules...                             [  OK  ]
   Unmounting configfs...                                  [  OK  ]
[root@example-01 ~]#
```

b. At each node, start the cluster software according to *Section 6.1.1, “Starting Cluster Software”*. For example:

```
[root@example-01 ~]# service cman start
Starting cluster:
   Checking Network Manager...                             [  OK  ]
   Global setup...                                         [  OK  ]
   Loading kernel modules...                               [  OK  ]
   Mounting configfs...                                    [  OK  ]
   Starting cman...                                        [  OK  ]
   Waiting for quorum...                                   [  OK  ]
   Starting fenced...                                      [  OK  ]
   Starting dlm_controld...                                [  OK  ]
   Starting gfs_controld...                                [  OK  ]
   Unfencing self...                                       [  OK  ]
   Joining fence domain...                                 [  OK  ]
[root@example-01 ~]# service clvmd start
Starting clvmd:                                            [  OK  ]
Activating VG(s):   2 logical volume(s) in volume group "vg_example" now active
                                                           [  OK  ]
[root@example-01 ~]# service gfs2 start
Mounting GFS2 filesystem (/mnt/gfsA):                      [  OK  ]
Mounting GFS2 filesystem (/mnt/gfsB):                      [  OK  ]
[root@example-01 ~]# service rgmanager start
Starting Cluster Service Manager:                          [  OK  ]
[root@example-01 ~]#
```

c. At any cluster node, run **cman_tools nodes** to verify that the nodes are functioning as members in the cluster (signified as "M" in the status column, "Sts"). For example:

```
[root@example-01 ~]# cman_tool nodes
Node  Sts   Inc   Joined               Name
   1   M    548   2010-09-28 10:52:21  node-01.example.com
   2   M    548   2010-09-28 10:52:21  node-02.example.com
```

d. At any node, using the **clustat** utility, verify that the HA services are running as expected. In addition, **clustat** displays status of the cluster nodes. For example:

```
[root@example-01 ~]#clustat
Cluster Status for mycluster @ Wed Nov 17 05:40:00 2010
Member Status: Quorate

 Member Name                             ID   Status
 ------ ----                             ---- ------
 node-02.example.com                         2 Online, rgmanager
 node-01.example.com                         1 Online, Local, rgmanager

 Service Name                   Owner (Last)                   State
 ------- ----                   ----- ------                   -----
 service:example_apache         node-01.example.com            started
 service:example_apache2        (none)                         disabled
```

6.2.2. Adding a Node to a Cluster

Adding a node to a cluster consists of updating the cluster configuration, propagating the updated configuration to the node to be added, and starting the cluster software on that node. To add a node to a cluster, perform the following steps:

1. At any node in the cluster, edit the **`/etc/cluster/cluster.conf`** to add a **`clusternode`** section for the node that is to be added. For example, in *Example 6.2, "Two-node Cluster Configuration"*, if node-03.example.com is supposed to be added, then add a **`clusternode`** section for that node. If adding a node (or nodes) causes the cluster to transition from a two-node cluster to a cluster with three or more nodes, remove the following **`cman`** attributes from **`/etc/cluster/cluster.conf`**:

   - **`cman two_node="1"`**
   - **`expected_votes="1"`**

   Refer to *Section 6.2.3, "Examples of Three-Node and Two-Node Configurations"* for comparison between a three-node and a two-node configuration.

2. Update the **`config_version`** attribute by incrementing its value (for example, changing from **`config_version="2"`** to **`config_version="3">`**).

3. Save **`/etc/cluster/cluster.conf`**.

4. **(Optional)** Validate the updated file against the cluster schema (**`cluster.rng`**) by running the **`ccs_config_validate`** command. For example:

```
[root@example-01 ~]# ccs_config_validate
Configuration validates
```

5. Run the **`cman_tool version -r`** command to propagate the configuration to the rest of the cluster nodes.

6. Verify that the updated configuration file has been propagated.

7. Propagate the updated configuration file to **`/etc/cluster/`** in each node to be added to the cluster. For example, use the **`scp`** command to send the updated configuration file to each node to be added to the cluster.

8. If the node count of the cluster has transitioned from two nodes to greater than two nodes, you must restart the cluster software in the existing cluster nodes as follows:

   a. At each node, stop the cluster software according to *Section 6.1.2, "Stopping Cluster Software"*. For example:

```
[root@example-01 ~]# service rgmanager stop
Stopping Cluster Service Manager:                          [  OK  ]
[root@example-01 ~]# service gfs2 stop
Unmounting GFS2 filesystem (/mnt/gfsA):                    [  OK  ]
Unmounting GFS2 filesystem (/mnt/gfsB):                    [  OK  ]
[root@example-01 ~]# service clvmd stop
Signaling clvmd to exit                                    [  OK  ]
clvmd terminated                                           [  OK  ]
[root@example-01 ~]# service cman stop
Stopping cluster:
   Leaving fence domain...                                 [  OK  ]
   Stopping gfs_controld...                                 [  OK  ]
   Stopping dlm_controld...                                 [  OK  ]
   Stopping fenced...                                       [  OK  ]
   Stopping cman...                                         [  OK  ]
   Waiting for corosync to shutdown:                        [  OK  ]
   Unloading kernel modules...                              [  OK  ]
   Unmounting configfs...                                   [  OK  ]
[root@example-01 ~]#
```

b. At each node, start the cluster software according to *Section 6.1.1, “Starting Cluster Software”*. For example:

```
[root@example-01 ~]# service cman start
Starting cluster:
   Checking Network Manager...                              [  OK  ]
   Global setup...                                          [  OK  ]
   Loading kernel modules...                                [  OK  ]
   Mounting configfs...                                     [  OK  ]
   Starting cman...                                         [  OK  ]
   Waiting for quorum...                                    [  OK  ]
   Starting fenced...                                       [  OK  ]
   Starting dlm_controld...                                 [  OK  ]
   Starting gfs_controld...                                 [  OK  ]
   Unfencing self...                                        [  OK  ]
   Joining fence domain...                                  [  OK  ]
[root@example-01 ~]# service clvmd start
Starting clvmd:                                             [  OK  ]
Activating VG(s):   2 logical volume(s) in volume group "vg_example" now active
                                                            [  OK  ]
[root@example-01 ~]# service gfs2 start
Mounting GFS2 filesystem (/mnt/gfsA):                       [  OK  ]
Mounting GFS2 filesystem (/mnt/gfsB):                       [  OK  ]
[root@example-01 ~]# service rgmanager start
Starting Cluster Service Manager:                           [  OK  ]
[root@example-01 ~]#
```

9. At each node to be added to the cluster, start the cluster software according to *Section 6.1.1, “Starting Cluster Software”*. For example:

```
[root@example-01 ~]# service cman start
Starting cluster:
   Checking Network Manager...                              [  OK  ]
   Global setup...                                          [  OK  ]
   Loading kernel modules...                                [  OK  ]
   Mounting configfs...                                     [  OK  ]
   Starting cman...                                         [  OK  ]
   Waiting for quorum...                                    [  OK  ]
   Starting fenced...                                       [  OK  ]
   Starting dlm_controld...                                 [  OK  ]
   Starting gfs_controld...                                 [  OK  ]
   Unfencing self...                                        [  OK  ]
   Joining fence domain...                                  [  OK  ]
[root@example-01 ~]# service clvmd start
Starting clvmd:                                             [  OK  ]
Activating VG(s):   2 logical volume(s) in volume group "vg_example" now active
                                                            [  OK  ]
[root@example-01 ~]# service gfs2 start
Mounting GFS2 filesystem (/mnt/gfsA):                       [  OK  ]
Mounting GFS2 filesystem (/mnt/gfsB):                       [  OK  ]

[root@example-01 ~]# service rgmanager start
```

```
Starting Cluster Service Manager:                          [  OK  ]
[root@example-01 ~]#
```

10. At any node, using the **clustat** utility, verify that each added node is running and part of the cluster. For example:

    ```
    [root@example-01 ~]#clustat
    Cluster Status for mycluster @ Wed Nov 17 05:40:00 2010
    Member Status: Quorate

     Member Name                             ID   Status
     ------ ----                             ---- ------
     node-03.example.com                         3 Online, rgmanager
     node-02.example.com                         2 Online, rgmanager
     node-01.example.com                         1 Online, Local, rgmanager

     Service Name                   Owner (Last)                   State
     ------- ----                   ----- ------                   -----
     service:example_apache         node-01.example.com            started
     service:example_apache2        (none)                         disabled
    ```

    For information about using **clustat**, refer to *Section 6.3, “Managing High-Availability Services”*.

    In addition, you can use **cman_tool status** to verify node votes, node count, and quorum count. For example:

    ```
    [root@example-01 ~]#cman_tool status
    Version: 6.2.0
    Config Version: 19
    Cluster Name: mycluster
    Cluster Id: 3794
    Cluster Member: Yes
    Cluster Generation: 548
    Membership state: Cluster-Member
    Nodes: 3
    Expected votes: 3
    Total votes: 3
    Node votes: 1
    Quorum: 2
    Active subsystems: 9
    Flags:
    Ports Bound: 0 11 177
    Node name: node-01.example.com
    Node ID: 3
    Multicast addresses: 239.192.14.224
    Node addresses: 10.15.90.58
    ```

11. At any node, you can use the **clusvcadm** utility to migrate or relocate a running service to the newly joined node. Also, you can enable any disabled services. For information about using **clusvcadm**, refer to *Section 6.3, “Managing High-Availability Services”*

6.2.3. Examples of Three-Node and Two-Node Configurations

Refer to the examples that follow for comparison between a three-node and a two-node configuration.

> Example 6.1. Three-node Cluster Configuration

```
<cluster name="mycluster" config_version="3">
   <cman/>
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC">
              <device name="apc" port="1"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="APC">
              <device name="apc" port="2"/>
            </method>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
            <method name="APC">
              <device name="apc" port="3"/>
            </method>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
         <fencedevice agent="fence_apc" ipaddr="apc_ip_example" login="login_example"
 name="apc" passwd="password_example"/>
   </fencedevices>
   <rm>
       <failoverdomains>
           <failoverdomain name="example_pri" nofailback="0" ordered="1" restricted="0">
               <failoverdomainnode name="node-01.example.com" priority="1"/>
               <failoverdomainnode name="node-02.example.com" priority="2"/>
               <failoverdomainnode name="node-03.example.com" priority="3"/>
           </failoverdomain>
       </failoverdomains>
       <resources>
           <fs name="web_fs" device="/dev/sdd2" mountpoint="/var/www" fstype="ext3"/>
           <ip address="127.143.131.100" monitor_link="on" sleeptime="10"/>
           <apache config_file="conf/httpd.conf" name="example_server" server_root="/etc/
httpd" shutdown_wait="0"/>
       </resources>
       <service autostart="0" domain="example_pri" exclusive="0" name="example_apache"
 recovery="relocate">
           <fs ref="web_fs"/>
           <ip ref="127.143.131.100"/>
           <apache ref="example_server"/>
       </service>
       <service autostart="0" domain="example_pri" exclusive="0" name="example_apache2"
 recovery="relocate">
           <fs name="web_fs2" device="/dev/sdd3" mountpoint="/var/www" fstype="ext3"/>
           <ip address="127.143.131.101" monitor_link="on" sleeptime="10"/>
           <apache config_file="conf/httpd.conf" name="example_server2" server_root="/etc/
httpd" shutdown_wait="0"/>
       </service>
   </rm>
</cluster>
```

Example 6.2. Two-node Cluster Configuration

```
<cluster name="mycluster" config_version="3">
   <cman two_node="1" expected_votes="1"/>
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC">
              <device name="apc" port="1"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="APC">
              <device name="apc" port="2"/>
            </method>
         </fence>
   </clusternodes>
   <fencedevices>
         <fencedevice agent="fence_apc" ipaddr="apc_ip_example" login="login_example"
 name="apc" passwd="password_example"/>
   </fencedevices>
   <rm>
       <failoverdomains>
           <failoverdomain name="example_pri" nofailback="0" ordered="1" restricted="0">
               <failoverdomainnode name="node-01.example.com" priority="1"/>
               <failoverdomainnode name="node-02.example.com" priority="2"/>
               <failoverdomainnode name="node-03.example.com" priority="3"/>
           </failoverdomain>
       </failoverdomains>
       <resources>
           <fs name="web_fs" device="/dev/sdd2" mountpoint="/var/www" fstype="ext3"/>
           <ip address="127.143.131.100" monitor_link="on" sleeptime="10"/>
           <apache config_file="conf/httpd.conf" name="example_server" server_root="/etc/
httpd" shutdown_wait="0"/>
       </resources>
       <service autostart="0" domain="example_pri" exclusive="0" name="example_apache"
 recovery="relocate">
           <fs ref="web_fs"/>
           <ip ref="127.143.131.100"/>
           <apache ref="example_server"/>
       </service>
       <service autostart="0" domain="example_pri" exclusive="0" name="example_apache2"
 recovery="relocate">
           <fs name="web_fs2" device="/dev/sdd3" mountpoint="/var/www" fstype="ext3"/>
           <ip address="127.143.131.101" monitor_link="on" sleeptime="10"/>
           <apache config_file="conf/httpd.conf" name="example_server2" server_root="/etc/
httpd" shutdown_wait="0"/>
       </service>
   </rm>
</cluster>
```

6.3. Managing High-Availability Services

You can manage high-availability services using the **Cluster Status Utility**, **`clustat`**, and the **Cluster User Service Administration Utility**, **`clusvcadm`**. **`clustat`** displays the status of a cluster and **`clusvcadm`** provides the means to manage high-availability services.

This section provides basic information about managing HA services using **`clustat`** and **`clusvcadm`** It consists of the following subsections:

- *Section 6.3.1, “Displaying HA Service Status with **`clustat`**”*
- *Section 6.3.2, “Managing HA Services with **`clusvcadm`**”*

6.3.1. Displaying HA Service Status with `clustat`

`clustat` displays cluster-wide status. It shows membership information, quorum view, the state of all high-availability services, and indicates which node the **`clustat`** command is being run at (Local). *Table 6.1, “Services Status”* describes the states that services can be in and are displayed when running **`clustat`**. *Example 6.3, “**`clustat`** Display”* shows an example of a **`clustat`** display. For more detailed information about running the **`clustat`** command refer to the **`clustat`** man page.

Table 6.1. Services Status

Services Status	Description
Started	The service resources are configured and available on the cluster system that owns the service.
Recovering	The service is pending start on another node.
Disabled	The service has been disabled, and does not have an assigned owner. A disabled service is never restarted automatically by the cluster.
Stopped	In the stopped state, the service will be evaluated for starting after the next service or node transition. This is a temporary state. You may disable or enable the service from this state.
Failed	The service is presumed dead. A service is placed into this state whenever a resource's *stop* operation fails. After a service is placed into this state, you must verify that there are no resources allocated (mounted file systems, for example) prior to issuing a **`disable`** request. The only operation that can take place when a service has entered this state is **`disable`**.
Uninitialized	This state can appear in certain cases during startup and running **`clustat -f`**.

Example 6.3. **`clustat`** Display

```
[root@example-01 ~]#clustat
Cluster Status for mycluster @ Wed Nov 17 05:40:15 2010
Member Status: Quorate

 Member Name                             ID   Status
 ------ ----                             ---- ------
 node-03.example.com                         3 Online, rgmanager
 node-02.example.com                         2 Online, rgmanager
 node-01.example.com                         1 Online, Local, rgmanager

 Service Name                   Owner (Last)                   State
 ------- ----                   ----- ------                   -----
 service:example_apache         node-01.example.com            started
 service:example_apache2        (none)                         disabled
```

6.3.2. Managing HA Services with `clusvcadm`

You can manage HA services using the **`clusvcadm`** command. With it you can perform the following operations:

- Enable and start a service.
- Disable a service.
- Stop a service.

- Freeze a service
- Unfreeze a service
- Migrate a service (for virtual machine services only)
- Relocate a service.
- Restart a service.

Table 6.2, "Service Operations" describes the operations in more detail. For a complete description on how do perform those operations, refer to the **clusvcadm** utility man page.

Table 6.2. Service Operations

Service Operation	Description	Command Syntax
Enable	Start the service, optionally on a preferred target and optionally according to failover domain rules. In absence of either, the local host where **clusvcadm** is run will start the service. If the original *start* fails, the service behaves as though a *relocate* operation was requested (refer to **Relocate** in this table). If the operation succeeds, the service is placed in the started state.	**clusvcadm -e <service_name>** or **clusvcadm -e <service_name> -m <member>** (Using the -m option specifies the preferred target member on which to start the service.)
Disable	Stop the service and place into the disabled state. This is the only permissible operation when a service is in the *failed* state.	**clusvcadm -d <service_name>**
Relocate	Move the service to another node. Optionally, you may specify a preferred node to receive the service, but the inability of the service to run on that host (for example, if the service fails to start or the host is offline) does not prevent relocation, and another node is chosen. **rgmanager** attempts to start the service on every permissible node in the cluster. If no permissible target node in the cluster successfully starts the service, the relocation fails and the service is attempted to be restarted on the original owner. If the original owner cannot restart the service, the service is placed in the *stopped* state.	**clusvcadm -r <service_name>** or **clusvcadm -r <service_name> -m <member>** (Using the -m option specifies the preferred target member on which to start the service.)
Stop	Stop the service and place into the *stopped* state.	**clusvcadm -s <service_name>**
Freeze	Freeze a service on the node where it is currently running. This prevents status checks of the service as well as failover in the event the node fails or rgmanager is stopped. This can be used to suspend	**clusvcadm -Z <service_name>**

Service Operation	Description	Command Syntax
	a service to allow maintenance of underlying resources. Refer to *the section called “Considerations for Using the Freeze and Unfreeze Operations”* for important information about using the *freeze* and *unfreeze* operations.	
Unfreeze	Unfreeze takes a service out of the *freeze* state. This re-enables status checks. Refer to *the section called “Considerations for Using the Freeze and Unfreeze Operations”* for important information about using the *freeze* and *unfreeze* operations.	`clusvcadm -U <service_name>`
Migrate	Migrate a virtual machine to another node. You must specify a target node. Depending on the failure, a failure to migrate may result with the virtual machine in the *failed* state or in the started state on the original owner.	`clusvcadm -M <service_name> -m <member>` **Important** For the *migrate* operation, you *must specify* a target node using the `-m <member>` option.
Restart	Restart a service on the node where it is currently running.	`clusvcadm -R <service_name>`

Considerations for Using the Freeze and Unfreeze Operations

Using the *freeze* operation allows maintenance of parts of **rgmanager** services. For example, if you have a database and a web server in one **rgmanager** service, you may freeze the **rgmanager** service, stop the database, perform maintenance, restart the database, and unfreeze the service.

When a service is frozen, it behaves as follows:

- *Status* checks are disabled.
- *Start* operations are disabled.
- *Stop* operations are disabled.
- Failover will not occur (even if you power off the service owner).

Important

Failure to follow these guidelines may result in resources being allocated on multiple hosts:

- You *must not* stop all instances of rgmanager when a service is frozen unless you plan to reboot the hosts prior to restarting rgmanager.
- You *must not* unfreeze a service until the reported owner of the service rejoins the cluster and restarts rgmanager.

6.4. Updating a Configuration

Updating the cluster configuration consists of editing the cluster configuration file (**`/etc/cluster/cluster.conf`**) and propagating it to each node in the cluster. You can update the configuration using either of the following procedures:

- *Section 6.4.1, "Updating a Configuration Using **`cman_tool version -r`**"*
- *Section 6.4.2, "Updating a Configuration Using **`scp`**"*

6.4.1. Updating a Configuration Using `cman_tool version -r`

To update the configuration using the **`cman_tool version -r`** command, perform the following steps:

1. At any node in the cluster, edit the **`/etc/cluster/cluster.conf`** file.
2. Update the **`config_version`** attribute by incrementing its value (for example, changing from **`config_version="2"`** to **`config_version="3">`**).
3. Save **`/etc/cluster/cluster.conf`**.
4. Run the **`cman_tool version -r`** command to propagate the configuration to the rest of the cluster nodes.
5. Verify that the updated configuration file has been propagated.
6. You may skip this step (restarting cluster software) if you have made only the following configuration changes:
   - Deleting a node from the cluster configuration—*except* where the node count changes from greater than two nodes to two nodes. For information about deleting a node from a cluster and transitioning from greater than two nodes to two nodes, refer to *Section 6.2, "Deleting or Adding a Node"*.
   - Adding a node to the cluster configuration—*except* where the node count changes from two nodes to greater than two nodes. For information about adding a node to a cluster and transitioning from two nodes tp greater than two nodes, refer to *Section 6.2.2, "Adding a Node to a Cluster"*.
   - Changes to how daemons log information.
   - HA service/VM maintenance (adding, editing, or deleting).
   - Resource maintenance (adding, editing, or deleting).
   - Failover domain maintenance (adding, editing, or deleting).

Otherwise, you must restart the cluster software as follows:

a. At each node, stop the cluster software according to *Section 6.1.2, “Stopping Cluster Software”*. For example:

```
[root@example-01 ~]# service rgmanager stop
Stopping Cluster Service Manager:                          [  OK  ]
[root@example-01 ~]# service gfs2 stop
Unmounting GFS2 filesystem (/mnt/gfsA):                    [  OK  ]
Unmounting GFS2 filesystem (/mnt/gfsB):                    [  OK  ]
[root@example-01 ~]# service clvmd stop
Signaling clvmd to exit                                    [  OK  ]
clvmd terminated                                           [  OK  ]
[root@example-01 ~]# service cman stop
Stopping cluster:
   Leaving fence domain...                                 [  OK  ]
   Stopping gfs_controld...                                [  OK  ]
   Stopping dlm_controld...                                [  OK  ]
   Stopping fenced...                                      [  OK  ]
   Stopping cman...                                        [  OK  ]
   Waiting for corosync to shutdown:                       [  OK  ]
   Unloading kernel modules...                             [  OK  ]
   Unmounting configfs...                                  [  OK  ]
[root@example-01 ~]#
```

b. At each node, start the cluster software according to *Section 6.1.1, “Starting Cluster Software”*. For example:

```
[root@example-01 ~]# service cman start
Starting cluster:
   Checking Network Manager...                             [  OK  ]
   Global setup...                                         [  OK  ]
   Loading kernel modules...                               [  OK  ]
   Mounting configfs...                                    [  OK  ]
   Starting cman...                                        [  OK  ]
   Waiting for quorum...                                   [  OK  ]
   Starting fenced...                                      [  OK  ]
   Starting dlm_controld...                                [  OK  ]
   Starting gfs_controld...                                [  OK  ]
   Unfencing self...                                       [  OK  ]
   Joining fence domain...                                 [  OK  ]
[root@example-01 ~]# service clvmd start
Starting clvmd:                                            [  OK  ]
Activating VG(s):   2 logical volume(s) in volume group "vg_example" now active
                                                           [  OK  ]
[root@example-01 ~]# service gfs2 start
Mounting GFS2 filesystem (/mnt/gfsA):                      [  OK  ]
Mounting GFS2 filesystem (/mnt/gfsB):                      [  OK  ]
[root@example-01 ~]# service rgmanager start
Starting Cluster Service Manager:                          [  OK  ]
[root@example-01 ~]#
```

Stopping and starting the cluster software ensures that any configuration changes that are checked only at startup time are included in the running configuration.

7. At any cluster node, run **cman_tools nodes** to verify that the nodes are functioning as members in the cluster (signified as "M" in the status column, "Sts"). For example:

```
[root@example-01 ~]# cman_tool nodes
Node  Sts   Inc   Joined               Name
   1   M    548   2010-09-28 10:52:21  node-01.example.com
   2   M    548   2010-09-28 10:52:21  node-02.example.com
   3   M    544   2010-09-28 10:52:21  node-03.example.com
```

8. At any node, using the **clustat** utility, verify that the HA services are running as expected. In addition, **clustat** displays status of the cluster nodes. For example:

```
[root@example-01 ~]#clustat
Cluster Status for mycluster @ Wed Nov 17 05:40:00 2010
Member Status: Quorate

 Member Name                             ID   Status
 ------ ----                             ---- ------
 node-03.example.com                         3 Online, rgmanager
 node-02.example.com                         2 Online, rgmanager
 node-01.example.com                         1 Online, Local, rgmanager

 Service Name                   Owner (Last)                   State
 ------- ----                   ----- ------                   -----
 service:example_apache         node-01.example.com            started
 service:example_apache2        (none)                         disabled
```

9. If the cluster is running as expected, you are done updating the configuration.

6.4.2. Updating a Configuration Using scp

To update the configuration using the **scp** command, perform the following steps:

1. At each node, stop the cluster software according to *Section 6.1.2, “Stopping Cluster Software”*. For example:

```
[root@example-01 ~]# service rgmanager stop
Stopping Cluster Service Manager:                          [  OK  ]
[root@example-01 ~]# service gfs2 stop
Unmounting GFS2 filesystem (/mnt/gfsA):                    [  OK  ]
Unmounting GFS2 filesystem (/mnt/gfsB):                    [  OK  ]
[root@example-01 ~]# service clvmd stop
Signaling clvmd to exit                                    [  OK  ]
clvmd terminated                                           [  OK  ]
[root@example-01 ~]# service cman stop
Stopping cluster:
   Leaving fence domain...                                 [  OK  ]
   Stopping gfs_controld...                                [  OK  ]
   Stopping dlm_controld...                                [  OK  ]
   Stopping fenced...                                      [  OK  ]
   Stopping cman...                                        [  OK  ]
   Waiting for corosync to shutdown:                       [  OK  ]
   Unloading kernel modules...                             [  OK  ]
   Unmounting configfs...                                  [  OK  ]
[root@example-01 ~]#
```

2. At any node in the cluster, edit the **/etc/cluster/cluster.conf** file.

3. Update the **config_version** attribute by incrementing its value (for example, changing from **config_version="2"** to **config_version="3">**).

4. Save **/etc/cluster/cluster.conf**.

5. Validate the updated file against the cluster schema (**`cluster.rng`**) by running the **`ccs_config_validate`** command. For example:

```
[root@example-01 ~]# ccs_config_validate
Configuration validates
```

6. If the updated file is valid, use the **`scp`** command to propagate it to **`/etc/cluster/`** in each cluster node.

7. Verify that the updated configuration file has been propagated.

8. At each node, start the cluster software according to *Section 6.1.1, “Starting Cluster Software”*. For example:

```
[root@example-01 ~]# service cman start
Starting cluster:
   Checking Network Manager...                             [  OK  ]
   Global setup...                                         [  OK  ]
   Loading kernel modules...                               [  OK  ]
   Mounting configfs...                                    [  OK  ]
   Starting cman...                                        [  OK  ]
   Waiting for quorum...                                   [  OK  ]
   Starting fenced...                                      [  OK  ]
   Starting dlm_controld...                                [  OK  ]
   Starting gfs_controld...                                [  OK  ]
   Unfencing self...                                       [  OK  ]
   Joining fence domain...                                 [  OK  ]
[root@example-01 ~]# service clvmd start
Starting clvmd:                                            [  OK  ]
Activating VG(s):   2 logical volume(s) in volume group "vg_example" now active
                                                           [  OK  ]
[root@example-01 ~]# service gfs2 start
Mounting GFS2 filesystem (/mnt/gfsA):                      [  OK  ]
Mounting GFS2 filesystem (/mnt/gfsB):                      [  OK  ]
[root@example-01 ~]# service rgmanager start
Starting Cluster Service Manager:                          [  OK  ]
[root@example-01 ~]#
```

9. At any cluster node, run **`cman_tools nodes`** to verify that the nodes are functioning as members in the cluster (signified as "M" in the status column, "Sts"). For example:

```
[root@example-01 ~]# cman_tool nodes
Node  Sts   Inc   Joined               Name
   1   M    548   2010-09-28 10:52:21  node-01.example.com
   2   M    548   2010-09-28 10:52:21  node-02.example.com
   3   M    544   2010-09-28 10:52:21  node-03.example.com
```

10. At any node, using the **`clustat`** utility, verify that the HA services are running as expected. In addition, **`clustat`** displays status of the cluster nodes. For example:

```
[root@example-01 ~]#clustat
Cluster Status for mycluster @ Wed Nov 17 05:40:00 2010
Member Status: Quorate

 Member Name                             ID   Status
 ------ ----                             ---- ------
```

```
node-03.example.com                            3 Online, rgmanager
node-02.example.com                            2 Online, rgmanager
node-01.example.com                            1 Online, Local, rgmanager

Service Name                   Owner (Last)                   State
------- ----                   ----- ------                   -----
service:example_apache         node-01.example.com            started
service:example_apache2        (none)                         disabled
```

11. If the cluster is running as expected, you are done updating the configuration.

6.5. Diagnosing and Correcting Problems in a Cluster

For information about diagnosing and correcting problems in a cluster, contact an authorized Red Hat support representative.

Appendix A. Fence Device Parameters

This appendix provides tables with parameter descriptions of fence devices as well as the name of the fence agent for each of those devices.

> **Note**
>
> The **Name** parameter for a fence device specifies an arbitrary name for the device that will be used by Red Hat High Availability Add-On. This is not the same as the DNS name for the device.

> **Note**
>
> Certain fence devices have an optional **Password Script** parameter. The **Password Script** parameter allows you to specify that a fence-device password is supplied from a script rather than from the **Password** parameter. Using the **Password Script** parameter supersedes the **Password** parameter, allowing passwords to not be visible in the cluster configuration file (**`/etc/cluster/cluster.conf`**).

Table A.1. APC Power Switch (telnet/SSH)

Field	Description
Name	A name for the APC device connected to the cluster into which the fence daemon logs via telnet/ssh.
IP Address	The IP address or hostname assigned to the device.
Login	The login name used to access the device.
Password	The password used to authenticate the connection to the device.
Password Script (optional)	The script that supplies a password for access to the fence device. Using this supersedes the **Password** parameter.
Port	Physical plug number or name of virtual machine.
Switch (optional)	The switch number for the APC switch that connects to the node when you have multiple daisy-chained switches.
Use SSH	Indicates that system will use SSH to access the device.
Path to the SSH identity file	The identity file for SSH.
Power wait	Number of seconds to wait after issuing a power off or power on command.
`fence_apc`	The fence agent for APC over telnet/SSH.

Table A.2. APC Power Switch over SNMP

Field	Description
Name	A name for the APC device connected to the cluster into which the fence daemon logs via the SNMP protocol.
IP Address	The IP address or hostname assigned to the device.
UDP/TCP port	The UDP/TCP port to use for connection with the device; the default value is 161.
Login	The login name used to access the device.

Field	Description
Password	The password used to authenticate the connection to the device.
Password Script (optional)	The script that supplies a password for access to the fence device. Using this supersedes the **Password** parameter.
Port	Physical plug number or name of virtual machine.
Switch (optional)	The switch number for the APC switch that connects to the node when you have multiple daisy-chained switches.
SNMP version	The SNMP version to use (1, 2c, 3); the default value is 1.
SNMP community	The SNMP community string; the default value is **`private`**.
SNMP security level	The SNMP security level (noAuthNoPriv, authNoPriv, authPriv).
SNMP authentication protocol	The SNMP authentication protocol (MD5, SHA).
SNMP privacy protocol	The SNMP privacy protocol (DES, AES).
SNMP privacy protocol password	The SNMP privacy protocol password.
SNMP privacy protocol script	The script that supplies a password for SNMP privacy protocol. Using this supersedes the **SNMP privacy protocol password** parameter.
Power wait	Number of seconds to wait after issuing a power off or power on command.
`fence_apc_snmp`	The fence agent for APC that logs into the SNP device via the SNMP protocol.

Table A.3. Cisco MDS

Field	Description
Name	A name for the Cisco MDS 9000 series device with SNMP enabled.
IP Address	The IP address or hostname assigned to the device.
Login	The login name used to access the device.
Password	The password used to authenticate the connection to the device.
Password Script (optional)	The script that supplies a password for access to the fence device. Using this supersedes the **Password** parameter.
Port	Physical plug number or name of virtual machine.
SNMP version	The SNMP version to use (1, 2c, 3).
SNMP community	The SNMP community string.
SNMP authentication protocol	The SNMP authentication protocol (MD5, SHA).
SNMP security level	The SNMP security level (noAuthNoPriv, authNoPriv, authPriv).
SNMP privacy protocol	The SNMP privacy protocol (DES, AES).
SNMP privacy protocol password	The SNMP privacy protocol password.
SNMP privacy protocol script	The script that supplies a password for SNMP privacy protocol. Using this supersedes the **SNMP privacy protocol password** parameter.
Power wait	Number of seconds to wait after issuing a power off or power on command.
`fence_cisco_mds`	The fence agent for Cisco MDS.

Table A.4. Dell DRAC 5

Field	Description
Name	The name assigned to the DRAC.
IP Address	The IP address or hostname assigned to the DRAC.
IP port (optional)	The TCP port to use to connect to the device.
Login	The login name used to access the DRAC.
Password	The password used to authenticate the connection to the DRAC.
Password Script (optional)	The script that supplies a password for access to the fence device. Using this supersedes the **Password** parameter.
Module name	(optional) The module name for the DRAC when you have multiple DRAC modules.
Use SSH	Indicates that system will use SSH to access the device.
Path to the SSH identity file	The identity file for SSH.
Power wait	Number of seconds to wait after issuing a power off or power on command.
`fence_drac5`	The fence agent for Dell DRAC 5.

Table A.5. Egenera SAN Controller

Field	Description
Name	A name for the eGenera BladeFrame device connected to the cluster.
CServer	The hostname (and optionally the username in the form of **`username@hostname`**) assigned to the device. Refer to the fence_egenera(8) man page for more information.
ESH Path (optional)	The path to the esh command on the cserver (default is /opt/pan- mgr/bin/esh)
lpan	The logical process area network (LPAN) of the device.
pserver	The processing blade (pserver) name of the device.
`fence_egenera`	The fence agent for the eGenera BladeFrame.

Table A.6. ePowerSwitch

Field	Description
Name	A name for the ePowerSwitch device connected to the cluster.
IP Address	The IP address or hostname assigned to the device.
Login	The login name used to access the device.
Password	The password used to authenticate the connection to the device.
Password Script (optional)	The script that supplies a password for access to the fence device. Using this supersedes the **Password** parameter.
Port	Physical plug number or name of virtual machine.
Hidden page	The name of the hidden page for the device.
`fence_eps`	The fence agent for ePowerSwitch.

Table A.7. Fujitsu Siemens Remoteview Service Board (RSB)

Field	Description
Name	A name for the RSB to use as a fence device.

Field	Description
Hostname	The hostname assigned to the device.
Login	The login name used to access the device.
Password	The password used to authenticate the connection to the device.
Password Script (optional)	The script that supplies a password for access to the fence device. Using this supersedes the **Password** parameter.
TCP port	The port number on which the telnet service listens.
`fence_rsb`	The fence agent for Fujitsu-Siemens RSB.

Table A.8. Fence virt

Field	Description
Name	A name for the Fence virt fence device.
Port	Virtual machine (domain UUID or name) to fence.
Serial device	On the host, the serial device must be mapped in each domain's configuration file. For more information, see the **`fence_virt.conf`** man page. If this field is specified, it causes the **`fence_virt`** fencing agent to operate in serial mode. Not specifying a value causes the **`fence_virt`** fencing agent to operate in VM channel mode.
Serial parameters	The serial parameters. The default is 115200, 8N1.
VM channel IP address	The channel IP. The default value is 10.0.2.179.
`Channel port`	The channel port. The default value is 1229

Table A.9. HP iLO/iLO2 (Integrated Lights Out)

Field	Description
Name	A name for the server with HP iLO support.
Hostname	The hostname assigned to the device.
IP port (optional)	TCP port to use for connection with the device.
Login	The login name used to access the device.
Password	The password used to authenticate the connection to the device.
Password Script (optional)	The script that supplies a password for access to the fence device. Using this supersedes the **Password** parameter.
Power wait	Number of seconds to wait after issuing a power off or power on command.
`fence_ilo`	The fence agent for HP iLO devices.

Table A.10. HP iLO (Integrated Lights Out) MP

Field	Description
Name	A name for the server with HP iLO support.
Hostname	The hostname assigned to the device.
IP port (optional)	TCP port to use for connection with the device.
Login	The login name used to access the device.
Password	The password used to authenticate the connection to the device.
Password Script (optional)	The script that supplies a password for access to the fence device. Using this supersedes the **Password** parameter.

Field	Description
SSH	Indicates that the system will use SSH to access the device.
Path to the SSH identity file	The identity file for SSH.
Force command prompt	The command prompt to use. The default value is ’MP>’, ’hpiLO->’.
Power wait	Number of seconds to wait after issuing a power off or power on command.
`fence_ilo_mp`	The fence agent for HP iLO MP devices.

Table A.11. IBM BladeCenter

Field	Description
Name	A name for the IBM BladeCenter device connected to the cluster.
IP Address	The IP address or hostname assigned to the device.
IP port (optional)	TCP port to use for connection with the device.
Login	The login name used to access the device.
Password	The password used to authenticate the connection to the device.
Password Script (optional)	The script that supplies a password for access to the fence device. Using this supersedes the **Password** parameter.
Power wait	Number of seconds to wait after issuing a power off or power on command.
Use SSH	Indicates that system will use SSH to access the device.
Path to the SSH identity file	The identity file for SSH.
`fence_bladecenter`	The fence agent for IBM BladeCenter.

Table A.12. IBM BladeCenter SNMP

Field	Description
Name	A name for the IBM BladeCenter SNMP device connected to the cluster.
IP Address	The IP address or hostname assigned to the device.
UDP/TCP port (optional)	UDP/TCP port to use for connections with the device; the default value is 161.
Login	The login name used to access the device.
Password	The password used to authenticate the connection to the device.
Password Script (optional)	The script that supplies a password for access to the fence device. Using this supersedes the **Password** parameter.
Port	Physical plug number or name of virtual machine.
SNMP version	The SNMP version to use (1, 2c, 3); the default value is 1.
SNMP community	The SNMP community string.
SNMP security level	The SNMP security level (noAuthNoPriv, authNoPriv, authPriv).
SNMP authentication protocol	The SNMP authentication protocol (MD5, SHA).
SNMP privacy protocol	The SNMP privacy protocol (DES, AES).
SNMP privacy protocol password	The SNMP privacy protocol password.

Field	Description
SNMP privacy protocol script	The script that supplies a password for SNMP privacy protocol. Using this supersedes the **SNMP privacy protocol password** parameter.
Power wait	Number of seconds to wait after issuing a power off or power on command.
`fence_bladecenter`	The fence agent for IBM BladeCenter.

Table A.13. IF MIB

Field	Description
Name	A name for the IF MIB device connected to the cluster.
IP Address	The IP address or hostname assigned to the device.
UDP/TCP port(optiona)	The UDP/TCP port to use for connection with the device; the default value is 161.
Login	The login name used to access the device.
Password	The password used to authenticate the connection to the device.
Password Script (optional)	The script that supplies a password for access to the fence device. Using this supersedes the **Password** parameter.
SNMP version	The SNMP version to use (1, 2c, 3); the default value is 1.
SNMP community	The SNMP community string.
SNMP security level	The SNMP security level (noAuthNoPriv, authNoPriv, authPriv).
SNMP authentication protocol	The SNMP authentication protocol (MD5, SHA).
SNMP privacy protocol	The SNMP privacy protocol (DES, AES).
SNMP privacy protocol password	The SNMP privacy protocol password.
SNMP privacy protocol script	The script that supplies a password for SNMP privacy protocol. Using this supersedes the **SNMP privacy protocol password** parameter.
Power wait	Number of seconds to wait after issuing a power off or power on command.
Port	Physical plug number or name of virtual machine.
`fence_ifmib`	The fence agent for IF-MIB devices.

Table A.14. Intel Modular

Field	Description
Name	A name for the Intel Modular device connected to the cluster.
IP Address	The IP address or hostname assigned to the device.
Login	The login name used to access the device.
Password	The password used to authenticate the connection to the device.
Password Script (optional)	The script that supplies a password for access to the fence device. Using this supersedes the **Password** parameter.
Port	Physical plug number or name of virtual machine.
SNMP version	The SNMP version to use (1, 2c, 3); the default value is 1.
SNMP community	The SNMP community string; the default value is **`private`**.
SNMP security level	The SNMP security level (noAuthNoPriv, authNoPriv, authPriv).

Field	Description
SNMP authentication protocol	The SNMP authentication protocol (MD5, SHA).
SNMP privacy protocol	The SNMP privacy protocol (DES, AES).
SNMP privacy protocol password	The SNMP privacy protocol password.
SNMP privacy protocol script	The script that supplies a password for SNMP privacy protocol. Using this supersedes the **SNMP privacy protocol password** parameter.
Power wait	Number of seconds to wait after issuing a power off or power on command.
`fence_intelmodular`	The fence agent for APC.

Table A.15. IPMI (Intelligent Platform Management Interface) LAN

Field	Description
Name	A name for the IPMI LAN device connected to the cluster.
IP Address	The IP address or hostname assigned to the device.
Login	The login name of a user capable of issuing power on/off commands to the given IPMI port.
Password	The password used to authenticate the connection to the IPMI port.
Password Script (optional)	The script that supplies a password for access to the fence device. Using this supersedes the **Password** parameter.
Authentication Type	**`none`**, **`password`**, **`md2`**, or **`md5`**
Use Lanplus	**`True`** or **`1`**. If blank, then value is **`False`**.
Ciphersuite to use	The remote server authentication, integrity, and encryption algorithms to use for IPMIv2 lanplus connections.
`fence_ipmilan`	The fence agent for machines controlled by IPMI.

Table A.16. SCSI Fencing

Field	Description
Name	A name for the SCSI fence device.
Node name	Name of the node to be fenced. Refer to the **`fence_scsi`**(8) man page for more information.
`fence_scsi`	The fence agent for SCSI persistent reservations.

Note

Use of SCSI persistent reservations as a fence method is supported with the following limitations:

- When using SCSI fencing, all nodes in the cluster must register with the same devices so that each node can remove another node's registration key from all the devices it is registered with.
- Devices used for the cluster volumes should be a complete LUN, not partitions. SCSI persistent reservations work on an entire LUN, meaning that access is controlled to each LUN, not individual partitions.

Table A.17. WTI Power Switch

Field	Description
Name	A name for the WTI power switch connected to the cluster.
IP Address	The IP or hostname address assigned to the device.
IP port (optional)	The TCP port to use to connect to the device.
Login	The login name used to access the device.
Password	The password used to authenticate the connection to the device.
Password Script (optional)	The script that supplies a password for access to the fence device. Using this supersedes the **Password** parameter.
Port	Physical plug number or name of virtual machine.
Force command prompt	The command prompt to use. The default value is [’RSM>’, ’>MPC’, ’IPS>’, ’TPS>’, ’NBB>’, ’NPS>’, ’VMR>’]
Power wait	Number of seconds to wait after issuing a power off or power on command.
Use SSH	Indicates that system will use SSH to access the device.
Path to the SSH identity file	The identity file for SSH.
`fence_wti`	The fence agent for the WTI network power switch.

Appendix B. HA Resource Parameters

This appendix provides descriptions of HA resource parameters. You can configure the parameters with **Luci** or by editing **`etc/cluster/cluster.conf`**. *Table B.1, “HA Resource Summary”* lists the resources, their corresponding resource agents, and references to other tables containing parameter descriptions. To understand resource agents in more detail you can view them in **`/usr/share/cluster`** of any cluster node.

Table B.1. HA Resource Summary

Resource	Resource Agent	Reference to Parameter Description
Apache	apache.sh	*Table B.2, “Apache Server”*
File System	fs.sh	*Table B.3, “File System”*
GFS2 File System	clusterfs.sh	*Table B.4, “GFS2”*
IP Address	ip.sh	*Table B.5, “IP Address”*
LVM	lvm.sh	*Table B.6, “LVM”*
MySQL	mysql.sh	*Table B.7, “MySQL®”*
NFS Client	nfsclient.sh	*Table B.8, “NFS Client”*
NFS Export	nfsexport.sh	*Table B.9, “NFS Export”*
NFS Mount	netfs.sh	*Table B.10, “NFS Mount”*
Open LDAP	openldap.sh	*Table B.11, “Open LDAP”*
Oracle 10g	oracledb.sh	*Table B.12, “Oracle® 10g”*
PostgreSQL 8	postgres-8.sh	*Table B.13, “PostgreSQL 8”*
SAP Database	SAPDatabase	*Table B.14, “SAP® Database”*
SAP Instance	SAPInstance	*Table B.15, “SAP® Instance”*
Samba	smb.sh	*Table B.16, “Samba Service”*
Script	script.sh	*Table B.17, “Script”*
Service	service.sh	*Table B.18, “Service”*
Sybase ASE	ASEHAagent.sh	*Table B.19, “Sybase® ASE Failover Instance”*
Tomcat 6	tomcat-6.sh	*Table B.20, “Tomcat 6”*
Virtual Machine	vm.sh	*Table B.21, “Virtual Machine”* NOTE: **Luci** displays this as a virtual service if the host cluster can support virtual machines.

Table B.2. Apache Server

Field	Description
Name	The name of the Apache Service.
Server Root	The default value is **`/etc/httpd`**.
Config File	Specifies the Apache configuration file. The default valuer is **`/etc/httpd/conf`**.
httpd Options	Other command line options for **`httpd`**.

Field	Description
Shutdown Wait (seconds)	Specifies the number of seconds to wait for correct end of service shutdown.

Table B.3. File System

Field	Description
Name	Specifies a name for the file system resource.
File System Type	If not specified, **`mount`** tries to determine the file system type.
Mount Point	Path in file system hierarchy to mount this file system.
Device	Specifies the device associated with the file system resource. This can be a block device, file system label, or UUID of a file system.
Options	Mount options; that is, options used when the file system is mounted. These may be file-system specific. Refer to the ***`mount`**(8)* man page for supported mount options.
File System ID	Note *`File System ID`* is used only by NFS services. When creating a new file system resource, you can leave this field blank. Leaving the field blank causes a file system ID to be assigned automatically after you commit the parameter during configuration. If you need to assign a file system ID explicitly, specify it in this field.
Force unmount	If enabled, forces the file system to unmount. The default setting is *`disabled`*. *`Force Unmount`* kills all processes using the mount point to free up the mount when it tries to unmount.
Reboot host node if unmount fails	If enabled, reboots the node if unmounting this file system fails. The default setting is *`disabled`*.
Check file system before mounting	If enabled, causes **`fsck`** to be run on the file system before mounting it. The default setting is *`disabled`*.

Table B.4. GFS2

Field	Description
Name	The name of the file system resource.
Mount Point	The path to which the file system resource is mounted.
Device	The device file associated with the file system resource.
Options	Mount options.
File System ID	Note *`File System ID`* is used only by NFS services. When creating a new GFS2 resource, you can leave this field blank. Leaving the field blank causes a file system ID to be assigned automatically after you commit

Field	Description
	the parameter during configuration. If you need to assign a file system ID explicitly, specify it in this field.
Force Unmount	If enabled, forces the file system to unmount. The default setting is *`disabled`*. *`Force Unmount`* kills all processes using the mount point to free up the mount when it tries to unmount. With GFS2 resources, the mount point is *not* unmounted at service tear-down unless *`Force Unmount`* is *enabled*.
Reboot Host Node if Unmount Fails (self fence)	If enabled and unmounting the file system fails, the node will immediately reboot. Generally, this is used in conjunction with force-unmount support, but it is not required.

Table B.5. IP Address

Field	Description
IP Address	The IP address for the resource. This is a virtual IP address. IPv4 and IPv6 addresses are supported, as is NIC link monitoring for each IP address.
Monitor Link	Enabling this causes the status check to fail if the link on the NIC to which this IP address is bound is not present.

Table B.6. LVM

Field	Description
Name	A unique name for this LVM resource.
Volume Group Name	A descriptive name of the volume group being managed.
Logical Volume Name (optional)	Name of the logical volume being managed. This parameter is optional if there is more than one logical volume in the volume group being managed.

Table B.7. MySQL®

Field	Description
Name	Specifies a name of the MySQL server resource.
Config File	Specifies the configuration file. The default value is **`/etc/my.cnf`**.
Listen Address	Specifies an IP address for MySQL server. If an IP address is not provided, the first IP address from the service is taken.
mysqld Options	Other command line options for **`httpd`**.
Shutdown Wait (seconds)	Specifies the number of seconds to wait for correct end of service shutdown.

Table B.8. NFS Client

Field	Description
Name	This is a symbolic name of a client used to reference it in the resource tree. This is *not* the same thing as the *`Target`* option.
Target	This is the server from which you are mounting. It can be specified using a hostname, a wildcard (IP address or hostname based), or a netgroup defining a host or hosts to export to.

Field	Description
Option	Defines a list of options for this client — for example, additional client access rights. For more information, refer to the ***`exports`*** *(5)* man page, *General Options*.

Table B.9. NFS Export

Field	Description
Name	Descriptive name of the resource. The NFS Export resource ensures that NFS daemons are running. It is fully reusable; typically, only one NFS Export resource is needed. **Tip** Name the NFS Export resource so it is clearly distinguished from other NFS resources.

Table B.10. NFS Mount

Field	Description
Name	Symbolic name for the NFS mount. **Note** This resource is required only when a cluster service is configured to be an NFS client.
Mount Point	Path to which the file system resource is mounted.
Host	NFS server IP address or hostname.
Export Path	NFS Export directory name.
NFS version	NFS protocol: • *`NFS3`* — Specifies using NFSv3 protocol. The default setting is *`NFS3`*. • *`NFS4`* — Specifies using NFSv4 protocol.
Options	Mount options. Specifies a list of mount options. If none are specified, the NFS file system is mounted **`-o sync`**. For more information, refer to the ***`nfs`****(5)* man page.
Force Unmount	If *`Force Unmount`* is enabled, the cluster kills all processes using this file system when the service is stopped. Killing all processes using the file system frees up the file system. Otherwise, the unmount will fail, and the service will be restarted.

Table B.11. Open LDAP

Field	Description
Name	Specifies a service name for logging and other purposes.
Config File	Specifies an absolute path to a configuration file. The default value is **`/etc/openldap/slapd.conf`**.
URL List	The default value is **`ldap:///`**.

Field	Description
slapd Options	Other command line options for **slapd**.
Shutdown Wait (seconds)	Specifies the number of seconds to wait for correct end of service shutdown.

Table B.12. Oracle® 10g

Field	Description
Instance name (SID) of Oracle instance	Instance name.
Oracle user name	This is the user name of the Oracle user that the Oracle AS instance runs as.
Oracle application home directory	This is the Oracle (application, not user) home directory. It is configured when you install Oracle.
Virtual hostname (optional)	Virtual Hostname matching the installation hostname of Oracle 10g. Note that during the start/stop of an oracledb resource, your hostname is changed temporarily to this hostname. Therefore, you should configure an oracledb resource as part of an exclusive service only.

Table B.13. PostgreSQL 8

Field	Description
Name	Specifies a service name for logging and other purposes.
Config File	Define absolute path to configuration file. The default value is **/var/lib/pgsql/data/postgresql.conf**.
Postmaster User	User who runs the database server because it can't be run by root. The default value is postgres.
Postmaster Options	Other command line options for postmaster.
Shutdown Wait (seconds)	Specifies the number of seconds to wait for correct end of service shutdown.

Table B.14. SAP® Database

Field	Description
SAP Database Name	Specifies a unique SAP system identifier. For example, P01.
SAP executable directory	Specifies the fully qualified path to **sapstartsrv** and **sapcontrol**.
Database type	Specifies one of the following database types: Oracle, DB6, or ADA.
Oracle TNS listener name	Specifies Oracle TNS listener name.
ABAP stack is not installed, only Java stack is installed	If you do not have an ABAP stack installed in the SAP database, enable this parameter.

Field	Description
J2EE instance bootstrap directory	The fully qualified path the J2EE instance bootstrap directory. For example, **`/usr/sap/P01/J00/j2ee/cluster/bootstrap`**.
J2EE security store path	The fully qualified path the J2EE security store directory. For example, **`/usr/sap/P01/SYS/global/security/lib/tools`**.

Table B.15. SAP® Instance

Field	Description
SAP Instance Name	The fully qualified SAP instance name. For example, P01_DVEBMGS00_sapp01ci.
SAP executable directory	The fully qualified path to **`sapstartsrv`** and **`sapcontrol`**.
Directory containing the SAP START profile	The fully qualified path to the SAP START profile.
Name of the SAP START profile	Specifies name of the SAP START profile.

Note

Regarding *Table B.16, “Samba Service”*, when creating or editing a cluster service, connect a Samba-service resource directly to the service, *not* to a resource within a service.

Table B.16. Samba Service

Field	Description
Name	Specifies the name of the Samba server.
Workgroup	Specifies a Windows workgroup name or Windows NT domain of the Samba service.

Table B.17. Script

Field	Description
Name	Specifies a name for the custom user script. The script resource allows a standard LSB-compliant init script to be used to start a clustered service.
File (with path)	Enter the path where this custom script is located (for example, **`/etc/init.d/`*`userscript`***).

Table B.18. Service

Field	Description
Service name	Name of service. This defines a collection of resources, known as a resource group or cluster service.
Automatically start this service	If enabled, this service (or resource group) is started automatically after the cluster forms a quorum. If this parameter is *disabled*, this service is *not* started

Field	Description
	automatically after the cluster forms a quorum; the service is put into the *`disabled`* state.
Run exclusive	If enabled, this service (resource group) can only be relocated to run on another node exclusively; that is, to run on a node that has no other services running on it. If no nodes are available for a service to run exclusively, the service is not restarted after a failure. Additionally, other services do not automatically relocate to a node running this service as *`Run exclusive`*. You can override this option by manual start or relocate operations.
Failover Domain	Defines lists of cluster members to try in the event that a service fails.
Recovery policy	*`Recovery policy`* provides the following options: • *`Disable`* — Disables the resource group if any component fails. • *`Relocate`* — Tries to restart service in another node; that is, it does not try to restart in the current node. • *`Restart`* — Tries to restart failed parts of this service locally (in the current node) before trying to relocate (default) to service to another node.

Table B.19. Sybase® ASE Failover Instance

Field	Description
Instance Name	Specifies the instance name of the Sybase ASE resource.
ASE server name	The ASE server name that is configured for the HA service.
Sybase home directory	The home directory of Sybase products.
Login file	The full path of login file that contains the login-password pair.
Interfaces file	The full path of the interfaces file that is used to start/access the ASE server.
SYBASE_ASE directory name	The directory name under sybase_home where ASE products are installed.
SYBASE_OCS directory name	The directory name under sybase_home where OCS products are installed. For example, ASE-15_0.
Sybase user	The user who can run ASE server.
Deep probe timeout	The maximum seconds to wait for the response of ASE server before determining that the server had no response while running deep probe.

Table B.20. Tomcat 6

Field	Description
Name	Specifies a service name for logging and other purposes.
Config File	Specifies the absolute path to the configuration file. The default value is **`/etc/tomcat6/tomcat6.conf`**.
Tomcat User	User who runs the Tomcat server. The default value is *tomcat*.
Catalina Options	Other command line options for Catalina.
Catalina Base	Catalina base directory (differs for each service) The default value is /usr/share/tomcat6.

Field	Description
Shutdown Wait (seconds)	Specifies the number of seconds to wait for correct end of service shutdown. The default value is 30.

Table B.21. Virtual Machine

Field	Description
Virtual machine name	Specifies the name of the virtual machine.
Path to VM configuration files	A colon-delimited path specification that the Virtual Machine Resource Agent (**`vm.sh`**) searches for the virtual machine configuration file. For example: **`/mnt/guests/config:/etc/libvirt/qemu`**. **Important** The path should *never* directly point to a virtual machine configuration file.
Automatically start this virtual machine	If enabled, this virtual machine is started automatically after the cluster forms a quorum. If this parameter is *disabled*, this virtual machine is *not* started automatically after the cluster forms a quorum; the virtual machine is put into the *`disabled`* state.
Run exclusive	If enabled, this virtual machine can only be relocated to run on another node exclusively; that is, to run on a node that has no other virtual machines running on it. If no nodes are available for a virtual machine to run exclusively, the virtual machine is not restarted after a failure. Additionally, other virtual machines do not automatically relocate to a node running this virtual machine as *`Run exclusive`*. You can override this option by manual start or relocate operations.
Failover Domain	Defines lists of cluster members to try in the event that a virtual machine fails.
Recovery policy	*`Recovery policy`* provides the following options: • *`Disable`* — Disables the virtual machine if it fails. • *`Relocate`* — Tries to restart the virtual machine in another node; that is, it does not try to restart in the current node. • *`Restart`* — Tries to restart the virtual machine locally (in the current node) before trying to relocate (default) to virtual machine to another node.
Migration type	Specifies a migration type of *`live`* or *`pause`*. The default setting is *`live`*.

Appendix C. HA Resource Behavior

This appendix describes common behavior of HA resources. It is meant to provide ancillary information that may be helpful in configuring HA services. You can configure the parameters with **Luci** or by editing **`etc/cluster/cluster.conf`**. For descriptions of HA resource parameters, refer to *Appendix B, HA Resource Parameters*. To understand resource agents in more detail you can view them in **`/usr/share/cluster`** of any cluster node.

> **Note**
>
> To fully comprehend the information in this appendix, you may require detailed understanding of resource agents and the cluster configuration file, **`/etc/cluster/cluster.conf`**.

An HA service is a group of cluster resources configured into a coherent entity that provides specialized services to clients. An HA service is represented as a resource tree in the cluster configuration file, **`/etc/cluster/cluster.conf`** (in each cluster node). In the cluster configuration file, each resource tree is an XML representation that specifies each resource, its attributes, and its relationship among other resources in the resource tree (parent, child, and sibling relationships).

> **Note**
>
> Because an HA service consists of resources organized into a hierarchical tree, a service is sometimes referred to as a *resource tree* or *resource group*. Both phrases are synonymous with *HA service*.

At the root of each resource tree is a special type of resource — a *service resource*. Other types of resources comprise the rest of a service, determining its characteristics. Configuring an HA service consists of creating a service resource, creating subordinate cluster resources, and organizing them into a coherent entity that conforms to hierarchical restrictions of the service.

This appendix consists of the following sections:

- *Section C.1, “Parent, Child, and Sibling Relationships Among Resources”*
- *Section C.2, “Sibling Start Ordering and Resource Child Ordering”*
- *Section C.3, “Inheritance, the <resources> Block, and Reusing Resources”*
- *Section C.4, “Failure Recovery and Independent Subtrees”*
- *Section C.5, “Debugging and Testing Services and Resource Ordering”*

> **Note**
>
> The sections that follow present examples from the cluster configuration file, **`/etc/cluster/cluster.conf`**, for illustration purposes only.

C.1. Parent, Child, and Sibling Relationships Among Resources

A cluster service is an integrated entity that runs under the control of **`rgmanager`**. All resources in a service run on the same node. From the perspective of **`rgmanager`**, a cluster service is one entity that can be started, stopped, or relocated. Within a cluster service, however, the hierarchy of the resources determines the order in which each resource is started and stopped.The hierarchical levels consist of parent, child, and sibling.

Example C.1, “Resource Hierarchy of Service foo” shows a sample resource tree of the service *foo*. In the example, the relationships among the resources are as follows:

- **`fs:myfs`** (<fs name="myfs" ...>) and **`ip:10.1.1.2`** (<ip address="10.1.1.2 .../>) are siblings.
- **`fs:myfs`** (<fs name="myfs" ...>) is the parent of **`script:script_child`** (<script name="script_child"/>).
- **`script:script_child`** (<script name="script_child"/>) is the child of **`fs:myfs`** (<fs name="myfs" ...>).

Example C.1. Resource Hierarchy of Service foo

```
<service name="foo" ...>
    <fs name="myfs" ...>
        <script name="script_child"/>
    </fs>
    <ip address="10.1.1.2" .../>
</service>
```

The following rules apply to parent/child relationships in a resource tree:

- Parents are started before children.
- Children must all stop cleanly before a parent may be stopped.
- For a resource to be considered in good health, all its children must be in good health.

C.2. Sibling Start Ordering and Resource Child Ordering

The Service resource determines the start order and the stop order of a child resource according to whether it designates a child-type attribute for a child resource as follows:

- Designates child-type attribute (*typed* child resource) — If the Service resource designates a child-type attribute for a child resource, the child resource is *typed*. The child-type attribute explicitly determines the start and the stop order of the child resource.
- *Does not designate* child-type attribute (*non-typed* child resource) — If the Service resource *does not designate* a child-type attribute for a child resource, the child resource is *non-typed*. The Service resource does not explicitly control the starting order and stopping order of a non-typed child resource. However, a non-typed child resource is started and stopped according to its order in **`/etc/cluster.cluster.conf`** In addition, non-typed child resources are started after all typed child resources have started and are stopped before any typed child resources have stopped.

Note

The only resource to implement defined *child resource type* ordering is the Service resource.

For more information about typed child resource start and stop ordering, refer to *Section C.2.1, “Typed Child Resource Start and Stop Ordering”*. For more information about non-typed child resource start and stop ordering, refer to *Section C.2.2, “Non-typed Child Resource Start and Stop Ordering”*.

C.2.1. Typed Child Resource Start and Stop Ordering

For a typed child resource, the type attribute for the child resource defines the start order and the stop order of each resource type with a number from 1 and 100; one value for start, and one value for stop. The lower the number, the earlier a resource type starts or stops. For example, *Table C.1, “Child Resource Type Start and Stop Order”* shows the start and stop values for each resource type; *Example C.2, “Resource Start and Stop Values: Excerpt from Service Resource Agent, **service.sh**”* shows the start and stop values as they appear in the Service resource agent, **service.sh**. For the Service resource, all LVM children are started first, followed by all File System children, followed by all Script children, and so forth.

Table C.1. Child Resource Type Start and Stop Order

Resource	Child Type	Start-order Value	Stop-order Value
LVM	lvm	1	9
File System	fs	2	8
GFS2 File System	clusterfs	3	7
NFS Mount	netfs	4	6
NFS Export	nfsexport	5	5
NFS Client	nfsclient	6	4
IP Address	ip	7	2
Samba	smb	8	3
Script	script	9	1

Example C.2. Resource Start and Stop Values: Excerpt from Service Resource Agent, **service.sh**

```
<special tag="rgmanager">
    <attributes root="1" maxinstances="1"/>
    <child type="lvm" start="1" stop="9"/>
    <child type="fs" start="2" stop="8"/>
    <child type="clusterfs" start="3" stop="7"/>
    <child type="netfs" start="4" stop="6"/>
    <child type="nfsexport" start="5" stop="5"/>
    <child type="nfsclient" start="6" stop="4"/>
    <child type="ip" start="7" stop="2"/>
    <child type="smb" start="8" stop="3"/>
    <child type="script" start="9" stop="1"/>
</special>
```

Ordering within a resource type is preserved as it exists in the cluster configuration file, `/etc/cluster/cluster.conf`. For example, consider the starting order and stopping order of the typed child resources in *Example C.3, “Ordering Within a Resource Type”*.

Example C.3. Ordering Within a Resource Type

```
<service name="foo">
  <script name="1" .../>
  <lvm name="1" .../>
  <ip address="10.1.1.1" .../>
  <fs name="1" .../>
  <lvm name="2" .../>
</service>
```

Typed Child Resource Starting Order

In *Example C.3, “Ordering Within a Resource Type”*, the resources are started in the following order:

1. `lvm:1` — This is an LVM resource. All LVM resources are started first. `lvm:1` (`<lvm name="1" .../>`) is the first LVM resource started among LVM resources because it is the first LVM resource listed in the Service *foo* portion of `/etc/cluster/cluster.conf`.

2. `lvm:2` — This is an LVM resource. All LVM resources are started first. `lvm:2` (`<lvm name="2" .../>`) is started after `lvm:1` because it is listed after `lvm:1` in the Service *foo* portion of `/etc/cluster/cluster.conf`.

3. `fs:1` — This is a File System resource. If there were other File System resources in Service *foo*, they would start in the order listed in the Service *foo* portion of `/etc/cluster/cluster.conf`.

4. `ip:10.1.1.1` — This is an IP Address resource. If there were other IP Address resources in Service *foo*, they would start in the order listed in the Service *foo* portion of `/etc/cluster/cluster.conf`.

5. `script:1` — This is a Script resource. If there were other Script resources in Service *foo*, they would start in the order listed in the Service *foo* portion of `/etc/cluster/cluster.conf`.

Typed Child Resource Stopping Order

In *Example C.3, “Ordering Within a Resource Type”*, the resources are stopped in the following order:

1. `script:1` — This is a Script resource. If there were other Script resources in Service *foo*, they would stop in the reverse order listed in the Service *foo* portion of `/etc/cluster/cluster.conf`.

2. `ip:10.1.1.1` — This is an IP Address resource. If there were other IP Address resources in Service *foo*, they would stop in the reverse order listed in the Service *foo* portion of `/etc/cluster/cluster.conf`.

3. `fs:1` — This is a File System resource. If there were other File System resources in Service *foo*, they would stop in the reverse order listed in the Service *foo* portion of `/etc/cluster/cluster.conf`.

4. `lvm:2` — This is an LVM resource. All LVM resources are stopped last. `lvm:2` (`<lvm name="2" .../>`) is stopped before `lvm:1`; resources within a group of a resource type are stopped in the reverse order listed in the Service *foo* portion of `/etc/cluster/cluster.conf`.

5. **lvm:1** — This is an LVM resource. All LVM resources are stopped last. **lvm:1** (**<lvm name="1" .../>**) is stopped after **lvm:2**; resources within a group of a resource type are stopped in the reverse order listed in the Service *foo* portion of **/etc/cluster/cluster.conf**.

C.2.2. Non-typed Child Resource Start and Stop Ordering

Additional considerations are required for non-typed child resources. For a non-typed child resource, starting order and stopping order are not explicitly specified by the Service resource. Instead, starting order and stopping order are determined according to the order of the child resource in **/etc/cluster.cluster.conf**. Additionally, non-typed child resources are started after all typed child resources and stopped before any typed child resources.

For example, consider the starting order and stopping order of the non-typed child resources in *Example C.4, "Non-typed and Typed Child Resource in a Service"*.

Example C.4. Non-typed and Typed Child Resource in a Service

```
<service name="foo">
  <script name="1" .../>
  <nontypedresource name="foo"/>
  <lvm name="1" .../>
  <nontypedresourcetwo name="bar"/>
  <ip address="10.1.1.1" .../>
  <fs name="1" .../>
  <lvm name="2" .../>
</service>
```

Non-typed Child Resource Starting Order

In *Example C.4, "Non-typed and Typed Child Resource in a Service"*, the child resources are started in the following order:

1. **lvm:1** — This is an LVM resource. All LVM resources are started first. **lvm:1** (**<lvm name="1" .../>**) is the first LVM resource started among LVM resources because it is the first LVM resource listed in the Service *foo* portion of **/etc/cluster/cluster.conf**.

2. **lvm:2** — This is an LVM resource. All LVM resources are started first. **lvm:2** (**<lvm name="2" .../>**) is started after **lvm:1** because it is listed after **lvm:1** in the Service *foo* portion of **/etc/cluster/cluster.conf**.

3. **fs:1** — This is a File System resource. If there were other File System resources in Service *foo*, they would start in the order listed in the Service *foo* portion of **/etc/cluster/cluster.conf**.

4. **ip:10.1.1.1** — This is an IP Address resource. If there were other IP Address resources in Service *foo*, they would start in the order listed in the Service *foo* portion of **/etc/cluster/cluster.conf**.

5. **script:1** — This is a Script resource. If there were other Script resources in Service *foo*, they would start in the order listed in the Service *foo* portion of **/etc/cluster/cluster.conf**.

6. **nontypedresource:foo** — This is a non-typed resource. Because it is a non-typed resource, it is started after the typed resources start. In addition, its order in the Service resource is before the other non-typed resource, **nontypedresourcetwo:bar**; therefore, it is started before **nontypedresourcetwo:bar**. (Non-typed resources are started in the order that they appear in the Service resource.)

7. **nontypedresourcetwo:bar** — This is a non-typed resource. Because it is a non-typed resource, it is started after the typed resources start. In addition, its order in the Service resource is after the other non-typed resource, **nontypedresource:foo**; therefore, it is started after **nontypedresource:foo**. (Non-typed resources are started in the order that they appear in the Service resource.)

Non-typed Child Resource Stopping Order

In *Example C.4, “Non-typed and Typed Child Resource in a Service”*, the child resources are stopped in the following order:

1. **nontypedresourcetwo:bar** — This is a non-typed resource. Because it is a non-typed resource, it is stopped before the typed resources are stopped. In addition, its order in the Service resource is after the other non-typed resource, **nontypedresource:foo**; therefore, it is stopped before **nontypedresource:foo**. (Non-typed resources are stopped in the reverse order that they appear in the Service resource.)

2. **nontypedresource:foo** — This is a non-typed resource. Because it is a non-typed resource, it is stopped before the typed resources are stopped. In addition, its order in the Service resource is before the other non-typed resource, **nontypedresourcetwo:bar**; therefore, it is stopped before **nontypedresourcetwo:bar**. (Non-typed resources are stopped in the reverse order that they appear in the Service resource.)

3. **script:1** — This is a Script resource. If there were other Script resources in Service *foo*, they would stop in the reverse order listed in the Service *foo* portion of **/etc/cluster/cluster.conf**.

4. **ip:10.1.1.1** — This is an IP Address resource. If there were other IP Address resources in Service *foo*, they would stop in the reverse order listed in the Service *foo* portion of **/etc/cluster/cluster.conf**.

5. **fs:1** — This is a File System resource. If there were other File System resources in Service *foo*, they would stop in the reverse order listed in the Service *foo* portion of **/etc/cluster/cluster.conf**.

6. **lvm:2** — This is an LVM resource. All LVM resources are stopped last. **lvm:2** (**<lvm name="2" .../>**) is stopped before **lvm:1**; resources within a group of a resource type are stopped in the reverse order listed in the Service *foo* portion of **/etc/cluster/cluster.conf**.

7. **lvm:1** — This is an LVM resource. All LVM resources are stopped last. **lvm:1** (**<lvm name="1" .../>**) is stopped after **lvm:2**; resources within a group of a resource type are stopped in the reverse order listed in the Service *foo* portion of **/etc/cluster/cluster.conf**.

C.3. Inheritance, the <resources> Block, and Reusing Resources

Some resources benefit by inheriting values from a parent resource; that is commonly the case in an NFS service. *Example C.5, “NFS Service Set Up for Resource Reuse and Inheritance”* shows a typical NFS service configuration, set up for resource reuse and inheritance.

Example C.5. NFS Service Set Up for Resource Reuse and Inheritance

```
<resources>
    <nfsclient name="bob" target="bob.test.com" options="rw,no_root_squash"/>
    <nfsclient name="jim" target="jim.test.com" options="rw,no_root_squash"/>
    <nfsexport name="exports"/>
</resources>
<service name="foo">
    <fs name="1" mountpoint="/mnt/foo" device="/dev/sdb1" fsid="12344">
        <nfsexport ref="exports">  <!-- nfsexport's path and fsid attributes
                                        are inherited from the mountpoint &
                                        fsid attribute of the parent fs
                                        resource -->
            <nfsclient ref="bob"/> <!-- nfsclient's path is inherited from the
                                        mountpoint and the fsid is added to the
                                        options string during export -->
            <nfsclient ref="jim"/>
        </nfsexport>
    </fs>
    <fs name="2" mountpoint="/mnt/bar" device="/dev/sdb2" fsid="12345">
        <nfsexport ref="exports">
            <nfsclient ref="bob"/> <!-- Because all of the critical data for this
                                        resource is either defined in the
                                        resources block or inherited, we can
                                        reference it again! -->
            <nfsclient ref="jim"/>
        </nfsexport>
    </fs>
    <ip address="10.2.13.20"/>
</service>
```

If the service were flat (that is, with no parent/child relationships), it would need to be configured as follows:

- The service would need four nfsclient resources — one per file system (a total of two for file systems), and one per target machine (a total of two for target machines).
- The service would need to specify export path and file system ID to each nfsclient, which introduces chances for errors in the configuration.

In *Example C.5, “NFS Service Set Up for Resource Reuse and Inheritance”* however, the NFS client resources *nfsclient:bob* and *nfsclient:jim* are defined once; likewise, the NFS export resource *nfsexport:exports* is defined once. All the attributes needed by the resources are inherited from parent resources. Because the inherited attributes are dynamic (and do not conflict with one another), it is possible to reuse those resources — which is why they are defined in the resources block. It may not be practical to configure some resources in multiple places. For example, configuring a file system resource in multiple places can result in mounting one file system on two nodes, therefore causing problems.

C.4. Failure Recovery and Independent Subtrees

In most enterprise environments, the normal course of action for failure recovery of a service is to restart the entire service if any component in the service fails. For example, in *Example C.6, “Service foo Normal Failure Recovery”*, if any of the scripts defined in this service fail, the normal course of action is to restart (or relocate or disable, according to the service recovery policy) the service.

However, in some circumstances certain parts of a service may be considered non-critical; it may be necessary to restart only part of the service in place before attempting normal recovery. To accomplish that, you can use the `__independent_subtree` attribute. For example, in *Example C.7, "Service foo Failure Recovery with `__independent_subtree` Attribute"*, the `__independent_subtree` attribute is used to accomplish the following actions:

- If script:script_one fails, restart script:script_one, script:script_two, and script:script_three.
- If script:script_two fails, restart just script:script_two.
- If script:script_three fails, restart script:script_one, script:script_two, and script:script_three.
- If script:script_four fails, restart the whole service.

Example C.6. Service *foo* Normal Failure Recovery

```
<service name="foo">
      <script name="script_one" ...>
          <script name="script_two" .../>
      </script>
      <script name="script_three" .../>
</service>
```

Example C.7. Service *foo* Failure Recovery with `__independent_subtree` Attribute

```
<service name="foo">
      <script name="script_one" __independent_subtree="1" ...>
          <script name="script_two" __independent_subtree="1" .../>
          <script name="script_three" .../>
      </script>
      <script name="script_four" .../>
</service>
```

C.5. Debugging and Testing Services and Resource Ordering

You can debug and test services and resource ordering with the **`rg_test`** utility. **`rg_test`** is a command-line utility provided by the **`rgmanager`** package that is run from a shell or a terminal (it is not available in **Conga**). *Table C.2, "`rg_test` Utility Summary"* summarizes the actions and syntax for the **`rg_test`** utility.

Table C.2. **`rg_test`** Utility Summary

Action	Syntax
Display the resource rules that **`rg_test`** understands.	**`rg_test rules`**
Test a configuration (and /usr/ share/	**`rg_test test /etc/cluster/cluster.conf`**

Action	Syntax
cluster) for errors or redundant resource agents.	
Display the start and stop ordering of a service.	Display start order: **`rg_test noop /etc/cluster/cluster.conf start service`** ***servicename*** Display stop order: **`rg_test noop /etc/cluster/cluster.conf stop service`** ***servicename***
Explicitly start or stop a service.	**Important** Only do this on one node, and always disable the service in rgmanager first. Start a service: **`rg_test test /etc/cluster/cluster.conf start service`** ***servicename*** Stop a service: **`rg_test test /etc/cluster/cluster.conf stop service`** ***servicename***
Calculate and display the resource tree delta between two cluster.conf files.	**`rg_test delta`** ***cluster.conf file 1*** ***cluster.conf file 2*** For example: **`rg_test delta /etc/cluster/cluster.conf.bak /etc/cluster/cluster.conf`**

Appendix D. Command Line Tools Summary

Table D.1, “Command Line Tool Summary” summarizes preferred command-line tools for configuring and managing the High Availability Add-On. For more information about commands and variables, refer to the man page for each command-line tool.

Table D.1. Command Line Tool Summary

Command Line Tool	Used With	Purpose
`ccs_config_dump` — Cluster Configuration Dump Tool	Cluster Infrastructure	**`ccs_config_dump`** generates XML output of running configuration. The running configuration is, sometimes, different from the stored configuration on file because some subsystems store or set some default information into the configuration. Those values are generally not present on the on-disk version of the configuration but are required at runtime for the cluster to work properly. For more information about this tool, refer to the ccs_config_dump(8) man page.
`ccs_config_validate` — Cluster Configuration Validation Tool	Cluster Infrastructure	**`ccs_config_validate`** validates **`cluster.conf`** against the schema, **`cluster.rng`** (located in **`/usr/share/cluster/cluster.rng`** on each node. For more information about this tool, refer to the ccs_config_validate(8) man page.
`clustat` — Cluster Status Utility	High-availability Service Management Components	The **`clustat`** command displays the status of the cluster. It shows membership information, quorum view, and the state of all configured user services. For more information about this tool, refer to the clustat(8) man page.
`clusvcadm` — Cluster User Service Administration Utility	High-availability Service Management Components	The **`clusvcadm`** command allows you to enable, disable, relocate, and restart high-availability services in a cluster. For more information about this tool, refer to the clusvcadm(8) man page.
`cman_tool` — Cluster Management Tool	Cluster Infrastructure	**`cman_tool`** is a program that manages the CMAN cluster manager. It provides the capability to join a cluster, leave a cluster, kill a node, or change the expected quorum votes of a node in a cluster. For more information about this tool, refer to the cman_tool(8) man page.
`fence_tool` — Fence Tool	Cluster Infrastructure	**`fence_tool`** is a program used to join and leave the fence domain. For more information about this tool, refer to the fence_tool(8) man page.

Appendix E. Revision History

Revision 1.0	**Wed Nov 10 2010**	**Paul Kennedy** *pkennedy@redhat.com*

Initial Release

Index

A

ACPI
- configuring, 7

B

behavior, HA resources, 101

C

cluster
- administration, 5, 41, 67
- diagnosing and correcting problems, 45, 84
- starting, stopping, restarting, 67

cluster administration, 5, 41, 67
- adding cluster node, 42
- compatible hardware, 6
- configuration validation, 14
- configuring ACPI, 7
- configuring iptables, 6
- Conga considerations, 19
- considerations for using qdisk, 17
- considerations for using quorum disk, 17
- deleting a cluster, 44
- deleting a node from the configuration; adding a node to the configuration , 69
- diagnosing and correcting problems in a cluster, 45, 84
- displaying HA services with clustat, 77
- enabling IP ports, 6
- general considerations, 5
- joining a cluster, 42
- leaving a cluster, 42
- managing cluster node, 41
- managing high-availability services, 44, 76
- managing high-availability services, freeze and unfreeze, 77, 79
- network switches and multicast addresses, 18
- NetworkManager, 17
- rebooting cluster node, 42
- removing cluster node, 43
- restarting a cluster, 44
- ricci considerations, 19
- SELinux, 18
- starting a cluster, 44
- starting, stopping, restarting a cluster, 67
- stopping a cluster, 44
- updating a cluster configuration using cman_tool version -r, 80
- updating a cluster configuration using scp, 82
- updating configuration, 80

cluster configuration, 21, 47
- deleting or adding a node, 69
- updating, 80

cluster resource relationships, 102

cluster resource types, 14

cluster service managers
- configuration, 37, 61

cluster services, 37, 61
- (see also adding to the cluster configuration)

cluster software
- configuration, 21, 47

configuration
- HA service, 12

Conga
- accessing, 2
- considerations for cluster administration, 19

F

feedback, viii, viii

G

general
- considerations for cluster administration, 5

H

HA service configuration
- overview, 12

hardware
- compatible, 6

I

integrated fence devices
- configuring ACPI, 7

introduction, v
- other Red Hat Enterprise Linux documents, v

IP ports
- enabling, 6

iptables
- configuring, 6

M

multicast addresses
- considerations for using with network switches and multicast addresses, 18

N

NetworkManager
- disable for use with cluster, 17

P

parameters, fence device, 85

parameters, HA resources, 93

power controller connection, configuring, 85

power switch, 85
- (see also power controller)

Q

qdisk
  considerations for using, 17
quorum disk
  considerations for using, 17

R

relationships
  cluster resource, 102
ricci
  considerations for cluster administration, 19

S

SELinux
  configuring, 18

T

tables
  HA resources, parameters, 93
  power controller connection, configuring, 85
tools, command line, 111
troubleshooting
  diagnosing and correcting problems in a cluster, 45, 84
types
  cluster resource, 14

V

validation
  cluster configuration, 14