Red Hat Enterprise Linux 6

Cluster Suite Overview

Overview of the High Availability Add-On for Red Hat Enterprise Linux 6

Red Hat Enterprise Linux 6 Cluster Suite Overview
Overview of the High Availability Add-On for Red Hat Enterprise Linux 6
Edition 0

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

Red Hat Cluster Suite Overview provides an overview of the High Availability Add-On for Red Hat Enterprise Linux 6.

Introduction

This document provides a high-level overview of the High Availability Add-On for Red Hat Enterprise Linux 6.

Although the information in this document is an overview, you should have advanced working knowledge of Red Hat Enterprise Linux and understand the concepts of server computing to gain a good comprehension of the information.

For more information about using Red Hat Enterprise Linux, 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 this and related products for Red Hat Enterprise Linux 6, refer to the following resources:

- *Configuring and Managing the High Availability Add-On* Provides information about configuring and managing the High Availability Add-On (also known as Red Hat Cluster) for Red Hat Enterprise Linux 6.
- *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.

This document 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].

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*[2] 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] http://docs.redhat.com//
[2] https://fedorahosted.org/liberation-fonts/

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.

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.

High Availability Add-On Overview

The High Availability Add-On is a clustered system that provides reliability, scalability, and availability to critical production services. The following sections provide a high-level description of the components and functions of the High Availability Add-On:

1.1. Cluster Basics

A cluster is two or more computers (called *nodes* or *members*) that work together to perform a task. There are four major types of clusters:

- Storage
- High availability
- Load balancing
- High performance

Storage clusters provide a consistent file system image across servers in a cluster, allowing the servers to simultaneously read and write to a single shared file system. A storage cluster simplifies storage administration by limiting the installation and patching of applications to one file system. Also, with a cluster-wide file system, a storage cluster eliminates the need for redundant copies of application data and simplifies backup and disaster recovery. The High Availability Add-On provides storage clustering in conjunction with Red Hat GFS2 (part of the Resilient Storage Add-On).

High-availability clusters provide highly available services by eliminating single points of failure and by failing over services from one cluster node to another in case a node becomes inoperative. Typically, services in a high-availability cluster read and write data (via read-write mounted file systems). Therefore, a high-availability cluster must maintain data integrity as one cluster node takes over control of a service from another cluster node. Node failures in a high-availability cluster are not visible from clients outside the cluster. (High-availability clusters are sometimes referred to as failover clusters.) The High Availability Add-On provides high-availability clustering through its High Availability Service Management component, **`rgmanager`**.

Load-balancing clusters dispatch network service requests to multiple cluster nodes to balance the request load among the cluster nodes. Load balancing provides cost-effective scalability because you can match the number of nodes according to load requirements. If a node in a load-balancing cluster becomes inoperative, the load-balancing software detects the failure and redirects requests to other cluster nodes. Node failures in a load-balancing cluster are not visible from clients outside the cluster. Load balancing is available with the Load Balancer Add-On.

High-performance clusters use cluster nodes to perform concurrent calculations. A high-performance cluster allows applications to work in parallel, therefore enhancing the performance of the applications. (High performance clusters are also referred to as computational clusters or grid computing.)

Note

The cluster types summarized in the preceding text reflect basic configurations; your needs might require a combination of the clusters described.

1.2. High Availability Add-On Introduction

The High Availability Add-On is an integrated set of software components that can be deployed in a variety of configurations to suit your needs for performance, high-availability, load balancing, scalability, file sharing, and economy.

The High Availability Add-On consists of the following major components:

- Cluster infrastructure — Provides fundamental functions for nodes to work together as a cluster: configuration-file management, membership management, lock management, and fencing.
- High-availability Service Management — Provides failover of services from one cluster node to another in case a node becomes inoperative.
- Cluster administration tools — Configuration and management tools for setting up, configuring, and managing a the High Availability Add-On. The tools are for use with the Cluster Infrastructure components, the High-availability and Service Management components, and storage.

You can supplement the High Availability Add-On with the following components:

- Red Hat GFS2 (Global File System 2) — Part of the Resilient Storage Add-On, this provides a cluster file system for use with the High Availability Add-On. GFS2 allows multiple nodes to share storage at a block level as if the storage were connected locally to each cluster node.
- Cluster Logical Volume Manager (CLVM) — Part of the Resilient Storage Add-On, this provides volume management of cluster storage.
- Load Balancer Add-On — Routing software that provides IP-Load-balancing. the Load Balancer Add-On runs in a pair of redundant virtual servers that distributes client requests evenly to real servers that are behind the virtual servers.

1.3. Cluster Infrastructure

The High Availability Add-On cluster infrastructure provides the basic functions for a group of computers (called *nodes* or *members*) to work together as a cluster. Once a cluster is formed using the cluster infrastructure, you can use other components to suit your clustering needs (for example, setting up a cluster for sharing files on a GFS2 file system or setting up service failover). The cluster infrastructure performs the following functions:

- Cluster management
- Lock management
- Fencing
- Cluster configuration management

1.3.1. Cluster Management

Cluster management manages cluster quorum and cluster membership. CMAN (an abbreviation for cluster manager) performs cluster management in the High Availability Add-On for Red Hat Enterprise

Linux 6. CMAN is a distributed cluster manager and runs in each cluster node; cluster management is distributed across all nodes in the cluster.

CMAN keeps track of cluster quorum by monitoring the count of cluster nodes. If more than half the nodes are active, the cluster has quorum. If half the nodes (or fewer) are active, the cluster does not have quorum, and all cluster activity is stopped. Cluster quorum prevents the occurrence of a "split-brain" condition — a condition where two instances of the same cluster are running. A split-brain condition would allow each cluster instance to access cluster resources without knowledge of the other cluster instance, resulting in corrupted cluster integrity.

Quorum is determined by communication of messages among cluster nodes via Ethernet. Optionally, quorum can be determined by a combination of communicating messages via Ethernet *and* through a quorum disk. For quorum via Ethernet, quorum consists of 50 percent of the node votes plus 1. For quorum via quorum disk, quorum consists of user-specified conditions.

Note

By default, each node has one quorum vote. Optionally, you can configure each node to have more than one vote.

CMAN keeps track of membership by monitoring messages from other cluster nodes. When cluster membership changes, the cluster manager notifies the other infrastructure components, which then take appropriate action. If a cluster node does not transmit a message within a prescribed amount of time, the cluster manager removes the node from the cluster and communicates to other cluster infrastructure components that the node is not a member. Again, other cluster infrastructure components determine what actions to take upon notification that node is no longer a cluster member. For example, Fencing would fence the node that is no longer a member.

1.3.2. Lock Management

Lock management is a common cluster-infrastructure service that provides a mechanism for other cluster infrastructure components to synchronize their access to shared resources. In a Red Hat cluster, DLM (Distributed Lock Manager) is the lock manager. As implied in its name, DLM is a distributed lock manager and runs in each cluster node; lock management is distributed across all nodes in the cluster. GFS2 and CLVM use locks from the lock manager. GFS2 uses locks from the lock manager to synchronize access to file system metadata (on shared storage). CLVM uses locks from the lock manager to synchronize updates to LVM volumes and volume groups (also on shared storage). In addition, **`rgmanager`** uses DLM to synchronize service states.

1.3.3. Fencing

Fencing is the disconnection of a node from the cluster's shared storage. Fencing cuts off I/O from shared storage, thus ensuring data integrity. The cluster infrastructure performs fencing through the fence daemon, **`fenced`**.

When CMAN determines that a node has failed, it communicates to other cluster-infrastructure components that the node has failed. **`fenced`**, when notified of the failure, fences the failed node. Other cluster-infrastructure components determine what actions to take — that is, they perform any recovery that needs to done. For example, DLM and GFS2, when notified of a node failure, suspend activity until they detect that **`fenced`** has completed fencing the failed node. Upon confirmation that the failed node is fenced, DLM and GFS2 perform recovery. DLM releases locks of the failed node; GFS2 recovers the journal of the failed node.

The fencing program determines from the cluster configuration file which fencing method to use. Two key elements in the cluster configuration file define a fencing method: fencing agent and fencing device. The fencing program makes a call to a fencing agent specified in the cluster configuration file. The fencing agent, in turn, fences the node via a fencing device. When fencing is complete, the fencing program notifies the cluster manager.

The High Availability Add-On provides a variety of fencing methods:

- Power fencing — A fencing method that uses a power controller to power off an inoperable node.
- Fibre Channel switch fencing — A fencing method that disables the Fibre Channel port that connects storage to an inoperable node.
- Other fencing — Several other fencing methods that disable I/O or power of an inoperable node, including IBM Bladecenters, PAP, DRAC/MC, HP ILO, IPMI, IBM RSA II, and others.

Figure 1.1, "Power Fencing Example" shows an example of power fencing. In the example, the fencing program in node A causes the power controller to power off node D. *Figure 1.2, "Fibre Channel Switch Fencing Example"* shows an example of Fibre Channel switch fencing. In the example, the fencing program in node A causes the Fibre Channel switch to disable the port for node D, disconnecting node D from storage.

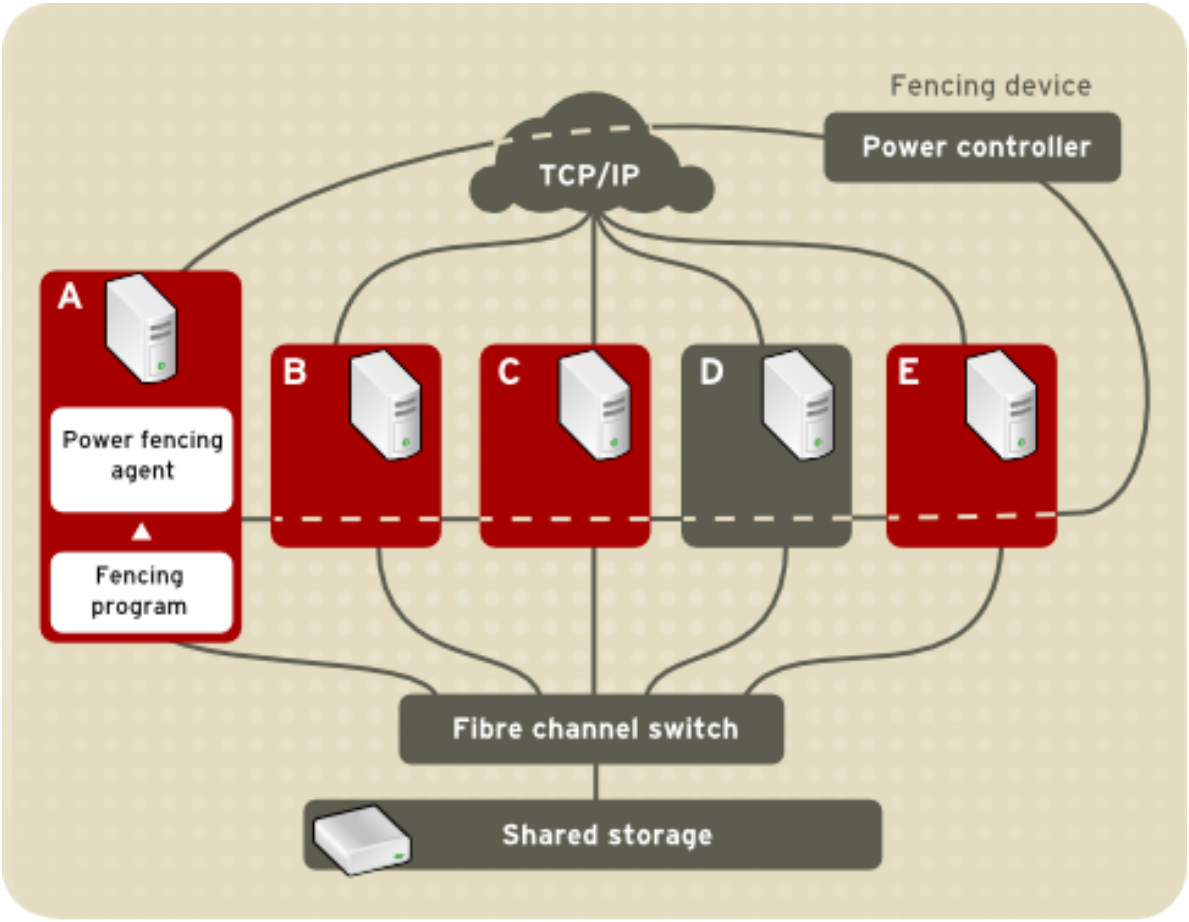

Figure 1.1. Power Fencing Example

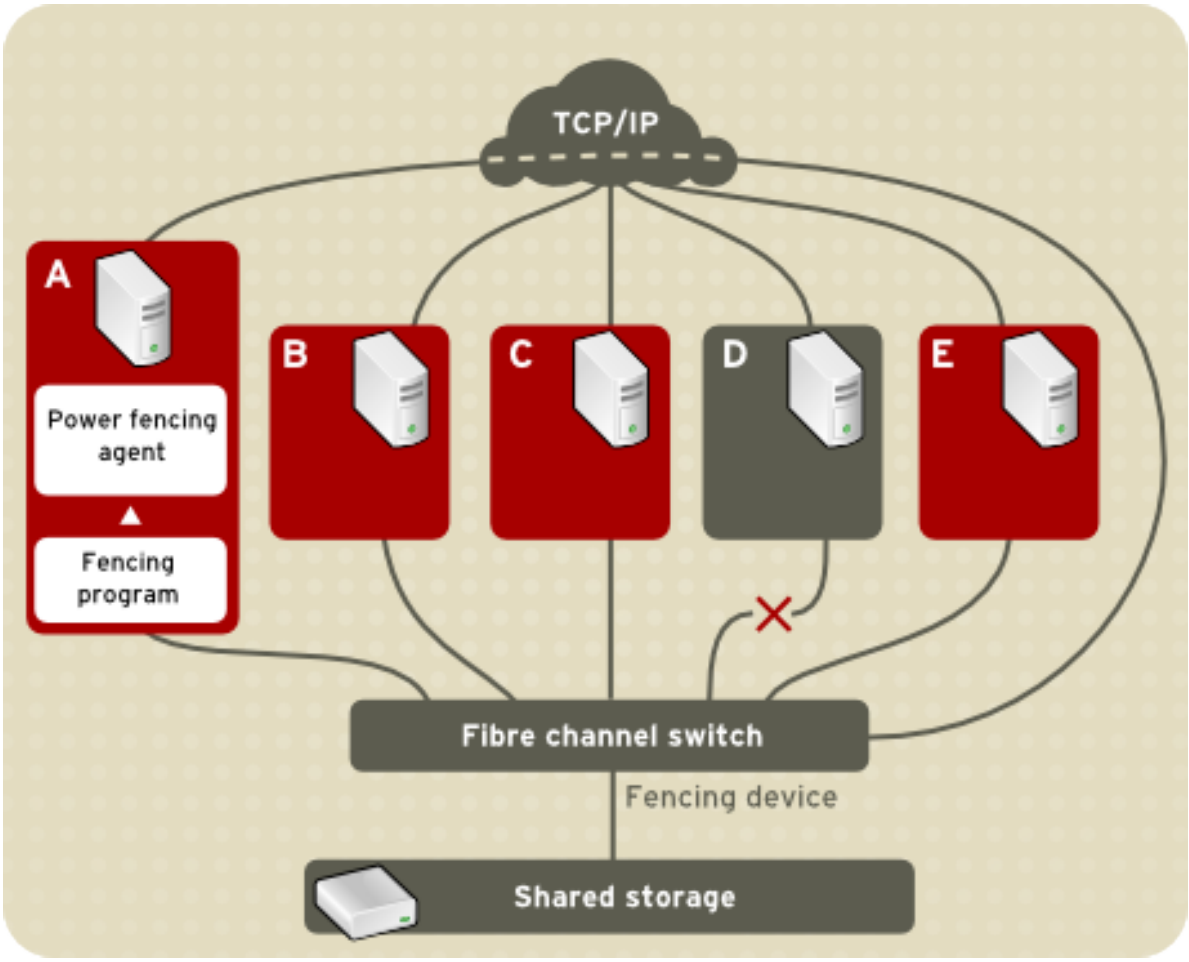

Figure 1.2. Fibre Channel Switch Fencing Example

Specifying a fencing method consists of editing a cluster configuration file to assign a fencing-method name, the fencing agent, and the fencing device for each node in the cluster.

The way in which a fencing method is specified depends on if a node has either dual power supplies or multiple paths to storage. If a node has dual power supplies, then the fencing method for the node must specify at least two fencing devices — one fencing device for each power supply (refer to *Figure 1.3, "Fencing a Node with Dual Power Supplies"*). Similarly, if a node has multiple paths to Fibre Channel storage, then the fencing method for the node must specify one fencing device for each path to Fibre Channel storage. For example, if a node has two paths to Fibre Channel storage, the fencing method should specify two fencing devices — one for each path to Fibre Channel storage (refer to *Figure 1.4, "Fencing a Node with Dual Fibre Channel Connections"*).

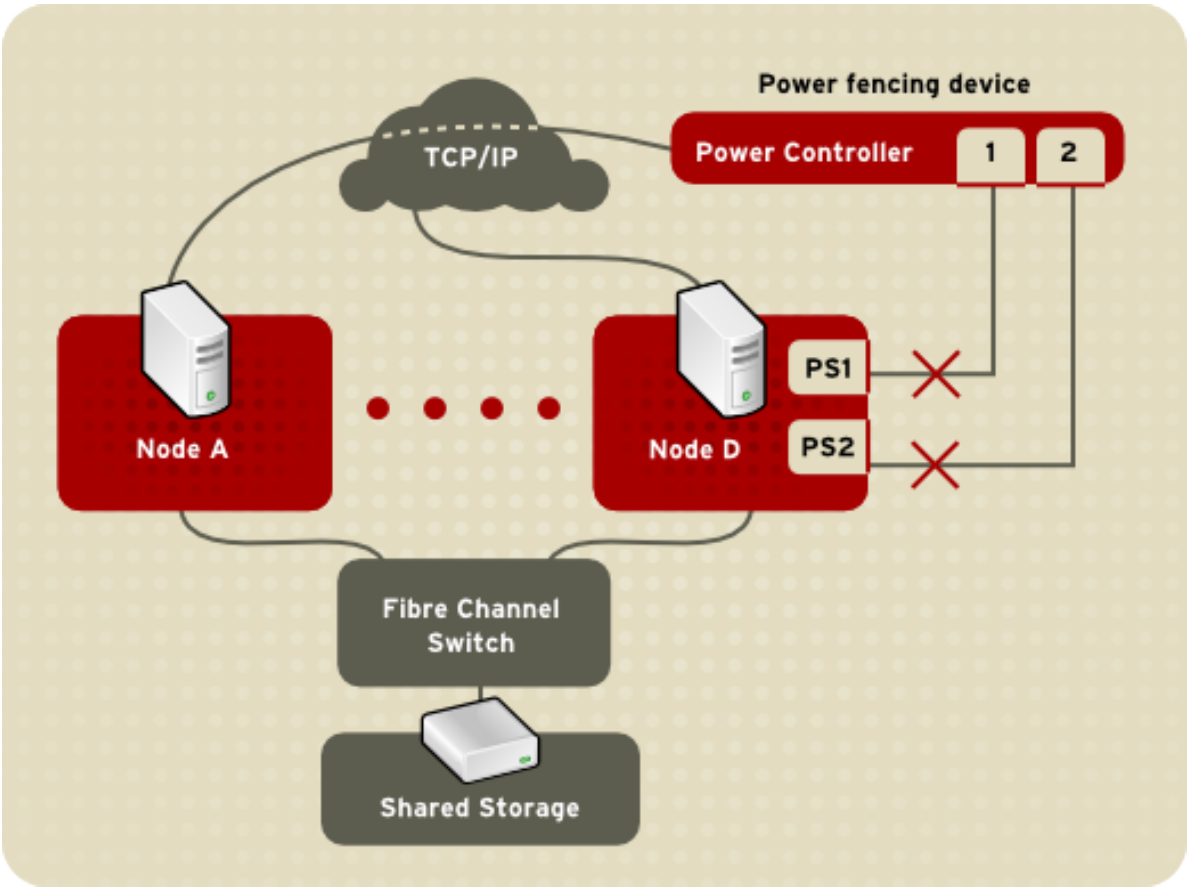

Figure 1.3. Fencing a Node with Dual Power Supplies

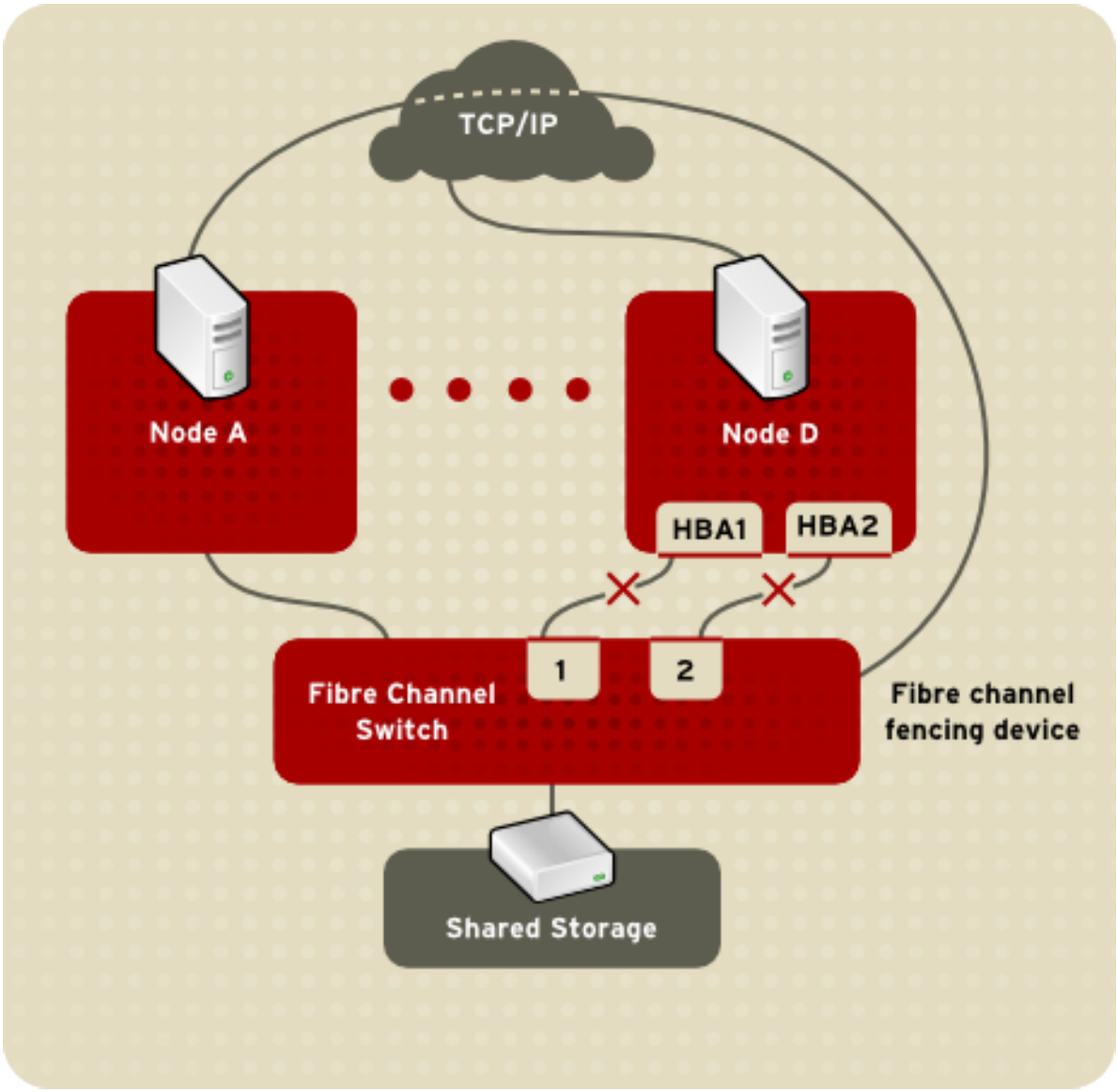

Figure 1.4. Fencing a Node with Dual Fibre Channel Connections

You can configure a node with one fencing method or multiple fencing methods. When you configure a node for one fencing method, that is the only fencing method available for fencing that node. When you configure a node for multiple fencing methods, the fencing methods are *cascaded* from one fencing method to another according to the order of the fencing methods specified in the cluster configuration file. If a node fails, it is fenced using the first fencing method specified in the cluster configuration file for that node. If the first fencing method is not successful, the next fencing method specified for that node is used. If none of the fencing methods is successful, then fencing starts again with the first fencing method specified, and continues looping through the fencing methods in the order specified in the cluster configuration file until the node has been fenced.

1.3.4. Configuration Management

The cluster configuration file, **`/etc/cluster/cluster.conf`** specifies the High Availability Add-On configuration.The configuration file is an XML file that describes the following cluster characteristics:

- Cluster name — Specifies the cluster name, cluster configuration file revision level, and basic fence timing properties used when a node joins a cluster or is fenced from the cluster.

- Cluster — Specifies each node of the cluster, specifying node name, node ID, number of quorum votes, and fencing method for that node.
- Fence Device — Specifies fence devices in the cluster. Parameters vary according to the type of fence device. For example for a power controller used as a fence device, the cluster configuration defines the name of the power controller, its IP address, login, and password.
- Managed Resources — Specifies resources required to create cluster services. Managed resources includes the definition of failover domains, resources (for example an IP address), and services. Together the managed resources define cluster services and failover behavior of the cluster services.

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).

1.4. High-availability Service Management

High-availability service management provides the ability to create and manage high-availability *cluster services* in the High Availability Add-On. The key component for high-availability service management in the High Availability Add-On, **`rgmanager`**, implements cold failover for off-the-shelf applications. In the High Availability Add-On, an application is configured with other cluster resources to form a high-availability cluster service. A high-availability cluster service can fail over from one cluster node to another with no apparent interruption to cluster clients. Cluster-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 a high-availability service, you must configure it in the cluster configuration file. A cluster 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.

You can associate a cluster service with a *failover domain*. A failover domain is a subset of cluster nodes that are eligible to run a particular cluster service (refer to *Figure 1.5, “Failover Domains”*).

Note

Failover domains are *not* required for operation.

A cluster 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 a cluster service should fail over to. If you do not specify failover priority, a cluster service can fail over to any node in its failover domain. Also, you can specify if a cluster service is restricted

to run only on nodes of its associated failover domain. (When associated with an unrestricted failover domain, a cluster service can start on any cluster node in the event no member of the failover domain is available.)

In *Figure 1.5, "Failover Domains"*, Failover Domain 1 is configured to restrict failover within that domain; therefore, Cluster Service X can only fail over between Node A and Node B. Failover Domain 2 is also configured to restrict failover with its domain; additionally, it is configured for failover priority. Failover Domain 2 priority is configured with Node C as priority 1, Node B as priority 2, and Node D as priority 3. If Node C fails, Cluster Service Y fails over to Node B next. If it cannot fail over to Node B, it tries failing over to Node D. Failover Domain 3 is configured with no priority and no restrictions. If the node that Cluster Service Z is running on fails, Cluster Service Z tries failing over to one of the nodes in Failover Domain 3. However, if none of those nodes is available, Cluster Service Z can fail over to any node in the cluster.

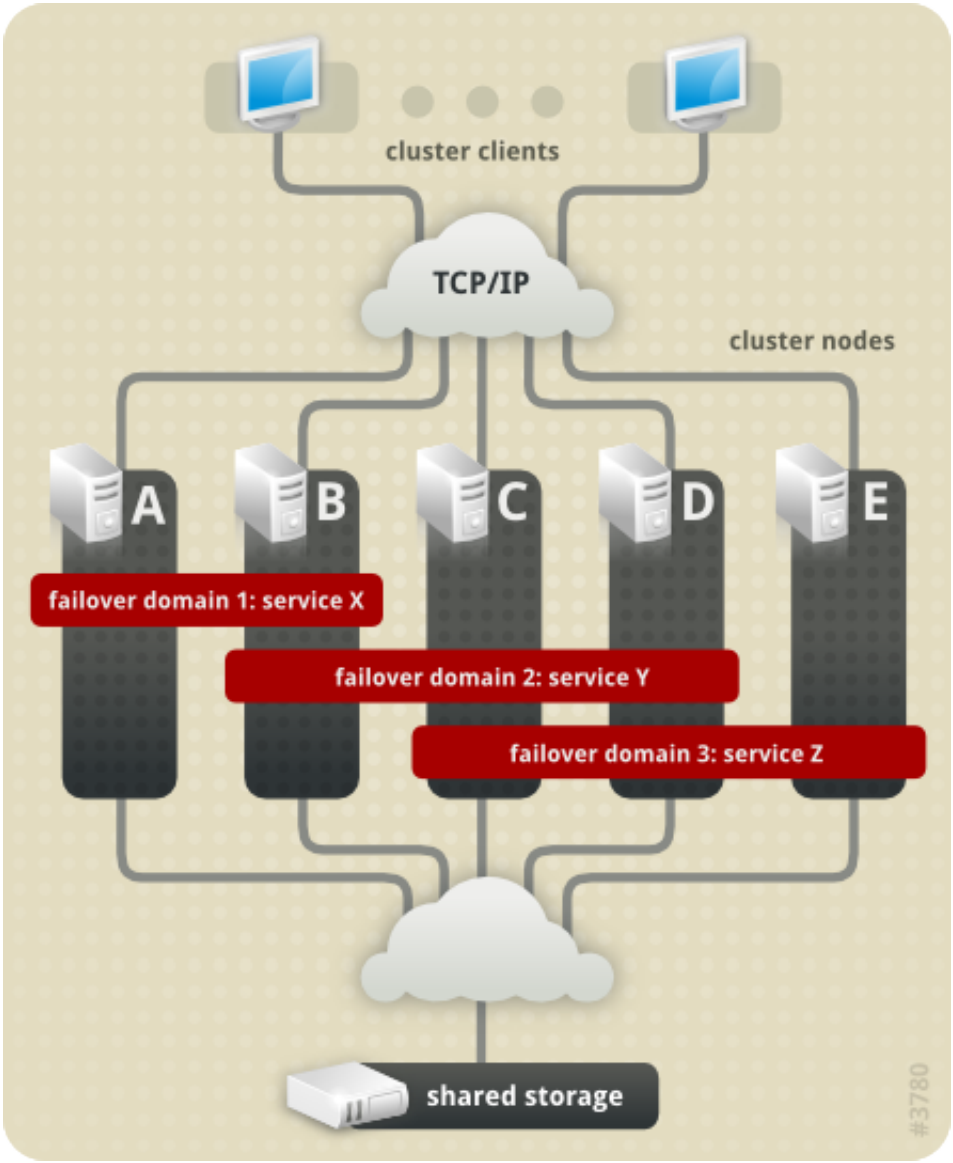

Figure 1.5. Failover Domains

Figure 1.6, "Web Server Cluster Service Example" shows an example of a high-availability cluster 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 cluster 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".

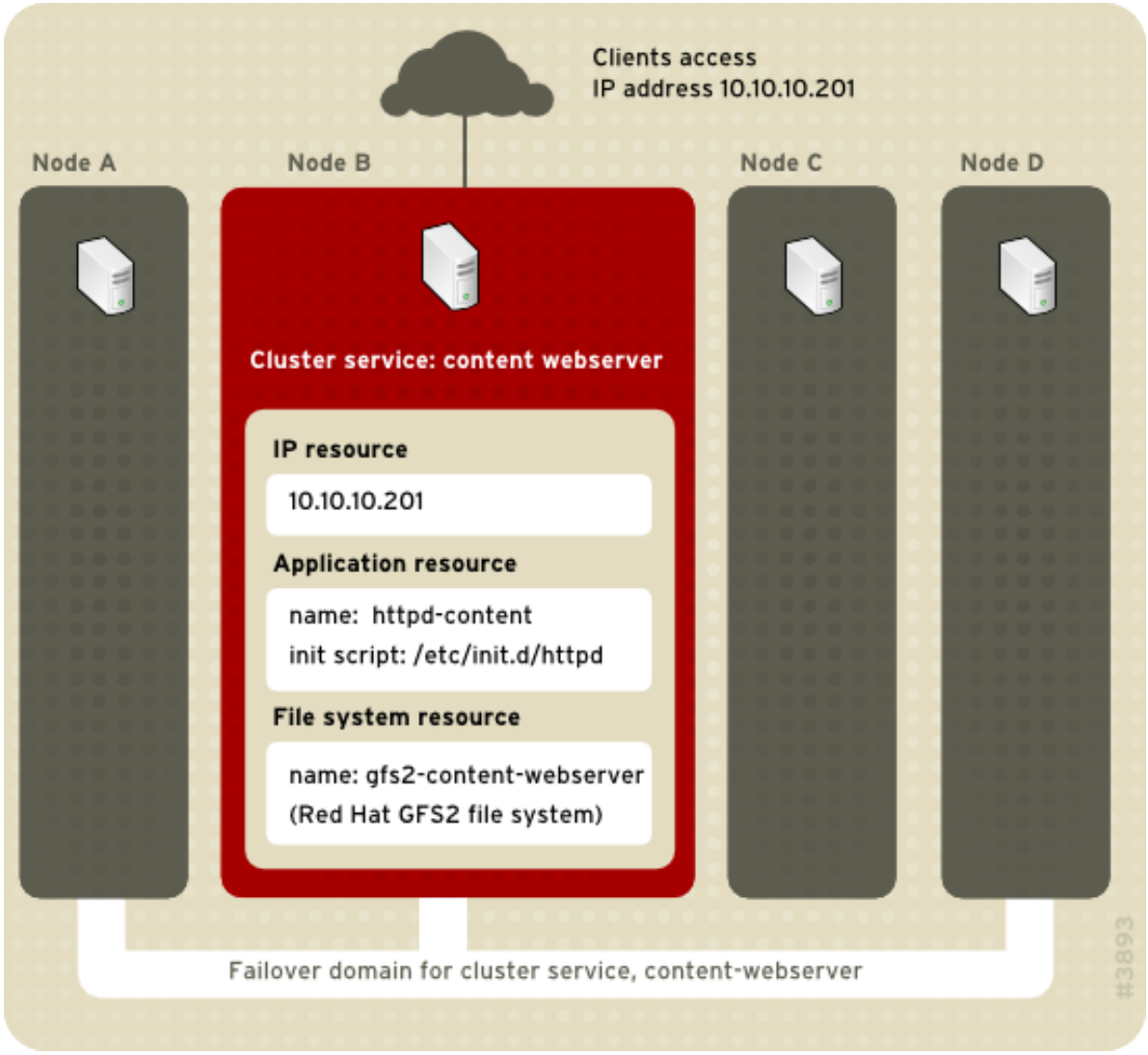

Figure 1.6. Web Server Cluster Service Example

Clients access the cluster 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 cluster 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 no apparent interruption to the cluster clients. The cluster service would be accessible from another cluster node via the same IP address as it was before failover.

1.5. Cluster Administration Tools

Managing 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 *Configuring and Managing the High Availability Add-On* 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 *Configuring and Managing the High Availability Add-On* for information about configuring and managing a cluster with command-line tools.

Note

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

Appendix A. Revision History

Revision 1.0 **Wed Nov 10 2010** **Paul Kennedy** *pkennedy@redhat.com*

Initial Release