Red Hat Cluster Manager

The Red Hat Cluster Manager Installation and Administration Guide

ISBN: N/A


Red Hat, Inc.

1801 Varsity Drive
Raleigh, NC 27606 USA
+1 919 754 3700 (Voice)
+1 919 754 3701 (FAX)
888 733 4281 (Voice)
P.O. Box 13588
Research Triangle Park, NC 27709 USA

rh-cm(EN)-1.0-Print-RHI (2002-05-17T11:40-0400)

Acknowledgments

The Red Hat Cluster Manager software was originally based on the open source Kimberlite http://oss.missioncriticallinux.com/kimberlite/ cluster project which was developed by Mission Critical Linux, Inc.

Subsequent to its inception based on Kimberlite, developers at Red Hat have made a large number of enhancements and modifications. The following is a non-comprehensive list highlighting some of these enhancements.

- Packaging and integration into the Red Hat installation paradigm in order to simplify the end user's experience.
- Addition of support for high availability NFS services.
- Addition of support for high availability Samba services.
- Addition of support for using watchdog timers as a data integrity provision
- Addition of service monitoring which will automatically restart a failed application.
- Rewrite of the service manager to facilitate additional cluster-wide operations.
- Addition of the Red Hat Cluster Manager GUI, a graphical monitoring tool.
- A set of miscellaneous bug fixes.

The Red Hat Cluster Manager software incorporates STONITH compliant power switch modules from the Linux-HA project http://www.linux-ha.org/stonith/.

Contents

Red Hat Cluster Manager

1 Introduction to Red Hat Cluster Manager

The Red Hat Cluster Manager is a collection of technologies working together to provide data integrity and the ability to maintain application availability in the event of a failure. Using redundant hardware, shared disk storage, power management, and robust cluster communication and application failover mechanisms, a cluster can meet the needs of the enterprise market.

Specially suited for database applications, network file servers, and World Wide Web (Web) servers with dynamic content, a cluster can also be used in conjunction with the Piranha load balancing cluster software, based on the Linux Virtual Server (LVS) project, to deploy a highly available e-commerce site that has complete data integrity and application availability, in addition to load balancing capabilities. See Section B.5, *Using Red Hat Cluster Manager with Piranha* for more information.

1.1 Cluster Overview

To set up a cluster, an administrator must connect the **cluster systems** (often referred to as **member systems**) to the cluster hardware, and configure the systems into the cluster environment. The foundation of a cluster is an advanced host membership algorithm. This algorithm ensures that the cluster maintains complete data integrity at all times by using the following methods of inter-node communication:

- **Quorum partitions** on shared disk storage to hold system status
- Ethernet and serial connections between the cluster systems for **heartbeat** channels

To make an application and data highly available in a cluster, the administrator must configure a **cluster service** — a discrete group of service properties and resources, such as an application and shared disk storage. A service can be assigned an IP address to provide transparent client access to the service. For example, an administrator can set up a cluster service that provides clients with access to highly-available database application data.

Both cluster systems can run any service and access the service data on shared disk storage. However, each service can run on only one cluster system at a time, in order to maintain data integrity. Administrators can set up an **active-active configuration** in which both cluster systems run different services, or a **hot-standby configuration** in which a primary cluster system runs all the services, and a backup cluster system takes over only if the primary system fails.

Figure 1–1 Example Cluster

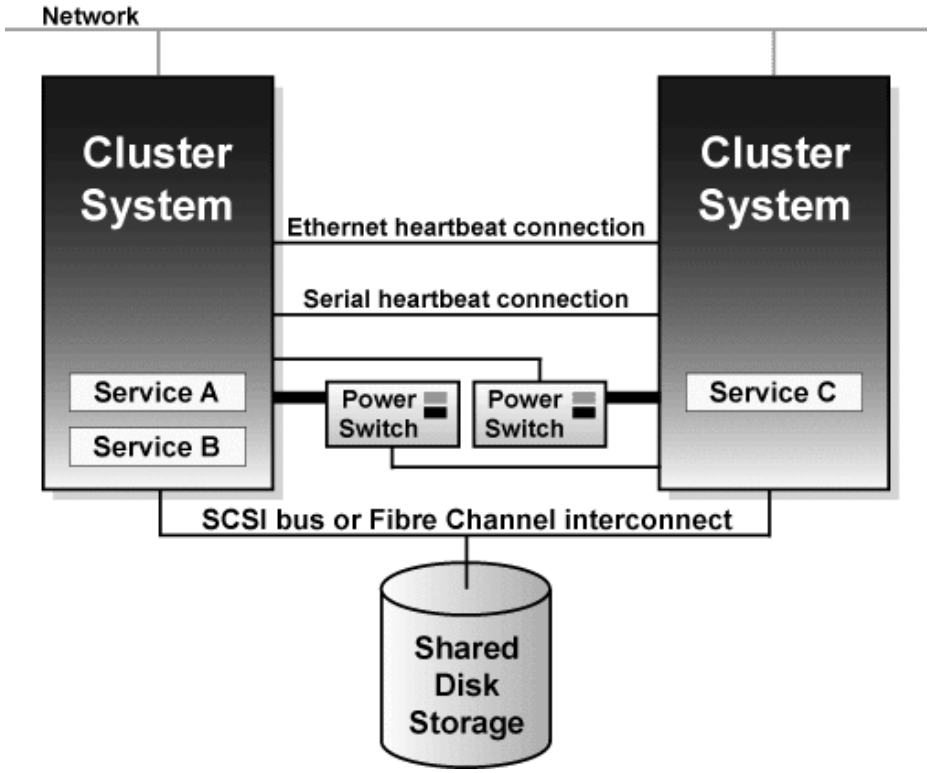


Figure 1–1, *Example Cluster* shows an example of a cluster in an active-active configuration.

If a hardware or software failure occurs, the cluster will automatically restart the failed system's services on the functional cluster system. This **service failover** capability ensures that no data is lost, and there is little disruption to users. When the failed system recovers, the cluster can re-balance the services across the two systems.

In addition, a cluster administrator can cleanly stop the services running on a cluster system and then restart them on the other system. This **service relocation** capability enables the administrator to maintain application and data availability when a cluster system requires maintenance.

1.2 Cluster Features

A cluster includes the following features:

- No-single-point-of-failure hardware configuration

  Clusters can include a dual-controller RAID array, multiple network and serial communication channels, and redundant uninterruptible power supply (UPS) systems to ensure that no single failure results in application down time or loss of data.

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

  Note

  Certain low-cost alternatives, such as software RAID and multi-initiator parallel SCSI, are not compatible or appropriate for use on the shared cluster storage. Refer to Section 2.1, *Choosing a Hardware Configuration*, for more information.

- Service configuration framework

  Clusters enable an administrator to easily configure individual services to make data and applications highly available. To create a service, an administrator specifies the resources used in the service and properties for the service, including the service name, application start and stop script, disk partitions, mount points, and the cluster system on which an administrator prefers to run the service. After the administrator adds a service, the cluster enters the information into the cluster database on shared storage, where it can be accessed by both cluster systems.

  The cluster provides an easy-to-use framework for database applications. For example, a **database service** serves highly-available data to a database application. The application running on a cluster system provides network access to database client systems, such as Web servers. If the service fails over to another cluster system, the application can still access the shared database data. A network-accessible database service is usually assigned an IP address, which is failed over along with the service to maintain transparent access for clients.

  The cluster service framework can be easily extended to other applications, as well.

- Data integrity assurance

  To ensure data integrity, only one cluster system can run a service and access service data at one time. Using power switches in the cluster configuration enable each cluster system to power-cycle the other cluster system before restarting its services during the failover process. This prevents

the two systems from simultaneously accessing the same data and corrupting it. Although not required, it is recommended that power switches are used to guarantee data integrity under all failure conditions. Watchdog timers are an optional variety of power control to ensure correct operation of service failover.

- Cluster administration user interface

  A user interface simplifies cluster administration and enables an administrator to easily create, start, stop, relocate services, and monitor the cluster.

- Multiple cluster communication methods

  To monitor the health of the other cluster system, each cluster system monitors the health of the remote power switch, if any, and issues heartbeat pings over network and serial channels to monitor the health of the other cluster system. In addition, each cluster system periodically writes a timestamp and cluster state information to two quorum partitions located on shared disk storage. System state information includes whether the system is an active cluster member. Service state information includes whether the service is running and which cluster system is running the service. Each cluster system checks to ensure that the other system's status is up to date.

  To ensure correct cluster operation, if a system is unable to write to both quorum partitions at startup time, it will not be allowed to join the cluster. In addition, if a cluster system is not updating its timestamp, and if heartbeats to the system fail, the cluster system will be removed from the cluster.

Figure 1–2 Cluster Communication Mechanisms

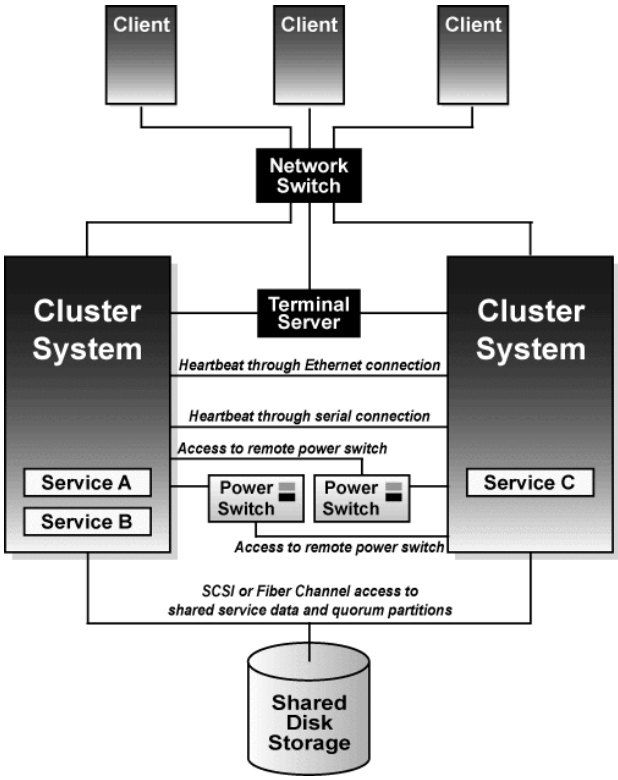


Figure 1–2, *Cluster Communication Mechanisms* shows how systems communicate in a cluster configuration. Note that the terminal server used to access system consoles via serial ports is not a required cluster component.

- Service failover capability

  If a hardware or software failure occurs, the cluster will take the appropriate action to maintain application availability and data integrity. For example, if a cluster system completely fails, the other cluster system will restart its services. Services already running on this system are not disrupted.

  When the failed system reboots and is able to write to the quorum partitions, it can rejoin the cluster and run services. Depending on how the services are configured, the cluster can re-balance the services across the two cluster systems.

- Manual service relocation capability

  In addition to automatic service failover, a cluster enables administrators to cleanly stop services on one cluster system and restart them on the other system. This allows administrators to perform planned maintenance on a cluster system, while providing application and data availability.

- Event logging facility

  To ensure that problems are detected and resolved before they affect service availability, the cluster daemons log messages by using the conventional Linux syslog subsystem. Administrators can customize the severity level of the logged messages.

- Application Monitoring

  The cluster services infrastructure can optionally monitor the state and health of an application. In this manner, should an application-specific failure occur, the cluster will automatically restart the application. In response to the application failure, the application will attempt to be restarted on the member it was initially running on; failing that, it will restart on the other cluster member.

- Status Monitoring Agent

  A cluster status monitoring agent is used to gather vital cluster and application state information. This information is then accessible both locally on the cluster member as well as remotely. A graphical user interface can then display status information from multiple clusters in a manner which does not degrade system performance.

1.3 How To Use This Manual

This manual contains information about setting up the cluster hardware, and installing the Linux distribution and the cluster software. These tasks are described in Chapter 2, *Hardware Installation and Operating System Configuration* and Chapter 3, *Cluster Software Installation and Configuration*.

For information about setting up and managing cluster services, see Chapter 4, *Service Configuration and Administration*. For information about managing a cluster, see Chapter 8, *Cluster Administration*.

Appendix A, *Supplementary Hardware Information* contains detailed configuration information on specific hardware devices and shared storage configurations. Appendix B, *Supplementary Software Information* contains background information on the cluster software and other related information.

2 Hardware Installation and Operating System Configuration

To set up the hardware configuration and install the Linux distribution, follow these steps:

- Choose a cluster hardware configuration that meets the needs of applications and users, see Section 2.1, *Choosing a Hardware Configuration*.
- Set up and connect the cluster systems and the optional console switch and network switch or hub, see Section 2.2, *Steps for Setting Up the Cluster Systems*.
- Install and configure the Linux distribution on the cluster systems, see Section 2.3, *Steps for Installing and Configuring the Red Hat Linux Distribution*.
- Set up the remaining cluster hardware components and connect them to the cluster systems, see Section 2.4, *Steps for Setting Up and Connecting the Cluster Hardware*.

After setting up the hardware configuration and installing the Linux distribution, installing the cluster software is possible.

2.1 Choosing a Hardware Configuration

The Red Hat Cluster Manager allows administrators to use commodity hardware to set up a cluster configuration that will meet the performance, availability, and data integrity needs of applications and users. Cluster hardware ranges from low-cost minimum configurations that include only the components required for cluster operation, to high-end configurations that include redundant heartbeat channels, hardware RAID, and power switches.

Regardless of configuration, the use of high-quality hardware in a cluster is recommended, as hardware malfunction is the primary cause of system down time.

Although all cluster configurations provide availability, some configurations protect against every single point of failure. In addition, all cluster configurations provide data integrity, but some configurations protect data under every failure condition. Therefore, administrators must fully understand the needs of their computing environment and also the availability and data integrity features of different hardware configurations in order to choose the cluster hardware that will meet the proper requirements.

When choosing a cluster hardware configuration, consider the following:

Performance requirements of applications and users

Choose a hardware configuration that will provide adequate memory, CPU, and I/O resources. Be sure that the configuration chosen will be able to handle any future increases in workload, as well.

Cost restrictions

The hardware configuration chosen must meet budget requirements. For example, systems with multiple I/O ports usually cost more than low-end systems with less expansion capabilities.

Availability requirements

If a computing environment requires the highest degree of availability, such as a production environment, then a cluster hardware configuration that protects against all single points of failure, including disk, storage interconnect, heartbeat channel, and power failures is recommended. Environments that can tolerate an interruption in availability, such as development environments, may not require as much protection. See Section 2.4.1, *Configuring Heartbeat Channels*, Section 2.4.3, *Configuring UPS Systems*, and Section 2.4.4, *Configuring Shared Disk Storage* for more information about using redundant hardware for high availability.

Data integrity under all failure conditions requirement

Using power switches in a cluster configuration guarantees that service data is protected under every failure condition. These devices enable a cluster system to power cycle the other cluster system before restarting its services during failover. Power switches protect against data corruption if an unresponsive (or hanging) system becomes responsive after its services have failed over, and then issues I/O to a disk that is also receiving I/O from the other cluster system.

In addition, if a quorum daemon fails on a cluster system, the system is no longer able to monitor the quorum partitions. If you are not using power switches in the cluster, this error condition may result in services being run on more than one cluster system, which can cause data corruption. See Section 2.4.2, *Configuring Power Switches* for more information about the benefits of using power switches in a cluster. It is recommended that production environments use power switches or watchdog timers in the cluster configuration.

2.1.1 Shared Storage Requirements

The operation of the cluster depends on reliable, coordinated access to shared storage. In the event of hardware failure, it is desirable to be able to disconnect one member from the shared storage for repair without disrupting the other member. Shared storage is truly vital to the cluster configuration.

Testing has shown that it is difficult, if not impossible, to configure reliable multi-initiator parallel SCSI configurations at data rates above 80 MBytes/sec. using standard SCSI adapters. Further tests have shown that these configurations can not support online repair because the bus does not work reliably when the HBA terminators are disabled, and external terminators are used. For these reasons, multi-initiator SCSI configurations using standard adapters are not supported. Single-initiator parallel SCSI buses, connected to multi-ported storage devices, or Fibre Channel, are required.

The Red Hat Cluster Manager requires that both cluster members have simultaneous access to the shared storage. Certain host RAID adapters are capable of providing this type of access to shared

RAID units. These products require extensive testing to ensure reliable operation, especially if the shared RAID units are based on parallel SCSI buses. These products typically do not allow for online repair of a failed system. No host RAID adapters are currently certified with Red Hat Cluster Manager. Refer to the Red Hat web site at http://www.redhat.com for the most up-to-date supported hardware matrix.

The use of software RAID, or software Logical Volume Management (LVM), is not supported on shared storage. This is because these products do not coordinate access from multiple hosts to shared storage. Software RAID or LVM may be used on non-shared storage on cluster members (for example, boot and system partitions and other filesysytems which are not associated with any cluster services).

2.1.2 Minimum Hardware Requirements

A **minimum hardware configuration** includes only the hardware components that are required for cluster operation, as follows:

- Two servers to run cluster services
- Ethernet connection for a heartbeat channel and client network access
- Shared disk storage for the cluster quorum partitions and service data.

See Section 2.1.5, *Example of a Minimum Cluster Configuration* for an example of this type of hardware configuration.

The minimum hardware configuration is the most cost-effective cluster configuration; however, it includes multiple points of failure. For example, if the RAID controller fails, then all cluster services will be unavailable. When deploying the minimal hardware configuration, software watchdog timers should be configured as a data integrity provision.

To improve availability, protect against component failure, and guarantee data integrity under all failure conditions, the minimum configuration can be expanded. Table 2–1, *Improving Availability and Guaranteeing Data Integrity* shows how to improve availability and guarantee data integrity:

Table 2–1 Improving Availability and Guaranteeing Data Integrity

Problem	Solution
Disk failure	Hardware RAID to replicate data across multiple disks.
RAID controller failure	Dual RAID controllers to provide redundant access to disk data.
Heartbeat channel failure	Point-to-point Ethernet or serial connection between the cluster systems.

Problem	Solution
Power source failure	Redundant uninterruptible power supply (UPS) systems.
Data corruption under all failure conditions	Power switches or hardware-based watchdog timers

A no-single-point-of-failure hardware configuration that guarantees data integrity under all failure conditions can include the following components:

- Two servers to run cluster services
- Ethernet connection between each system for a heartbeat channel and client network access
- Dual-controller RAID array to replicate quorum partitions and service data
- Two power switches to enable each cluster system to power-cycle the other system during the failover process
- Point-to-point Ethernet connection between the cluster systems for a redundant Ethernet heartbeat channel
- Point-to-point serial connection between the cluster systems for a serial heartbeat channel
- Two UPS systems for a highly-available source of power

See Section 2.1.6, *Example of a No-Single-Point-Of-Failure Configuration* for an example of this type of hardware configuration.

Cluster hardware configurations can also include other optional hardware components that are common in a computing environment. For example, a cluster can include a **network switch** or **network hub**, which enables the connection of the cluster systems to a network. A cluster may also include a **console switch**, which facilitates the management of multiple systems and eliminates the need for separate monitors, mouses, and keyboards for each cluster system.

One type of console switch is a **terminal server**, which enables connection to serial consoles and management of many systems from one remote location. As a low-cost alternative, you can use a **KVM** (keyboard, video, and mouse) switch, which enables multiple systems to share one keyboard, monitor, and mouse. A KVM is suitable for configurations in which access to a graphical user interface (GUI) to perform system management tasks is preferred.

When choosing a cluster system, be sure that it provides the PCI slots, network slots, and serial ports that the hardware configuration requires. For example, a no-single-point-of-failure configuration requires multiple serial and Ethernet ports. Ideally, choose cluster systems that have at least two serial ports. See Section 2.2.1, *Installing the Basic System Hardware* for more information.

2.1.3 Choosing the Type of Power Controller

The Red Hat Cluster Manager implementation consists of a generic power management layer and a set of device specific modules which accommodate a range of power management types. When selecting the appropriate type of power controller to deploy in the cluster, it is important to recognize the implications of specific device types. The following describes the types of supported power switches followed by a summary table. For a more detailed description of the role a power switch plays to ensure data integrity, refer to Section 2.4.2, *Configuring Power Switches*.

Serial- and Network-attached power switches are separate devices which enable one cluster member to power cycle another member. They resemble a power plug strip on which individual outlets can be turned on and off under software control through either a serial or network cable.

Watchdog timers provide a means for failed systems to remove themselves from the cluster prior to another system taking over its services, rather than allowing one cluster member to power cycle another. The normal operational mode for watchdog timers is that the cluster software must periodically reset a timer prior to its expiration. If the cluster software fails to reset the timer, the watchdog will trigger under the assumption that the system may have hung or otherwise failed. The healthy cluster member allows a window of time to pass prior to concluding that another cluster member has failed (by default, this window is 12 seconds). The watchdog timer interval must be less than the duration of time for one cluster member to conclude that another has failed. In this manner, a healthy system can assume that prior to taking over services for a failed cluster member, that it has safely removed itself from the cluster (by rebooting) and therefore is no risk to data integrity. The underlying watchdog support is included in the core Linux kernel. Red Hat Cluster Manager utilizes these watchdog features via its standard APIs and configuration mechanism.

There are two types of watchdog timers: Hardware-based and software-based. Hardware-based watchdog timers typically consist of system board components such as the Intel® i810 TCO chipset. This circuitry has a high degree of independence from the main system CPU. This independence is beneficial in failure scenarios of a true system hang, as in this case it will pull down the system's reset lead resulting in a system reboot. There are some PCI expansion cards that provide watchdog features.

The second type of watchdog timer is software-based. This category of watchdog does not have any dedicated hardware. The implementation is a kernel thread which is periodically run and if the timer duration has expired will initiate a system reboot. The vulnerability of the software watchdog timer is that under certain failure scenarios such as system hangs while interrupts are blocked, the kernel thread will not be called. As a result, in such conditions it can not be definitively depended on for data integrity. This can cause the healthy cluster member to take over services for a hung node which could cause data corruption under certain scenarios.

Finally, administrators can choose not to employ a power controller at all. If choosing the "None" type, note that there are no provisions for a cluster member to power cycle a failed member. Similarly, the failed member can not be guaranteed to reboot itself under all failure conditions. Deploying clusters

with a power controller type of "None" is useful for simple evaluation purposes, but because it affords the weakest data integrity provisions, it is not recommended for usage in a production environment.

Ultimately, the right type of power controller deployed in a cluster environment depends on the data integrity requirements weighed against the cost and availability of external power switches.

Table 2–2, *Power Switches* summarizes the types of supported power management modules and discusses their advantages and disadvantages individually.

Table 2–2 Power Switches

Type	Notes	Pros	Cons
Serial-attached power switches	Two serial attached power controllers are used in a cluster (one per member system)	Affords strong data integrity guarantees. the power controller itself is not a single point of failure as there are two in a cluster.	Requires purchase of power controller hardware and cables; consumes serial ports
Network-attached power switches	A single network attached power controller is required per cluster	Affords strong data integrity guarantees.	Requires purchase of power controller hardware. The power controller itself can be come a single point of failure (although they are typically very reliable devices).
Hardware Watchdog Timer	Affords strong data integrity guarantees	Obviates the need to purchase external power controller hardware	Not all systems include supported watchdog hardware
Software Watchdog Timer	Offers acceptable data integrity provisions	Obviates the need to purchase external power controller hardware; works on any system	Under some failure scenarios, the software watchdog will not be operational, opening a small vulnerability window
No power controller	No power controller function is in use	Obviates the need to purchase external power controller hardware; works on any system	Vulnerable to data corruption under certain failure scenarios

2.1.4 Cluster Hardware Tables

Use the following tables to identify the hardware components required for your cluster configuration. In some cases, the tables list specific products that have been tested in a cluster, although a cluster is expected to work with other products.

The complete set of qualified cluster hardware components change over time. Consequently, the table below may be incomplete. For the most up-do-date itemization of supported hardware components, refer to the Red Hat documentation website at http://www.redhat.com/docs.

Table 2–3 Cluster System Hardware Table

Hardware	Quantity	Description	Required
Cluster system	Two	Red Hat Cluster Manager supports IA-32 hardware platforms. Each cluster system must provide enough PCI slots, network slots, and serial ports for the cluster hardware configuration. Because disk devices must have the same name on each cluster system, it is recommended that the systems have symmetric I/O subsystems. In addition, it is recommended that each system have a minimum of 450 MHz CPU speed and 256 MB of memory. See Section 2.2.1, *Installing the Basic System Hardware* for more information.	Yes

Table 2–4, *Power Switch Hardware Table* includes several different types of power switches. A single cluster requires only one type of power switch shown below.

Table 2–4 Power Switch Hardware Table

Hardware	Quantity	Description	Required
Serial power switches	Two	Power switches enable each cluster system to power-cycle the other cluster system. See Section 2.4.2, *Configuring Power Switches* for information about using power switches in a cluster. Note that clusters are configured with either serial or network attached power switches and not both. The following serial attached power switch has been fully tested: RPS-10 (model M/HD in the US, and model M/EC in Europe), which is available from http://www.wti.com/rps-10.htm. Refer to Section A.1.1, *Setting up RPS-10 Power Switches* Latent support is provided for the following serial attached power switch. This switch has not yet been fully tested: APC Serial On/Off Switch (partAP9211), http://www.apc.com	Strongly recommended for data integrity under all failure conditions
Null modem cable	Two	Null modem cables connect a serial port on a cluster system to a serial power switch. This serial connection enables each cluster system to power-cycle the other system. Some power switches may require different cables.	Only if using serial power switches
Mounting bracket	One	Some power switches support rack mount configurations and require a separate mounting bracket (e.g. RPS-10).	Only for rack mounting power switches

Hardware	Quantity	Description	Required
Network power switch	One	Network attached power switches enable each cluster member to power cycle all others. Refer to Section 2.4.2, *Configuring Power Switches* for information about using network attached power switches, as well as caveats associated with each. The following network attached power switch has been fully tested: · WTI NPS-115, or NPS-230, available from http://www.wti.com. Note that the NPS power switch can properly accommodate systems with dual redundant power supplies. Refer to Section A.1.2, *Setting up WTI NPS Power Switches*. · Baytech RPC-3 and RPC-5, http://www.baytech.net Latent support is provided for the APC Master Switch (AP9211, or AP9212), www.apc.com	Strongly recommended for data integrity under all failure conditions
Watchdog Timer	Two	Watchdog timers cause a failed cluster member to remove itself from a cluster prior to a healthy member taking over its services. Refer to Section 2.4.2, *Configuring Power Switches* for more information	Recommended for data integrity on systems which provide integrated watchdog hardware

The following table shows a variety of storage devices for an administrator to choose from. An individual cluster does *not* require all of the components listed below.

Table 2–5 Shared Disk Storage Hardware Table

Hardware	Quantity	Description	Required
External disk storage enclosure	One	Use Fibre Channel or single-initiator parallel SCSI to connect the cluster systems to a single or dual-controller RAID array. To use single-initiator buses, a RAID controller must have multiple host ports and provide simultaneous access to all the logical units on the host ports. To use a dual-controller RAID array, a logical unit must fail over from one controller to the other in a way that is transparent to the operating system. The following are recommended SCSI RAID arrays that provide simultaneous access to all the logical units on the host ports (this is not a comprehensive list; rather its limited to those RAID boxes which have been tested): · Winchester Systems FlashDisk RAID Disk Array, which is available from http://www.winsys.com. · Dot Hill's SANnet Storage Systems, which is available from http://www.dothill.com · Silicon Image CRD-7040 & CRA-7040, CRD -7220, CRD-7240 & CRA-7240, CRD-7400 & CRA-7400 controller based RAID arrays. Available from http://www.synetexinc.com In order to ensure symmetry of device IDs and LUNs, many RAID arrays with dual redundant controllers are required to be configured in an active/passive mode. See Section 2.4.4, *Configuring Shared Disk Storage* for more information.	Yes

Hardware	Quantity	Description	Required
Host bus adapter	Two	To connect to shared disk storage, you must install either a parallel SCSI or a Fibre Channel host bus adapter in a PCI slot in each cluster system. For parallel SCSI, use a low voltage differential (LVD) host bus adapter. Adapters have either HD68 or VHDCI connectors. Recommended parallel SCSI host bus adapters include the following: · Adaptec 2940U2W, 29160, 29160LP, 39160, and 3950U2 · Adaptec AIC-7896 on the Intel L440GX+ motherboard · Qlogic QLA1080 and QLA12160 · Tekram Ultra2 DC-390U2W · LSI Logic SYM22915 · A recommended Fibre Channel host bus adapter is the Qlogic QLA2200. See Section A.6, *Host Bus Adapter Features and Configuration Requirements* for device features and configuration information. Host-bus adapter based RAID cards are only supported if they correctly support multi-host operation. At the time of publication, there were no fully tested host-bus adapter based RAID cards. Refer to http://www.redhat.com for more the latest hardware information.	Yes
SCSI cable	Two	SCSI cables with 68 pins connect each host bus adapter to a storage enclosure port. Cables have either HD68 or VHDCI connectors. Cables vary based on adapter type	Only for parallel SCSI configurations

Hardware	Quantity	Description	Required
SCSI terminator	Two	For a RAID storage enclosure that uses "out" ports (such as FlashDisk RAID Disk Array) and is connected to single-initiator SCSI buses, connect terminators to the "out" ports in order to terminate the buses.	Only for parallel SCSI configurations and only if necessary for termination
Fibre Channel hub or switch	One or two	A Fibre Channel hub or switch is required.	Only for some Fibre Channel configurations
Fibre Channel cable	Two to six	A Fibre Channel cable connects a host bus adapter to a storage enclosure port, a Fibre Channel hub, or a Fibre Channel switch. If a hub or switch is used, additional cables are needed to connect the hub or switch to the storage adapter ports.	Only for Fibre Channel configurations

Table 2–6 Network Hardware Table

Hardware	Quantity	Description	Required
Network interface	One for each network connection	Each network connection requires a network interface installed in a cluster system.	Yes
Network switch or hub	One	A network switch or hub allows connection of multiple systems to a network.	No
Network cable	One for each network interface	A conventional network cable, such as a cable with an RJ45 connector, connects each network interface to a network switch or a network hub.	Yes

Table 2–7 Point-To-Point Ethernet Heartbeat Channel Hardware Table

Hardware	Quantity	Description	Required
Network interface	Two for each channel	Each Ethernet heartbeat channel requires a network interface installed in both cluster systems.	No
Network crossover cable	One for each channel	A network crossover cable connects a network interface on one cluster system to a network interface on the other cluster system, creating an Ethernet heartbeat channel.	Only for a redundant Ethernet heartbeat channel

Table 2–8 Point-To-Point Serial Heartbeat Channel Hardware Table

Hardware	Quantity	Description	Required
Serial card	Two for each serial channel	Each serial heartbeat channel requires a serial port on both cluster systems. To expand your serial port capacity, you can use multi-port serial PCI cards. Recommended multi-port cards include the following: Vision Systems VScom 200H PCI card, which provides two serial ports, is available from http://www.vscom.de Cyclades-4YoPCI+ card, which provides four serial ports, is available from http://www.cyclades.com. Note that since configuration of serial heartbeat channels is optional, it is not required to invest in additional hardware specifically for this purpose. Should future support be provided for more than 2 cluster members, serial heartbeat channel support may be deprecated.	No
Null modem cable	One for each channel	A null modem cable connects a serial port on one cluster system to a corresponding serial port on the other cluster system, creating a serial heartbeat channel.	Only for serial heartbeat channel

Table 2–9 Console Switch Hardware Table

Hardware	Quantity	Description	Required
Terminal server	One	A terminal server enables you to manage many systems from one remote location.	No
KVM	One	A KVM enables multiple systems to share one keyboard, monitor, and mouse. Cables for connecting systems to the switch depend on the type of KVM.	No

Table 2–10 UPS System Hardware Table

Hardware	Quantity	Description	Required
UPS system	One or two	**Uninterruptible power supply** (UPS) systems protect against downtime if a power outage occurs. UPS systems are highly recommended for cluster operation. Ideally, connect the power cables for the shared storage enclosure and both power switches to redundant UPS systems. In addition, a UPS system must be able to provide voltage for an adequate period of time, and should be connected to its own power circuit. A recommended UPS system is the APC Smart-UPS 1400 Rackmount available from http://www.apc.com.	Strongly recommended for availability

2.1.5 Example of a Minimum Cluster Configuration

The hardware components described in Table 2–11, *Minimum Cluster Hardware Configuration Components* can be used to set up a minimum cluster configuration. This configuration does not guarantee data integrity under all failure conditions, because it does not include power switches. Note that this is a sample configuration; it is possible to set up a minimum configuration using other hardware.

Table 2–11 Minimum Cluster Hardware Configuration Components

Hardware	Quantity
Two servers	Each cluster system includes the following hardware: Network interface for client access and an Ethernet heartbeat channel One Adaptec 29160 SCSI adapter (termination disabled) for the shared storage connection
Two network cables with RJ45 connectors	Network cables connect a network interface on each cluster system to the network for client access and Ethernet heartbeats.

Hardware	Quantity
RAID storage enclosure	The RAID storage enclosure contains one controller with at least two host ports.
Two HD68 SCSI cables	Each cable connects one HBA to one port on the RAID controller, creating two single-initiator SCSI buses.

2.1.6 Example of a No-Single-Point-Of-Failure Configuration

The components described in Table 2–12, *No-Single-Point-Of-Failure Configuration Components* can be used to set up a no-single-point-of-failure cluster configuration that includes two single-initiator SCSI buses and power switches to guarantee data integrity under all failure conditions. Note that this is a sample configuration; it is possible to set up a no-single-point-of-failure configuration using other hardware.

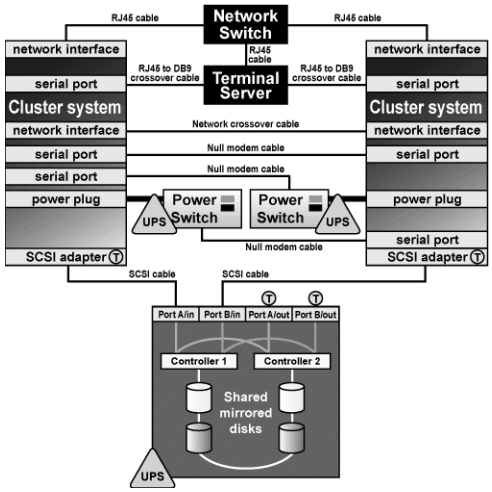
Table 2–12 No-Single-Point-Of-Failure Configuration Components

Hardware	Quantity
Two servers	Each cluster system includes the following hardware: Two network interfaces for: Point-to-point Ethernet heartbeat channel Client network access and Ethernet heartbeat connection Three serial ports for: Point-to-point serial heartbeat channel Remote power switch connection Connection to the terminal server One Tekram Ultra2 DC-390U2W adapter (termination enabled) for the shared disk storage connection
One network switch	A network switch enables the connection of multiple systems to a network.
One Cyclades terminal server	A terminal server allows for management of remote systems from a central location. (A terminal server is not required for cluster operation.)
Three network cables	Network cables connect the terminal server and a network interface on each cluster system to the network switch.
Two RJ45 to DB9 crossover cables	RJ45 to DB9 crossover cables connect a serial port on each cluster system to the Cyclades terminal server.

Hardware	Quantity
One network crossover cable	A network crossover cable connects a network interface on one cluster system to a network interface on the other system, creating a point-to-point Ethernet heartbeat channel.
Two RPS-10 power switches	Power switches enable each cluster system to power-cycle the other system before restarting its services. The power cable for each cluster system is connected to its own power switch.
Three null modem cables	Null modem cables connect a serial port on each cluster system to the power switch that provides power to the other cluster system. This connection enables each cluster system to power-cycle the other system. A null modem cable connects a serial port on one cluster system to a corresponding serial port on the other system, creating a point-to-point serial heartbeat channel.
FlashDisk RAID Disk Array with dual controllers	Dual RAID controllers protect against disk and controller failure. The RAID controllers provide simultaneous access to all the logical units on the host ports.
Two HD68 SCSI cables	HD68 cables connect each host bus adapter to a RAID enclosure "in" port, creating two single-initiator SCSI buses.
Two terminators	Terminators connected to each "out" port on the RAID enclosure terminate both single-initiator SCSI buses.
Redundant UPS Systems	UPS systems provide a highly-available source of power. The power cables for the power switches and the RAID enclosure are connected to two UPS systems.

Figure 2–1, *No-Single-Point-Of-Failure Configuration Example* shows an example of a no-single-point-of-failure hardware configuration that includes the hardware described in the previous table, two single-initiator SCSI buses, and power switches to guarantee data integrity under all error conditions. A "T" enclosed in a circle represents a SCSI terminator.

Figure 2–1 No-Single-Point-Of-Failure Configuration Example



2.2 Steps for Setting Up the Cluster Systems

After identifying the cluster hardware components described in Section 2.1, *Choosing a Hardware Configuration*, set up the basic cluster system hardware and connect the systems to the optional console switch and network switch or hub. Follow these steps:

1. In both cluster systems, install the required network adapters, serial cards, and host bus adapters. See Section 2.2.1, *Installing the Basic System Hardware* for more information about performing this task.
2. Set up the optional console switch and connect it to each cluster system. See Section 2.2.2, *Setting Up a Console Switch* for more information about performing this task.

   If a console switch is not used, then connect each system to a console terminal.
3. Set up the optional network switch or hub and use conventional network cables to connect it to the cluster systems and the terminal server (if applicable). See Section 2.2.3, *Setting Up a Network Switch or Hub* for more information about performing this task.

   If a network switch or hub is not used, then conventional network cables should be used to connect each system and the terminal server (if applicable) to a network.

After performing the previous tasks, install the Linux distribution as described in Section 2.3, *Steps for Installing and Configuring the Red Hat Linux Distribution*.

2.2.1 Installing the Basic System Hardware

Cluster systems must provide the CPU processing power and memory required by applications. It is recommended that each system have a minimum of 450 MHz CPU speed and 256 MB of memory.

In addition, cluster systems must be able to accommodate the SCSI or FC adapters, network interfaces, and serial ports that the hardware configuration requires. Systems have a limited number of preinstalled serial and network ports and PCI expansion slots. The following table will help to determine how much capacity the cluster systems employed will require:

Table 2–13 Installing the Basic System Hardware

Cluster Hardware Component	Serial Ports	Network Slots	PCI Slots
Remote power switch connection (optional, but strongly recommended)	One		
SCSI or Fibre Channel adapter to shared disk storage			One for each bus adapter
Network connection for client access and Ethernet heartbeat		One for each network connection	
Point-to-point Ethernet heartbeat channel (optional)		One for each channel	
Point-to-point serial heartbeat channel (optional)	One for each channel		
Terminal server connection (optional)	One		

Most systems come with at least one serial port. Ideally, choose systems that have at least two serial ports. If a system has graphics display capability, it is possible to use the serial console port for a serial heartbeat channel or a power switch connection. To expand your serial port capacity, use multi-port serial PCI cards.

In addition, be sure that local system disks will not be on the same SCSI bus as the shared disks. For example, use two-channel SCSI adapters, such as the Adaptec 39160-series cards, and put the internal

devices on one channel and the shared disks on the other channel. Using multiple SCSI cards is also possible.

See the system documentation supplied by the vendor for detailed installation information. See Appendix A, *Supplementary Hardware Information* for hardware-specific information about using host bus adapters in a cluster.

Figure 2–2 Typical Cluster System External Cabling

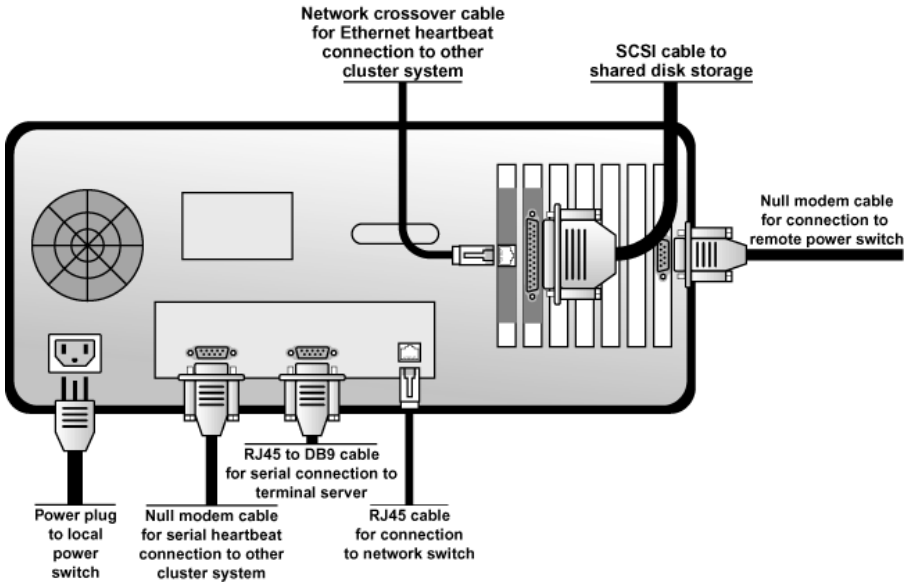


Figure 2–2, *Typical Cluster System External Cabling* shows the bulkhead of a sample cluster system and the external cable connections for a typical cluster configuration.

2.2.2 Setting Up a Console Switch

Although a console switch is not required for cluster operation, it can be used to facilitate cluster system management and eliminate the need for separate monitors, mouses, and keyboards for each cluster system. There are several types of console switches.

For example, a terminal server enables connection to serial consoles and management of many systems from a remote location. For a low-cost alternative, use a KVM (keyboard, video, and mouse) switch, which enables multiple systems to share one keyboard, monitor, and mouse. A KVM switch is suitable for configurations in which access to a graphical user interface (GUI) to perform system management tasks is preferred.

Set up the console switch according to the documentation provided by the vendor.

After the console switch has been set up, connect it to each cluster system. The cables used depend on the type of console switch. For example, if you a Cyclades terminal server uses RJ45 to DB9 crossover cables to connect a serial port on each cluster system to the terminal server.

2.2.3 Setting Up a Network Switch or Hub

Although a network switch or hub is not required for cluster operation, it can be used to facilitate cluster and client system network operations.

Set up a network switch or hub according to the documentation provided by the vendor.

After the network switch or hub has been set up, connect it to each cluster system by using conventional network cables. When using a terminal server, a network cable connects it to the network switch or hub.

2.3 Steps for Installing and Configuring the Red Hat Linux Distribution

After the setup of basic system hardware, proceed with installation of Red Hat Linux on both cluster systems and ensure that they recognize the connected devices. Follow these steps:

1. Install the Red Hat Linux distribution on both cluster systems. If customizing the kernel, be sure to follow the kernel requirements and guidelines described in Section 2.3.1, *Kernel Requirements*.
2. Reboot the cluster systems.
3. When using a terminal server, configure Linux to send console messages to the console port.
4. Edit the `/etc/hosts` file on each cluster system and include the IP addresses used in the cluster. See Section 2.3.2, *Editing the `/etc/hosts` File* for more information about performing this task.
5. Decrease the alternate kernel boot timeout limit to reduce cluster system boot time. See Section 2.3.3, *Decreasing the Kernel Boot Timeout Limit* for more information about performing this task.
6. Ensure that no login (or `getty`) programs are associated with the serial ports that are being used for the serial heartbeat channel or the remote power switch connection (if applicable). To perform this task, edit the `/etc/inittab` file and use a pound symbol (#) to comment out the entries that correspond to the serial ports used for the serial channel and the remote power switch. Then, invoke the `init q` command.
7. Verify that both systems detect all the installed hardware:
   - Use the `dmesg` command to display the console startup messages. See Section 2.3.4, *Displaying Console Startup Messages* for more information about performing this task.

- Use the `cat /proc/devices` command to display the devices configured in the kernel. See Section 2.3.5, *Displaying Devices Configured in the Kernel* for more information about performing this task.

8. Verify that the cluster systems can communicate over all the network interfaces by using the `ping` command to send test packets from one system to the other.
9. If intending to configure Samba services, verify that the Samba related RPM packages are installed on your system.

2.3.1 Kernel Requirements

When manually configuring the kernel, adhere to the following are kernel requirements:

- Enable IP Aliasing support in the kernel by setting the `CONFIG_IP_ALIAS` kernel option to `y`. When specifying kernel options, under `Networking Options`, select `IP aliasing support`.
- Enable support for the `/proc` file system by setting the `CONFIG_PROC_FS` kernel option to `y`. When specifying kernel options, under `Filesystems`, select `/proc filesystem support`.
- Ensure that the SCSI driver is started before the cluster software. For example, edit the startup scripts so that the driver is started before the cluster script. It is also possible to statically build the SCSI driver into the kernel, instead of including it as a loadable module, by modifying the `/etc/modules.conf` file.

In addition, when installing the Linux distribution, it is *strongly recommended* to do the following:

- Gather the IP addresses for the cluster systems and for the point-to-point Ethernet heartbeat interfaces, before installing a Linux distribution. Note that the IP addresses for the point-to-point Ethernet interfaces can be private IP addresses, (for example, 10.*x.x.x*).
- Optionally, reserve an IP address to be used as the "cluster alias". This address is typically used to facilitate remote monitoring.
- Enable the following Linux kernel options to provide detailed information about the system configuration and events and help you diagnose problems:
  - Enable SCSI logging support by setting the `CONFIG_SCSI_LOGGING` kernel option to `y`. When specifying kernel options, under `SCSI Support`, select `SCSI logging facility`.
  - Enable support for `sysctl` by setting the `CONFIG_SYSCTL` kernel option to `y`. When specifying kernel options, under `General Setup`, select `Sysctl support`.

- Do not place local file systems, such as /, /etc, /tmp, and /var on shared disks or on the same SCSI bus as shared disks. This helps prevent the other cluster member from accidentally mounting these file systems, and also reserves the limited number of SCSI identification numbers on a bus for cluster disks.
- Place /tmp and /var on different file systems. This may improve system performance.
- When a cluster system boots, be sure that the system detects the disk devices in the same order in which they were detected during the Linux installation. If the devices are not detected in the same order, the system may not boot.
- When using RAID storage configured with Logical Unit Numbers (LUNs) greater than zero, it is necessary to enable LUN support by adding the following to /etc/modules.conf:

  ```
  options scsi_mod max_scsi_luns=255
  ```

  After modifying modules.conf, it is necessary to rebuild the initial ram disk using mkinitrd. Refer to the Official Red Hat Linux Customization Guide for more information about creating ramdisks using mkinitrd.

2.3.2 Editing the /etc/hosts File

The /etc/hosts file contains the IP address-to-hostname translation table. The /etc/hosts file on each cluster system must contain entries for the following:

- IP addresses and associated host names for both cluster systems
- IP addresses and associated host names for the point-to-point Ethernet heartbeat connections (these can be private IP addresses)

As an alternative to the /etc/hosts file, naming services such as DNS or NIS can be used to define the host names used by a cluster. However, to limit the number of dependencies and optimize availability, it is strongly recommended to use the /etc/hosts file to define IP addresses for cluster network interfaces.

The following is an example of an /etc/hosts file on a cluster system:

```
127.0.0.1          localhost.localdomain   localhost
193.186.1.81       cluster2.yourdomain.com      cluster2
10.0.0.1           ecluster2.yourdomain.com     ecluster2
193.186.1.82       cluster3.yourdomain.com      cluster3
10.0.0.2           ecluster3.yourdomain.com     ecluster3
193.186.1.83       clusteralias.yourdomain.com  clusteralias
```

The previous example shows the IP addresses and host names for two cluster systems (**cluster2** and **cluster3**), and the private IP addresses and host names for the Ethernet interface used for the point-to-

point heartbeat connection on each cluster system (**ecluster2** and **ecluster3**) as well as the IP alias **clusteralias** used for remote cluster monitoring.

Verify correct formatting of the local host entry in the `/etc/hosts` file to ensure that it does not include non-local systems in the entry for the local host. An example of an incorrect local host entry that includes a non-local system (**server1**) is shown next:

```
127.0.0.1     localhost.localdomain     localhost server1
```

A heartbeat channel may not operate properly if the format is not correct. For example, the channel will erroneously appear to be offline. Check the `/etc/hosts` file and correct the file format by removing non-local systems from the local host entry, if necessary.

Note that each network adapter must be configured with the appropriate IP address and netmask.

The following is an example of a portion of the output from the `/sbin/ifconfig` command on a cluster system:

```
# ifconfig

eth0      Link encap:Ethernet  HWaddr 00:00:BC:11:76:93
          inet addr:192.186.1.81  Bcast:192.186.1.245  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:65508254 errors:225 dropped:0 overruns:2 frame:0
          TX packets:40364135 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:19 Base address:0xfce0

eth1      Link encap:Ethernet  HWaddr 00:00:BC:11:76:92
          inet addr:10.0.0.1  Bcast:10.0.0.245  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:18 Base address:0xfcc0
```

The previous example shows two network interfaces on a cluster system: The **eth0** network interface for the cluster system and the **eth1** (network interface for the point-to-point heartbeat connection).

2.3.3 Decreasing the Kernel Boot Timeout Limit

It is possible to reduce the boot time for a cluster system by decreasing the kernel boot timeout limit. During the Linux boot sequence, the bootloader allows for specifying an alternate kernel to boot. The default timeout limit for specifying a kernel is ten seconds.

To modify the kernel boot timeout limit for a cluster system, edit the `/etc/lilo.conf` file and specify the desired value (in tenths of a second) for the *timeout* parameter. The following example sets the timeout limit to three seconds:

```
timeout = 30
```

To apply any changes made to the `/etc/lilo.conf` file, invoke the `/sbin/lilo` command.

Similarly, when using the `grub` boot loader, the timeout parameter in `/boot/grub/grub.conf` should be modified to specify the appropriate number of seconds before timing out. To set this interval to 3 seconds, edit the parameter to the following:

```
timeout = 3
```

2.3.4 Displaying Console Startup Messages

Use the `dmesg` command to display the console startup messages. See the `dmesg`(8) manual page for more information.

The following example of the `dmesg` command output shows that a serial expansion card was recognized during startup:

```
May 22 14:02:10 storage3 kernel: Cyclades driver 2.3.2.5 2000/01/19 14:35:33
May 22 14:02:10 storage3 kernel: built May 8 2000 12:40:12
May 22 14:02:10 storage3 kernel: Cyclom-Y/PCI #1: 0xd0002000-0xd0005fff, IRQ9,
   4 channels starting from port 0.
```

The following example of the `dmesg` command output shows that two external SCSI buses and nine disks were detected on the system (note that lines with forward slashes will be printed as one line on most screens):

```
May 22 14:02:10 storage3 kernel: scsi0 : Adaptec AHA274x/284x/294x \
    (EISA/VLB/PCI-Fast SCSI) 5.1.28/3.2.4
May 22 14:02:10 storage3 kernel:
May 22 14:02:10 storage3 kernel: scsi1 : Adaptec AHA274x/284x/294x \
        (EISA/VLB/PCI-Fast SCSI) 5.1.28/3.2.4
May 22 14:02:10 storage3 kernel:
May 22 14:02:10 storage3 kernel: scsi : 2 hosts.
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST39236LW       Rev: 0004
May 22 14:02:11 storage3 kernel: Detected scsi disk sda at scsi0, channel 0, id 0, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST318203LC      Rev: 0001
May 22 14:02:11 storage3 kernel: Detected scsi disk sdb at scsi1, channel 0, id 0, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST318203LC      Rev: 0001
May 22 14:02:11 storage3 kernel: Detected scsi disk sdc at scsi1, channel 0, id 1, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST318203LC      Rev: 0001
May 22 14:02:11 storage3 kernel: Detected scsi disk sdd at scsi1, channel 0, id 2, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST318203LC      Rev: 0001
```

```
May 22 14:02:11 storage3 kernel: Detected scsi disk sde at scsi1, channel 0, id 3, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST318203LC       Rev: 0001
May 22 14:02:11 storage3 kernel: Detected scsi disk sdf at scsi1, channel 0, id 8, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST318203LC       Rev: 0001
May 22 14:02:11 storage3 kernel: Detected scsi disk sdg at scsi1, channel 0, id 9, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST318203LC       Rev: 0001
May 22 14:02:11 storage3 kernel: Detected scsi disk sdh at scsi1, channel 0, id 10, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST318203LC       Rev: 0001
May 22 14:02:11 storage3 kernel: Detected scsi disk sdi at scsi1, channel 0, id 11, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: Dell      Model: 8 BAY U2W CU      Rev: 0205
May 22 14:02:11 storage3 kernel:   Type:   Processor \
                    ANSI SCSI revision: 03
May 22 14:02:11 storage3 kernel: scsi1 : channel 0 target 15 lun 1 request sense \
    failed, performing reset.
May 22 14:02:11 storage3 kernel: SCSI bus is being reset for host 1 channel 0.
May 22 14:02:11 storage3 kernel: scsi : detected 9 SCSI disks total.
```

The following example of the `dmesg` command output shows that a quad Ethernet card was detected on the system:

```
May 22 14:02:11 storage3 kernel: 3c59x.c:v0.99H 11/17/98 Donald Becker
May 22 14:02:11 storage3 kernel: tulip.c:v0.91g-ppc 7/16/99 becker@cesdis.gsfc.nasa.gov
May 22 14:02:11 storage3 kernel: eth0: Digital DS21140 Tulip rev 34 at 0x9800, \
    00:00:BC:11:76:93, IRQ 5.
May 22 14:02:12 storage3 kernel: eth1: Digital DS21140 Tulip rev 34 at 0x9400, \
    00:00:BC:11:76:92, IRQ 9.
May 22 14:02:12 storage3 kernel: eth2: Digital DS21140 Tulip rev 34 at 0x9000, \
    00:00:BC:11:76:91, IRQ 11.
May 22 14:02:12 storage3 kernel: eth3: Digital DS21140 Tulip rev 34 at 0x8800, \
    00:00:BC:11:76:90, IRQ 10.
```

2.3.5 Displaying Devices Configured in the Kernel

To be sure that the installed devices, including serial and network interfaces, are configured in the kernel, use the `cat /proc/devices` command on each cluster system. Use this command to also determine if there is raw device support installed on the system. For example:

```
# cat /proc/devices
Character devices:
  1 mem
  2 pty
  3 ttyp
  4 ttyS
  5 cua
  7 vcs
 10 misc
```

```
 19 ttyC
 20 cub
128 ptm
136 pts
162 raw

Block devices:
  2 fd
  3 ide0
  8 sd
 65 sd
#
```

The previous example shows:

- Onboard serial ports (ttyS)
- Serial expansion card (ttyC)
- Raw devices (raw)
- SCSI devices (sd)

2.4 Steps for Setting Up and Connecting the Cluster Hardware

After installing Red Hat Linux, set up the cluster hardware components and verify the installation to ensure that the cluster systems recognize all the connected devices. Note that the exact steps for setting up the hardware depend on the type of configuration. See Section 2.1, *Choosing a Hardware Configuration* for more information about cluster configurations.

To set up the cluster hardware, follow these steps:

1. Shut down the cluster systems and disconnect them from their power source.
2. Set up the point-to-point Ethernet and serial heartbeat channels, if applicable. See Section 2.4.1, *Configuring Heartbeat Channels* for more information about performing this task.
3. When using power switches, set up the devices and connect each cluster system to a power switch. See Section 2.4.2, *Configuring Power Switches* for more information about performing this task.

   In addition, it is recommended to connect each power switch (or each cluster system's power cord if not using power switches) to a different UPS system. See Section 2.4.3, *Configuring UPS Systems* for information about using optional UPS systems.

4. Set up the shared disk storage according to the vendor instructions and connect the cluster systems to the external storage enclosure.See Section 2.4.4, *Configuring Shared Disk Storage* for more information about performing this task.

   In addition, it is recommended to connect the storage enclosure to redundant UPS systems. See Section 2.4.3, *Configuring UPS Systems* for more information about using optional UPS systems.

5. Turn on power to the hardware, and boot each cluster system. During the boot-up process, enter the BIOS utility to modify the system setup, as follows:

   - Ensure that the SCSI identification number used by the HBA is unique for the SCSI bus it is attached to. See Section A.5, *SCSI Identification Numbers* for more information about performing this task.
   - Enable or disable the onboard termination for each host bus adapter, as required by the storage configuration. See Section 2.4.4, *Configuring Shared Disk Storage* and Section A.3, *SCSI Bus Termination* for more information about performing this task.
   - Enable the cluster system to automatically boot when it is powered on.

6. Exit from the BIOS utility, and continue to boot each system. Examine the startup messages to verify that the Linux kernel has been configured and can recognize the full set of shared disks. Use the `dmesg` command to display console startup messages. See Section 2.3.4, *Displaying Console Startup Messages* for more information about using this command.

7. Verify that the cluster systems can communicate over each point-to-point Ethernet heartbeat connection by using the `ping` command to send packets over each network interface.

8. Set up the quorum disk partitions on the shared disk storage. See *Configuring Quorum Partitions* in Section 2.4.4 for more information about performing this task.

2.4.1 Configuring Heartbeat Channels

The cluster uses heartbeat channels as a policy input during failover of the cluster systems. For example, if a cluster system stops updating its timestamp on the quorum partitions, the other cluster system will check the status of the heartbeat channels to determine if additional time should be alloted prior to initiating a failover.

A cluster must include at least one heartbeat channel. It is possible to use an Ethernet connection for both client access and a heartbeat channel. However, it is recommended to set up additional heartbeat channels for high availability, using redundant Ethernet heartbeat channels, in addition to one or more serial heartbeat channels.

For example, if using both an Ethernet and a serial heartbeat channel, and the cable for the Ethernet channel is disconnected, the cluster systems can still check status through the serial heartbeat channel.

To set up a redundant Ethernet heartbeat channel, use a network crossover cable to connect a network interface on one cluster system to a network interface on the other cluster system.

To set up a serial heartbeat channel, use a null modem cable to connect a serial port on one cluster system to a serial port on the other cluster system. Be sure to connect corresponding serial ports on the cluster systems; do not connect to the serial port that will be used for a remote power switch connection. In the future, should support be added for more than two cluster members, then usage of serial based heartbeat channels may be deprecated.

2.4.2 Configuring Power Switches

Power switches enable a cluster system to power-cycle the other cluster system before restarting its services as part of the failover process. The ability to remotely disable a system ensures data integrity is maintained under any failure condition. It is recommended that production environments use power switches or watchdog timers in the cluster configuration. Only development (test) environments should use a configuration without power switches (type "None"). Refer to Section 2.1.3, *Choosing the Type of Power Controller* for a description of the various types of power switches. Note that within this section, the general term "power switch" also includes watchdog timers.

In a cluster configuration that uses physical power switches, each cluster system's power cable is connected to a power switch through either a serial or network connection (depending on switch type). When failover occurs, a cluster system can use this connection to power-cycle the other cluster system before restarting its services.

Power switches protect against data corruption if an unresponsive (or hanging) system becomes responsive after its services have failed over, and issues I/O to a disk that is also receiving I/O from the other cluster system. In addition, if a quorum daemon fails on a cluster system, the system is no longer able to monitor the quorum partitions. If power switches or watchdog timers are not used in the cluster, then this error condition may result in services being run on more than one cluster system, which can cause data corruption and possibly system crashes.

It is strongly recommended to use power switches in a cluster. However, administrators who are aware of the risks may choose to set up a cluster without power switches.

A cluster system may hang for a few seconds if it is swapping or has a high system workload. For this reason, adequate time is allowed prior to concluding another system has failed (typically 12 seconds).

A cluster system may "hang" indefinitely because of a hardware failure or kernel error. In this case, the other cluster will notice that the hung system is not updating its timestamp on the quorum partitions, and is not responding to pings over the heartbeat channels.

If a cluster system determines that a hung system is down, and power switches are used in the cluster, the cluster system will power-cycle the hung system before restarting its services. Clusters configured to use watchdog timers will self-reboot under most system hangs. This will cause the hung system to reboot in a clean state, and prevent it from issuing I/O and corrupting service data.

If power switches are not used in cluster, and a cluster system determines that a hung system is down, it will set the status of the failed system to DOWN on the quorum partitions, and then restart the hung system's services. If the hung system becomes becomes responsive, it will notice that its status is DOWN, and initiate a system reboot. This will minimize the time that both cluster systems may be able to issue I/O to the same disk, but it does not provide the data integrity guarantee of power switches. If the hung system never becomes responsive and no power switches are in use, then a manual reboot is required.

When used, power switches must be set up according to the vendor instructions. However, some cluster-specific tasks may be required to use a power switch in the cluster. See Section A.1, *Setting Up Power Switches* for detailed information on power switches (including information about watchdog timers). Be sure to take note of any caveats or functional attributes of specific power switch types. Note that the cluster-specific information provided in this document supersedes the vendor information.

When cabling power switches, take special care to ensure that each cable is plugged into the appropriate outlet. This is crucial because there is no independent means for the software to verify correct cabling. Failure to cable correctly can lead to an incorrect system being power cycled, or for one system to inappropriately conclude that it has successfully power cycled another cluster member.

After setting up the power switches, perform these tasks to connect them to the cluster systems:

1. Connect the power cable for each cluster system to a power switch.
2. On each cluster system, connect a serial port to the serial port on the power switch that provides power to the other cluster system. The cable used for the serial connection depends on the type of power switch. For example, an RPS-10 power switch uses null modem cables, while a network attached power switch requires a network cable.
3. Connect the power cable for each power switch to a power source. It is recommended to connect each power switch to a different UPS system. See Section 2.4.3, *Configuring UPS Systems* for more information.

After the installation of the cluster software, test the power switches to ensure that each cluster system can power-cycle the other system before starting the cluster. See Section 3.2.2, *Testing the Power Switches* for information.

2.4.3 Configuring UPS Systems

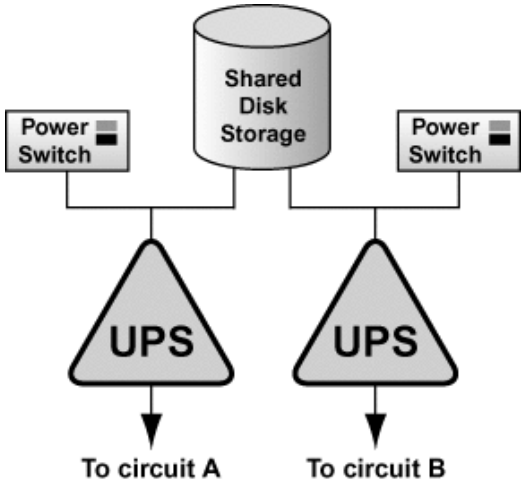
Uninterruptible power supply (UPS) systems provide a highly-available source of power. Ideally, a redundant solution should be used that incorporates multiple UPS's (one per server). For maximal fault-tolerance, it is possible to incorporate two UPS's per server as well as APC's Automatic Transfer Switches to manage the power and shutdown management of the server. Both solutions are solely dependent on the level of availability desired.

It is not recommended to use a large UPS infrastructure as the sole source of power for the cluster. A UPS solution dedicated to the cluster itself allows for more flexibility in terms of manageability and availability.

A complete UPS system must be able to provide adequate voltage and current for a prolonged period of time. While there is no single UPS to fit every power requirement, a solution can be tailored to fit a particular configuration. Visit APC's UPS configurator at http://www.apcc.com/template/size/apc to find the correct UPS configuration for your server. The APC Smart-UPS product line ships with software management for Red Hat Linux. The name of the RPM package is `pbeagent`.

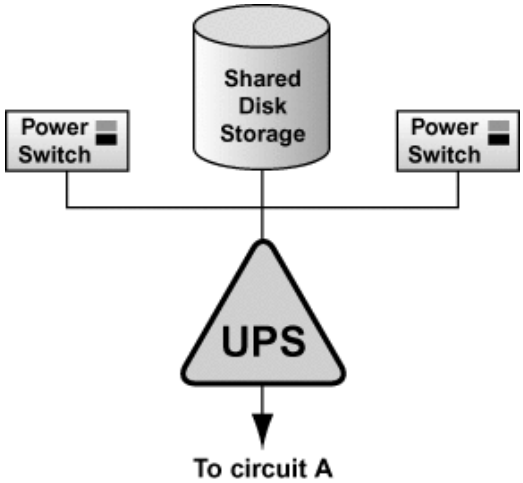
If the cluster disk storage subsystem has two power supplies with separate power cords, set up two UPS systems, and connect one power switch (or one cluster system's power cord if not using power switches) and one of the storage subsystem's power cords to each UPS system. A redundant UPS system configuration is shown in Figure 2–3, *Redundant UPS System Configuration*.

Figure 2–3 Redundant UPS System Configuration



An alternative redundant power configuration is to connect both power switches (or both cluster systems' power cords) and the disk storage subsystem to the same UPS system. This is the most cost-effective configuration, and provides some protection against power failure. However, if a power outage occurs, the single UPS system becomes a possible single point of failure. In addition, one UPS system may not be able to provide enough power to all the attached devices for an adequate amount of time. A single UPS system configuration is shown in Figure 2–4, *Single UPS System Configuration*.

Figure 2–4 Single UPS System Configuration



Many vendor-supplied UPS systems include Linux applications that monitor the operational status of the UPS system through a serial port connection. If the battery power is low, the monitoring software will initiate a clean system shutdown. As this occurs, the cluster software will be properly stopped, because it is controlled by a System V run level script (for example, `/etc/rc.d/init.d/cluster`).

See the UPS documentation supplied by the vendor for detailed installation information.

2.4.4 Configuring Shared Disk Storage

In a cluster, shared disk storage is used to hold service data and two quorum partitions. Because this storage must be available to both cluster systems, it cannot be located on disks that depend on the availability of any one system. See the vendor documentation for detailed product and installation information.

There are some factors to consider when setting up shared disk storage in a cluster:

- External RAID

  It is strongly recommended to use use RAID 1 (mirroring) to make service data and the quorum partitions highly available. Optionally, parity RAID can also be employed for high-availability. Do not use RAID 0 (striping) alone for quorum partitions because this reduces storage availability.

- Multi-Initiator SCSI configurations

Multi-initiator SCSI configurations are not supported due to the difficulty in obtaining proper bus termination.

- The Linux device name for each shared storage device must be the same on each cluster system. For example, a device named `/dev/sdc` on one cluster system must be named `/dev/sdc` on the other cluster system. Using identical hardware for both cluster systems usually ensures that these devices will be named the same.
- A disk partition can be used by only one cluster service.
- Do not include any file systems used in a cluster service in the cluster system's local `/etc/fstab` files, because the cluster software must control the mounting and unmounting of service file systems.
- For optimal performance, use a 4 KB block size when creating shared file systems. Note that some of the `mkfs` file system build utilities have a default 1 KB block size, which can cause long `fsck` times.

The following list details the **parallel SCSI requirements**, and must be adhered to if employed in a cluster environment:

- SCSI buses must be terminated at each end, and must adhere to length and hot plugging restrictions.
- Devices (disks, host bus adapters, and RAID controllers) on a SCSI bus must have a unique SCSI identification number.

See Section A.2, *SCSI Bus Configuration Requirements* for more information.

In addition, it is *strongly recommended* to connect the storage enclosure to redundant UPS systems for a highly-available source of power. See Section 2.4.3, *Configuring UPS Systems* for more information.

See *Setting Up a Single-Initiator SCSI Bus* in Section 2.4.4 and *Setting Up a Fibre Channel Interconnect* in Section 2.4.4 for more information about configuring shared storage.

After setting up the shared disk storage hardware, partition the disks and then either create file systems or raw devices on the partitions. Two raw devices must be created for the primary and the backup quorum partitions. See *Configuring Quorum Partitions* in Section 2.4.4, *Partitioning Disks* in Section 2.4.4, *Creating Raw Devices* in Section 2.4.4, and *Creating File Systems* in Section 2.4.4 for more information.

Setting Up a Single-Initiator SCSI Bus

A single-initiator SCSI bus has only one cluster system connected to it, and provides host isolation and better performance than a multi-initiator bus. Single-initiator buses ensure that each cluster system is protected from disruptions due to the workload, initialization, or repair of the other cluster system.

When using a single or dual-controller RAID array that has multiple host ports and provides simultaneous access to all the shared logical units from the host ports on the storage enclosure, the setup of

two single-initiator SCSI buses to connect each cluster system to the RAID array is possible. If a logical unit can fail over from one controller to the other, the process must be transparent to the operating system. Note that some RAID controllers restrict a set of disks to a specific controller or port. In this case, single-initiator bus setups are not possible.

A single-initiator bus must adhere to the requirements described in Section A.2, *SCSI Bus Configuration Requirements*. In addition, see Section A.6, *Host Bus Adapter Features and Configuration Requirements* for detailed information about terminating host bus adapters and configuring a single-initiator bus.

To set up a single-initiator SCSI bus configuration, the following is required:

- Enable the on-board termination for each host bus adapter.
- Enable the termination for each RAID controller.
- Use the appropriate SCSI cable to connect each host bus adapter to the storage enclosure.

Setting host bus adapter termination is usually done in the adapter BIOS utility during system boot. To set RAID controller termination, refer to the vendor documentation. shows a configuration that uses two single-initiator SCSI buses.

Figure 2–5 Single-Initiator SCSI Bus Configuration

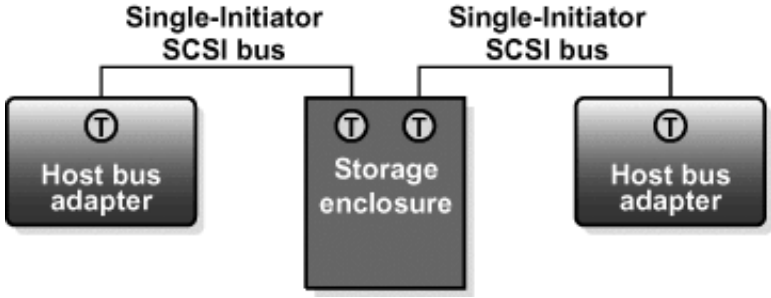


Figure 2–6, *Single-Controller RAID Array Connected to Single-Initiator SCSI Buses* shows the termination in a single-controller RAID array connected to two single-initiator SCSI buses.

Figure 2–6 Single-Controller RAID Array Connected to Single-Initiator SCSI Buses

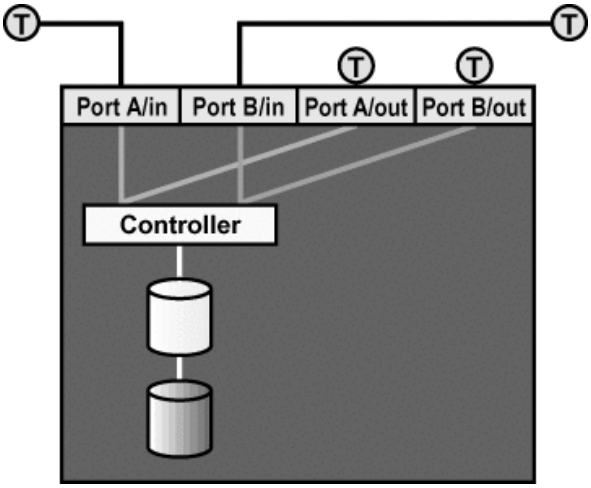
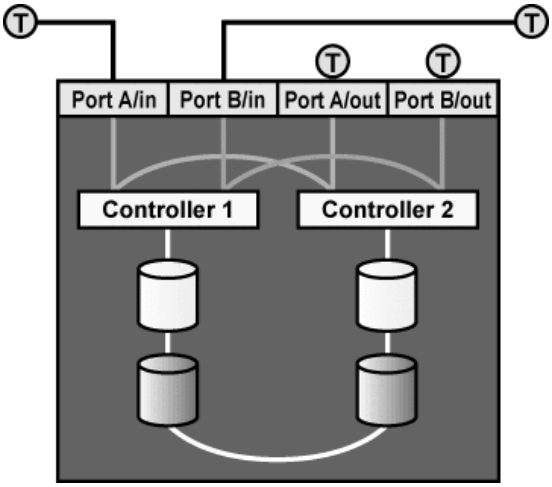


Figure 2–7 Dual-Controller RAID Array Connected to Single-Initiator SCSI Buses



Setting Up a Fibre Channel Interconnect

Fibre Channel can be used in either single-initiator or multi-initiator configurations

A single-initiator Fibre Channel interconnect has only one cluster system connected to it. This may provide better host isolation and better performance than a multi-initiator bus. Single-initiator interconnects ensure that each cluster system is protected from disruptions due to the workload, initialization, or repair of the other cluster system.

If employing a RAID array that has multiple host ports, and the RAID array provides simultaneous access to all the shared logical units from the host ports on the storage enclosure, set up two single-initiator Fibre Channel interconnects to connect each cluster system to the RAID array. If a logical unit can fail over from one controller to the other, the process must be transparent to the operating system.

Figure 2–8, *Single-Controller RAID Array Connected to Single-Initiator Fibre Channel Interconnects* shows a single-controller RAID array with two host ports, and the host bus adapters connected directly to the RAID controller, without using Fibre Channel hubs or switches.

Figure 2–8 Single-Controller RAID Array Connected to Single-Initiator Fibre Channel Interconnects

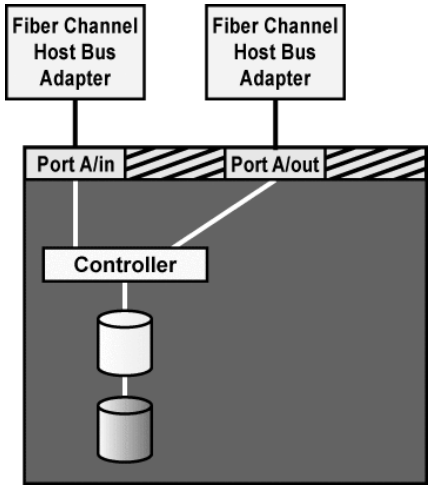
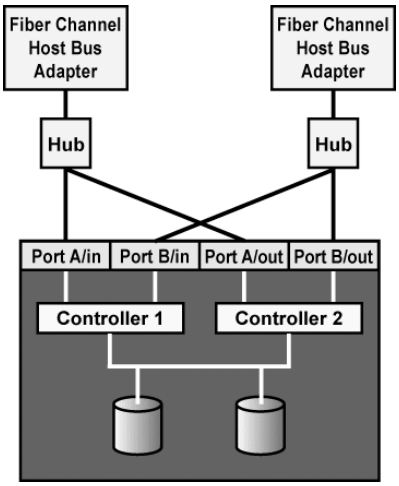


Figure 2–9 Dual-Controller RAID Array Connected to Single-Initiator Fibre Channel Interconnects



If a dual-controller RAID array with two host ports on each controller is used, a Fibre Channel hub or switch is required to connect each host bus adapter to one port on both controllers, as shown in Figure 2–9, *Dual-Controller RAID Array Connected to Single-Initiator Fibre Channel Interconnects*.

If a multi-initiator Fibre Channel is used, then a Fibre Channel hub or switch is required. In this case, each HBA is connected to the hub or switch, and the hub or switch is connected to a host port on each RAID controller.

Configuring Quorum Partitions

Two raw devices on shared disk storage must be created for the primary quorum partition and the backup quorum partition. Each quorum partition must have a minimum size of 10 MB. The amount of data in a quorum partition is constant; it does not increase or decrease over time.

The quorum partitions are used to hold cluster state information. Periodically, each cluster system writes its status (either UP or DOWN), a timestamp, and the state of its services. In addition, the quorum partitions contain a version of the cluster database. This ensures that each cluster system has a common view of the cluster configuration.

To monitor cluster health, the cluster systems periodically read state information from the primary quorum partition and determine if it is up to date. If the primary partition is corrupted, the cluster systems read the information from the backup quorum partition and simultaneously repair the primary

partition. Data consistency is maintained through checksums and any inconsistencies between the partitions are automatically corrected.

If a system is unable to write to both quorum partitions at startup time, it will not be allowed to join the cluster. In addition, if an active cluster system can no longer write to both quorum partitions, the system will remove itself from the cluster by rebooting (and may be remotely power cycled by the healthy cluster member).

The following are **quorum partition requirements**:

- Both quorum partitions must have a minimum size of 10 MB.
- Quorum partitions must be raw devices. They cannot contain file systems.
- Quorum partitions can be used only for cluster state and configuration information.

The following are *recommended guidelines* for configuring the quorum partitions:

- It is strongly recommended to set up a RAID subsystem for shared storage, and use RAID 1 (mirroring) to make the logical unit that contains the quorum partitions highly available. Optionally, parity RAID can be used for high-availability. Do not use RAID 0 (striping) alone for quorum partitions.
- Place both quorum partitions on the same RAID set, or on the same disk if RAID is not employed, because both quorum partitions must be available in order for the cluster to run.
- Do not put the quorum partitions on a disk that contains heavily-accessed service data. If possible, locate the quorum partitions on disks that contain service data that is rarely accessed.

See *Partitioning Disks* in Section 2.4.4 and *Creating Raw Devices* in Section 2.4.4 for more information about setting up the quorum partitions.

See Section 3.1.1, *Editing the* `rawdevices` *File* for information about editing the `rawdevices` file to bind the raw character devices to the block devices each time the cluster systems boot.

Partitioning Disks

After shared disk storage hardware has been set up, partition the disks so they can be used in the cluster. Then, create file systems or raw devices on the partitions. For example, two raw devices must be created for the quorum partitions using the guidelines described in *Configuring Quorum Partitions* in Section 2.4.4.

Invoke the interactive `fdisk` command to modify a disk partition table and divide the disk into partitions. While in `fdisk`, use the `p` command to display the current partition table and the `n` command to create new partitions.

The following example shows how to use the fdisk command to partition a disk:

1. Invoke the interactive `fdisk` command, specifying an available shared disk device. At the prompt, specify the `p` command to display the current partition table.

```
# fdisk /dev/sde
Command (m for help): p

Disk /dev/sde: 255 heads, 63 sectors, 2213 cylinders
Units = cylinders of 16065 * 512 bytes

Device    Boot    Start       End    Blocks   Id  System
/dev/sde1             1       262   2104483+  83  Linux
/dev/sde2           263       288    208845   83  Linux
```

2. Determine the number of the next available partition, and specify the `n` command to add the partition. If there are already three partitions on the disk, then specify `e` for extended partition or `p` to create a primary partition.

```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
```

3. Specify the partition number required:

```
Partition number (1-4): 3
```

4. Press the [Enter] key or specify the next available cylinder:

```
First cylinder (289-2213, default 289): 289
```

5. Specify the partition size that is required:

```
Last cylinder or +size or +sizeM or +sizeK (289-2213,
default 2213): +2000M
```

   Note that large partitions will increase the cluster service failover time if a file system on the partition must be checked with `fsck`. Quorum partitions must be at least 10 MB.

6. Specify the `w` command to write the new partition table to disk:

```
Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.

WARNING: If you have created or modified any DOS 6.x
partitions, please see the fdisk manual page for additional
information.
```

```
Syncing disks.
```

7. If a partition was added while both cluster systems are powered on and connected to the shared storage, reboot the other cluster system in order for it to recognize the new partition.

After partitioning a disk, format the partition for use in the cluster. For example, create file systems or raw devices for quorum partitions.

See *Creating Raw Devices* in Section 2.4.4 and *Creating File Systems* in Section 2.4.4 for more information.

For basic information on partitioning hard disks at installation time, see *The Official Red Hat Linux x86 Installation Guide*. *Appendix E. An Introduction to Disk Partitions* of *The Official Red Hat Linux x86 Installation Guide* also explains the basic concepts of partitioning.

For basic information on partitioning disks using `fdisk`, refer to the following URL http://kb.redhat.com/view.php?eid=175.

Creating Raw Devices

After partitioning the shared storage disks, create raw devices on the partitions. File systems are block devices (for example, `/dev/sda1`) that cache recently-used data in memory in order to improve performance. Raw devices do not utilize system memory for caching. See *Creating File Systems* in Section 2.4.4 for more information.

Linux supports raw character devices that are not hard-coded against specific block devices. Instead, Linux uses a character major number (currently 162) to implement a series of unbound raw devices in the `/dev/raw` directory. Any block device can have a character raw device front-end, even if the block device is loaded later at runtime.

To create a raw device, edit the `/etc/sysconfig/rawdevices` file to bind a raw character device to the appropriate block device. Once bound to a block device, a raw device can be opened, read, and written.

Quorum partitions and some database applications require raw devices, because these applications perform their own buffer caching for performance purposes. Quorum partitions cannot contain file systems because if state data was cached in system memory, the cluster systems would not have a consistent view of the state data.

Raw character devices must be bound to block devices each time a system boots. To ensure that this occurs, edit the `/etc/sysconfig/rawdevices` file and specify the quorum partition bindings. If using a raw device in a cluster service, use this file to bind the devices at boot time. See Section 3.1.1, *Editing the `rawdevices` File* for more information.

After editing `/etc/sysconfig/rawdevices`, the changes will take effect either by rebooting or by execute the following command:

```
# service rawdevices restart
```

Query all the raw devices by using the command `raw -aq`:

```
# raw -aq
/dev/raw/raw1   bound to major 8, minor 17
/dev/raw/raw2   bound to major 8, minor 18
```

Note that, for raw devices, there is no cache coherency between the raw device and the block device. In addition, requests must be 512-byte aligned both in memory and on disk. For example, the standard `dd` command cannot be used with raw devices because the memory buffer that the command passes to the write system call is not aligned on a 512-byte boundary.

For more information on using the `raw` command, refer to the `raw`(8) manual page.

Creating File Systems

Use the `mkfs` command to create an ext2 file system on a partition. Specify the drive letter and the partition number. For example:

```
# mkfs -t ext2 -b 4096 /dev/sde3
```

For optimal performance of shared filesystems, a 4 KB block size was specified in the above example. Note that it is necessary in most cases to specify a 4 KB block size when creating a filesystem since many of the mkfs file system build utilities default to a 1 KB block size, which can cause long `fsck` times.

Similarly, to create an ext3 filesystem, the following command can be used:

```
# mkfs -t ext2 -j -b 4096 /dev/sde3
```

For more information on creating filesystems, refer to the `mkfs`(8) manual page.

3 Cluster Software Installation and Configuration

After installing and configuring the cluster hardware, the cluster system software can be installed. The following sections describe installing and initializing of cluster software, checking cluster configuration, configuring `syslog` event logging, and using the `cluadmin` utility.

3.1 Steps for Installing and Initializing the Cluster Software

Before installing Red Hat Cluster Manager, be sure to install all of the required software, as described in Section 2.3.1, *Kernel Requirements*.

In order to preserve the existing cluster configuration database when running updates to the cluster software, back up the cluster database and stop the cluster software before reinstallation. See Section 8.7, *Updating the Cluster Software* for more information.

To install Red Hat Cluster Manager, invoke the command `rpm --install clumanager-`*`x`*`.rpm`, where *x* is the version of Red Hat Cluster Manager currently available. This package is installed by default in Red Hat Linux Advanced Server so it is typically not necessary to manually install this individual package.

To initialize and start the cluster software, perform the following tasks:

1. Edit the `/etc/sysconfig/rawdevices` file on both cluster systems and specify the raw device special files and character devices for the primary and backup quorum partitions. See *Configuring Quorum Partitions* in Section 2.4.4 and Section 3.1.1, *Editing the* `rawdevices` *File* for more information.
2. Run the /sbin/cluconfig utility on one cluster system. If updating the cluster software, the utility will inquire before using the the existing cluster database. The utility will remove the cluster database if it is not used.

   The utility will prompt for the following cluster-specific information, which will be entered into the **member** fields in the cluster database. A copy of this is located in the `/etc/cluster.conf` file:

   - Raw device special files for the primary and backup quorum partitions, as specified in the `/etc/sysconfig/rawdevices` file (for example, `/dev/raw/raw1` and `/dev/raw/raw2`)
   - Cluster system host names that are returned by the `hostname` command

- Number of heartbeat connections (channels), both Ethernet and serial
- Device special file for each heartbeat serial line connection (for example, `/dev/ttyS1`)
- IP host name associated with each heartbeat Ethernet interface
- IP address for remote cluster monitoring, also referred to as the "cluster alias". Refer to Section 3.1.2, *Configuring the Cluster Alias* for further information.
- Device special files for the serial ports to which the power switches are connected, if any (for example, /dev/ttyS0), or IP address of a network attached power switch.
- Power switch type (for example, **RPS10** or **None** if not using power switches)
- The system will prompt whether or not to enable remote monitoring. Refer to Section 3.1.2, *Configuring the Cluster Alias* for more information.

See Section 3.1.4, *Example of the cluconfig Utility* for an example of running the utility.

3. After completing the cluster initialization on one cluster system, perform the following tasks on the other cluster system:
   - Run the `/sbin/cluconfig --init=`*`raw_file`* command, where *raw_file* specifies the primary quorum partition. The script will use the information specified for the first cluster system as defaults. For example:

     ```
     cluconfig --init=/dev/raw/raw1
     ```

4. Check the cluster configuration:
   - Invoke the `cludiskutil` utility with the `-t` option on both cluster systems to ensure that the quorum partitions map to the same physical device. See Section 3.2.1, *Testing the Quorum Partitions* for more information.
   - If using power switches, invoke the `clustonith` command on both cluster systems to test the remote connections to the power switches. See Section 3.2.2, *Testing the Power Switches* for more information.
5. Optionally, configure event logging so that cluster messages are logged to a separate file. See Section 3.3, *Configuring syslog Event Logging* for information.
6. Start the cluster by invoking the `cluster start` command located in the System V `init` directory on both cluster systems. For example:

   ```
   service cluster start
   ```

After initializing the cluster, proceed to add cluster services. See Section 3.4, *Using the `cluadmin` Utility* and Section 4.1, *Configuring a Service* for more information.

3.1.1 Editing the rawdevices File

The /etc/sysconfig/rawdevices file is used to map the raw devices for the quorum partitions each time a cluster system boots. As part of the cluster software installation procedure, edit the rawdevices file on each cluster system and specify the raw character devices and block devices for the primary and backup quorum partitions. This must be done prior to running the cluconfig utility.

If raw devices are employed in a cluster service, the rawdevices file is also used to bind the devices at boot time. Edit the file and specify the raw character devices and block devices that you want to bind each time the system boots. To make the changes to the rawdevices file take effect without requiring a reboot, perform the following command:

```
service rawdevices restart
```

The following is an example rawdevices file that designates two quorum partitions:

```
# raw device bindings
# format:  <rawdev> <major> <minor>
#          <rawdev> <blockdev>
# example: /dev/raw/raw1 /dev/sda1
#          /dev/raw/raw2 8 5
/dev/raw/raw1 /dev/sdb1
/dev/raw/raw2 /dev/sdb2
```

See *Configuring Quorum Partitions* in Section 2.4.4 for more information about setting up the quorum partitions. See *Creating Raw Devices* in Section 2.4.4 for more information on using the raw command to bind raw character devices to block devices.

Note

The rawdevices configuration must be performed on both cluster members.

3.1.2 Configuring the Cluster Alias

A **cluster alias** is a means of binding an IP address to one of the active cluster members. At any point in time this IP address will only be bound by one of the cluster members. This IP address is a useful convenience for system management and monitoring purposes. For example, suppose an administrator wishes to be able to telnet into an active cluster member, but does not care which cluster member. In this case, simply telnet to the cluster alias IP address (or associated name). The principal usage of the cluster alias is to enable the direction of the cluster GUI monitoring interface to connect to an active cluster member. In this manner, if either of the cluster members are not currently active it is still possible to derive cluster status while being abstracted from having to designate a specific cluster member to connect to.

While running cluconfig, you will be prompted as to whether or not you wish to configure a cluster alias. This appears as the following prompt:

```
Enter IP address for cluster alias [NONE]: 172.16.33.105
```

As shown above, the default value is set to **NONE**, which means that there is no cluster alias, but the user overrides this default and configures an alias using an IP address of 172.16.33.105. The IP address used for a cluster alias is distinct from the IP addresses associated with the cluster member's hostnames. It is also different from IP addresses associated with cluster services.

3.1.3 Enabling Remote Monitoring

While running cluconfig to specify cluster configuration parameters, the utility will prompt for the following:

```
Do you wish to allow remote monitoring of the cluster? yes/no [yes]:
```

If **yes** (the default) is answered, it enables the cluster to be remotely monitored by the cluster GUI. This is currently the only security provision controlling cluster monitoring access. The cluster GUI is only capable of performing monitoring requests and cannot make any active configuration changes.

If **no** is answered, then the cluster GUI can still be run locally on a cluster member, but remote operations will not be allowed.

3.1.4 Example of the cluconfig Utility

This section details an example of the cluconfig cluster configuration utility, which prompts you for information about the cluster members, and then enters the information into the cluster database. A copy of this is located in the `cluster.conf` file. In this example, the information entered in cluconfig prompts applies to the following configuration:

- On the **storage0** cluster system:

  ```
  Ethernet heartbeat channels: storage0
  Power switch serial port: /dev/ttyC0
  Power switch: RPS10
  Quorum partitions: /dev/raw/raw1 and /dev/raw/raw2
  ```

- On the **storage1** cluster system:

  ```
  Ethernet heartbeat channels: storage1 and cstorage1
  Serial heartbeat channel: /dev/ttyS1
  Power switch serial port: /dev/ttyS0
  Power switch: RPS10
  Quorum partitions: /dev/raw/raw1 and /dev/raw/raw2
  ```

- IP address to be used for the cluster alias: **10.0.0.154**

```
/sbin/cluconfig

Red Hat Cluster Manager Configuration Utility (running on storage0)

- Configuration file exists already.
    Would you like to use those prior settings as defaults? (yes/no) [yes]: yes
Enter cluster name [Development Cluster]:
    Enter IP address for cluster alias [10.0.0.154]: 10.0.0.154

--------------------------------
Information for Cluster Member 0
--------------------------------
   Enter name of cluster member [storage0]: storage0
Looking for host storage0 (may take a few seconds)...

Enter number of heartbeat channels (minimum = 1) [1]: 1
Information about Channel 0
Channel type: net or serial [net]:
Enter hostname of the cluster member on heartbeat channel 0 \
   [storage0]: storage0
Looking for host storage0 (may take a few seconds)...

Information about Quorum Partitions
Enter Primary Quorum Partition [/dev/raw/raw1]: /dev/raw/raw1
Enter Shadow Quorum Partition [/dev/raw/raw2]: /dev/raw/raw2

Information About the Power Switch That Power Cycles Member 'storage0'
Choose one of the following power switches:
  o NONE
  o RPS10
  o BAYTECH
  o APCSERIAL
  o APCMASTER
  o WTI_NPS
Power switch [RPS10]: RPS10
Enter the serial port connected to the power switch \
   [/dev/ttyS0]: /dev/ttyS0

--------------------------------
Information for Cluster Member 1
--------------------------------
Enter name of cluster member [storage1]: storage1
Looking for host storage1 (may take a few seconds)...

Information about Channel 0
```

```
Enter hostname of the cluster member on heartbeat channel 0 \
   [storage1]: storage1
Looking for host storage1 (may take a few seconds)...

Information about Quorum Partitions
Enter Primary Quorum Partition [/dev/raw/raw1]: /dev/raw/raw1
Enter Shadow Quorum Partition [/dev/raw/raw2]: /dev/raw/raw2

Information About the Power Switch That Power Cycles Member 'storage1'
Choose one of the following power switches:
  o NONE
  o RPS10
  o BAYTECH
  o APCSERIAL
  o APCMASTER
  o WTI_NPS
Power switch [RPS10]: RPS10
Enter the serial port connected to the power switch \
   [/dev/ttyS0]: /dev/ttyS0

Cluster name: Development Cluster
Cluster alias IP address: 10.0.0.154
Cluster alias netmask: 255.255.254.0

Serial port connected to the power switch \
   [/dev/ttyS0]: /dev/ttyS0

Cluster name: Development Cluster
Cluster alias IP address: 10.0.0.154
Cluster alias netmask: 255.255.254.0

--------------------
Member 0 Information
--------------------
Name: storage0
Primary quorum partition: /dev/raw/raw1
Shadow quorum partition: /dev/raw/raw2
Heartbeat channels: 1
Channel type: net, Name: storage0
Power switch IP address or hostname: storage0
Identifier on power controller for member storage0: storage0
--------------------
Member 1 Information
--------------------
Name: storage1
```

```
Primary quorum partition: /dev/raw/raw1
Shadow quorum partition: /dev/raw/raw2
Heartbeat channels: 1
Channel type: net, Name: storage1
Power switch IP address or hostname: storage1
Identifier on power controller for member storage1: storage1

--------------------------
Power Switch 0 Information
--------------------------
Power switch IP address or hostname: storage0
Type: RPS10
Login or port: /dev/ttyS0
Password: 10
--------------------------
Power Switch 1 Information
--------------------------
Power switch IP address or hostname: storage1
Type: RPS10
Login or port: /dev/ttyS0
Password: 10

Save the cluster member information? yes/no [yes]:
Writing to configuration file...done
Configuration information has been saved to /etc/cluster.conf.
----------------------------
Setting up Quorum Partitions
----------------------------
Running cludiskutil -I to initialize the quorum partitions: done
Saving configuration information to quorum partitions: done
Do you wish to allow remote monitoring of the cluster? yes/no \
   [yes]: yes

------------------------------------------------------------------

Configuration on this member is complete.

To configure the next member, invoke the following command on that system:

# /sbin/cluconfig --init=/dev/raw/raw1

See the manual to complete the cluster installation
```

3.2 Checking the Cluster Configuration

To ensure that the cluster software has been correctly configured, use the following tools located in the `/sbin` directory:

- Test the quorum partitions and ensure that they are accessible.

  Invoke the `cludiskutil` utility with the `-t` option to test the accessibility of the quorum partitions. See Section 3.2.1, *Testing the Quorum Partitions* for more information.

- Test the operation of the power switches.

  If power switches are used in the cluster hardware configuration, run the `clustonith` command on each cluster system to ensure that it can remotely power-cycle the other cluster system. Do not run this command while the cluster software is running. See Section 3.2.2, *Testing the Power Switches* for more information.

- Ensure that both cluster systems are running the same software version.

  Invoke the `rpm -q clumanager` command on each cluster system to display the revision of the installed cluster RPM.

The following section explains the cluster utilities in further detail.

3.2.1 Testing the Quorum Partitions

The quorum partitions must refer to the same physical device on both cluster systems. Invoke the `cludiskutil` utility with the `-t` command to test the quorum partitions and verify that they are accessible.

If the command succeeds, run the `cludiskutil -p` command on both cluster systems to display a summary of the header data structure for the quorum partitions. If the output is different on the systems, the quorum partitions do not point to the same devices on both systems. Check to make sure that the raw devices exist and are correctly specified in the `/etc/sysconfig/rawdevices` file. See *Configuring Quorum Partitions* in Section 2.4.4 for more information.

The following example shows that the quorum partitions refer to the same physical device on two cluster systems (devel0 and devel1):

```
/sbin/cludiskutil -p
----- Shared State Header ------
Magic# = 0x39119fcd
Version = 1
Updated on Thu Sep 14 05:43:18 2000
Updated by node 0
--------------------------------
```

```
/sbin/cludiskutil -p
----- Shared State Header ------
Magic# = 0x39119fcd
Version = 1
Updated on Thu Sep 14 05:43:18 2000
Updated by node 0
-------------------------------
```

The **Magic#** and **Version** fields will be the same for all cluster configurations. The last two lines of output indicate the date that the quorum partitions were initialized with `cludiskutil -I`, and the numeric identifier for the cluster system that invoked the initialization command.

If the output of the `cludiskutil` utility with the `-p` option is not the same on both cluster systems, perform the following:

- Examine the `/etc/sysconfig/rawdevices` file on each cluster system and ensure that the raw character devices and block devices for the primary and backup quorum partitions have been accurately specified. If they are not the same, edit the file and correct any mistakes. Then re-run the cluconfig utility. See Section 3.1.1, *Editing the `rawdevices` File* for more information.
- Ensure that you have created the raw devices for the quorum partitions on each cluster system. See *Configuring Quorum Partitions* in Section 2.4.4 for more information.
- On each cluster system, examine the system startup messages at the point where the system probes the SCSI subsystem to determine the bus configuration. Verify that both cluster systems identify the same shared storage devices and assign them the same name.
- Verify that a cluster system is not attempting to mount a file system on the quorum partition. For example, make sure that the actual device (for example, `/dev/sdb1`) is not included in an `/etc/fstab` file.

After performing these tasks, re-run the `cludiskutil` utility with the `-p` option.

3.2.2 Testing the Power Switches

If either network- or serial-attached power switches are employed in the cluster hardware configuration, install the cluster software and invoke the `clustonith` command to test the power switches. Invoke the command on each cluster system to ensure that it can remotely power-cycle the other cluster system. If testing is successful, then the cluster can be started. If using watchdog timers or the switch type "None", then this test can be omitted.

The `clustonith` command can accurately test a power switch only if the cluster software is not running. This is due to the fact that for serial attached switches, only one program at a time can access the serial port that connects a power switch to a cluster system. When the `clustonith` command is

invoked, it checks the status of the cluster software. If the cluster software is running, the command exits with a message to stop the cluster software.

The format of the `clustonith` command is as follows:

```
clustonith [-sSlLvr] [-t devicetype] [-F options-file] \
   [-p stonith-parameters]
Options:
-s              Silent mode, supresses error and log messages
-S              Display switch status
-l              List the hosts a switch can access
-L              List the set of supported switch types
-r hostname     Power cycle the specified host
-v              Increases verbose debugging level
```

When testing power switches, the first step is to ensure that each cluster member can successfully communicate with its attached power switch. The following example of the `clustonith` command output shows that the cluster member is able to communicate with its power switch:

```
 clustonith -S
WTI Network Power Switch device OK.
An example output of the clustonith command when it is unable
to communicate with its power switch appears below:
  clustonith -S
Unable to determine power switch type.
Unable to determine default power switch type.
```

The above error could be indicative of the following types of problems:

- For serial attached power switches:
  - Verify that the device special file for the remote power switch connection serial port (for example, `/dev/ttyS0`) is specified correctly in the cluster database, as established via the `cluconfig` command. If necessary, use a terminal emulation package such as minicom to test if the cluster system can access the serial port.
  - Ensure that a non-cluster program (for example, a `getty` program) is not using the serial port for the remote power switch connection. You can use the `lsof` command to perform this task.
  - Check that the cable connection to the remote power switch is correct. Verify that the correct type of cable is used (for example, an RPS-10 power switch requires a null modem cable), and that all connections are securely fastened.
  - Verify that any physical dip switches or rotary switches on the power switch are set properly. If using an RPS-10 power switch, see Section A.1.1, *Setting up RPS-10 Power Switches* for more information.
- For network based power switches:

– Verify that the network connection to network-based switches is operational. Most switches have a link light that indicates connectivity.

– It should be possible to `ping` the network switch; if not, then the switch may not be properly configured for its network parameters.

– Verify that the correct password and login name (depending on switch type) have been specified in the cluster configuration database (as established by running `cluconfig`). A useful diagnostic approach is to verify telnet access to the network switch using the same parameters as specified in the cluster configuration.

After successfully verifying communication with the switch, attempt to power cycle the other cluster member. Prior to doing this, it would is recommended to verify that the other cluster member is not actively performing any important functions (such as serving cluster services to active clients). The following command depicts a successful power cycle operation:

```
clustonith -r clu3
Successfully power cycled host clu3.
```

3.2.3 Displaying the Cluster Software Version

Invoke the `rpm -qa clumanager` command to display the revision of the installed cluster RPM. Ensure that both cluster systems are running the same version.

3.3 Configuring syslog Event Logging

It is possible to edit the `/etc/syslog.conf` file to enable the cluster to log events to a file that is different from the `/var/log/messages` log file. Logging cluster messages to a separate file will help to diagnose problems more clearly.

The cluster systems use the syslogd daemon to log cluster-related events to a file, as specified in the `/etc/syslog.conf` file. The log file facilitates diagnosis of problems in the cluster. It is recommended to set up event logging so that the syslogd daemon logs cluster messages only from the system on which it is running. Therefore, you need to examine the log files on both cluster systems to get a comprehensive view of the cluster.

The syslogd daemon logs messages from the following cluster daemons:

- `cluquorumd` — Quorum daemon
- `clusvcmgrd` — Service manager daemon
- `clupowerd` — Power daemon
- `cluhbd` — Heartbeat daemon
- `clumibd` — Administrative system monitoring daemon

The importance of an event determines the severity level of the log entry. Important events should be investigated before they affect cluster availability. The cluster can log messages with the following severity levels, listed in order of severity level:

- **emerg** — The cluster system is unusable.
- **alert** — Action must be taken immediately to address the problem.
- **crit** — A critical condition has occurred.
- **err** — An error has occurred.
- **warning** — A significant event that may require attention has occurred.
- **notice** — An event that does not affect system operation has occurred.
- **info** — An normal cluster operation has occurred.
- **debug** — Diagnostic output detailing normal cluster operations.

The default logging severity levels for the cluster daemons are **warning** and higher.

Examples of log file entries are as follows:

```
May 31 20:42:06 clu2 clusvcmgrd[992]: <info>  Service Manager starting
May 31 20:42:06 clu2 clusvcmgrd[992]: <info>  mount.ksh info: /dev/sda3 \
  is not mounted
May 31 20:49:38 clu2 clulog[1294]: <notice>  stop_service.ksh notice: \
  Stopping service dbase_home
May 31 20:49:39 clu2 clusvcmgrd[1287]: <notice>  Service Manager received \
  a NODE_UP event for stor5
Jun 01 12:56:51 clu2 cluquorumd[1640]: <err>  updateMyTimestamp: unable to \
  update status block.
Jun 01 12:34:24 clu2 cluquorumd[1268]: <warning>  Initiating cluster stop
Jun 01 12:34:24 clu2 cluquorumd[1268]: <warning>  Completed cluster stop
Jul 27 15:28:40 clu2 cluquorumd[390]: <err>  shoot_partner: successfully shot partner.
   [1]       [2]      [3]           [4]               [5]
```

Each entry in the log file contains the following information:

- [1]Timestamp
- [2] Cluster system on which the event was logged
- [3] Subsystem that generated the event
- [4] Severity level of the event
- [5] Description of the event

After configuring the cluster software, optionally edit the /etc/syslog.conf file to enable the cluster to log events to a file that is different from the default log file, /var/log/messages. The cluster utilities and daemons log their messages using a syslog tag called *local4*. Using a cluster-specific log file facilitates cluster monitoring and problem solving. To log cluster events to both the /var/log/cluster and /var/log/messages files, add lines similar to the following to the /etc/syslog.conf file:

```
#
# Cluster messages coming in on local4 go to /var/log/cluster
#
local4.*                        /var/log/cluster
```

To prevent the duplication of messages and log cluster events only to the /var/log/cluster file, add lines similar to the following to the /etc/syslog.conf file:

```
# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;news.none;authpriv.none;local4.none   /var/log/messages
```

To apply the previous changes, you can invoke the killall -HUP syslogd command, or restart syslog with a command similar to /etc/rc.d/init.d/syslog restart.

In addition, it is possible to modify the severity level of the events that are logged by the individual cluster daemons. See Section 8.6, *Modifying Cluster Event Logging* for more information.

3.4 Using the cluadmin Utility

The cluadmin utility provides a command-line user interface that enables an administrator to monitor and manage the cluster systems and services. Use the cluadmin utility to perform the following tasks:

- Add, modify, and delete services
- Disable and enable services
- Display cluster and service status
- Modify cluster daemon event logging
- Backup and restore the cluster database

The cluster uses an advisory lock to prevent the cluster database from being simultaneously modified by multiple users on either cluster system. Users can only modify the database if they hold the advisory lock.

When the cluadmin utility is invoked, the cluster software checks if the lock is already assigned to a user. If the lock is not already assigned, the cluster software assigns the requesting user the lock. When the user exits from the cluadmin utility, the lock is relinquished.

If another user holds the lock, a warning will be displayed indicating that there is already a lock on the database. The cluster software allows for the option of taking the lock. If the lock is taken by the current requesting user, the previous holder of the lock can no longer modify the cluster database.

Take the lock only if necessary, because uncoordinated simultaneous configuration sessions may cause unpredictable cluster behavior. In addition, it is recommended to make only one change to the cluster database (for example, adding, modifying, or deleting services) at a time. The `cluadmin` command line options are as follows:

`-d` or `--debug`

Displays extensive diagnostic information.

`-h`, `-?`, or `--help`

Displays help about the utility, and then exits.

`-n` or `--nointeractive`

Bypasses the `cluadmin` utility's top-level command loop processing. This option is used for `cluadmin` debugging purposes.

`-t` or `--tcl`

Adds a Tcl command to the `cluadmin` utility's top-level command interpreter. To pass a Tcl command directly to the utility's internal Tcl interpreter, at the `cluadmin>` prompt, preface the Tcl command with `tcl`. This option is used for `cluadmin` debugging purposes.

`-V` or `--version`

Displays information about the current version of `cluadmin`.

When the `cluadmin` utility is invoked without the `-n` option, the `cluadmin>` prompt appears. You can then specify commands and subcommands. Table 3–1, *`cluadmin` Commands* describes the commands and subcommands for the `cluadmin` utility:

Table 3–1 cluadmin Commands

clu-admin Com-mand	cluadmin Subcom-mand	Description	Example
help	None	Displays help for the specified cluadmin command or subcommand.	help service add
cluster	status	Displays a snapshot of the current cluster status. See Section 8.1, *Displaying Cluster and Service Status* for information.	cluster status
	loglevel	Sets the logging for the specified cluster daemon to the specified severity level. See Section 8.6, *Modifying Cluster Event Logging* for information.	cluster loglevel cluquorumd 7
	reload	Forces the cluster daemons to re-read the cluster configuration database. See Section 8.8, *Reloading the Cluster Database* for information.	cluster reload
	name	Sets the name of the cluster to the specified name. The cluster name is included in the output of the clustat cluster monitoring command. See Section 8.9, *Changing the Cluster Name* for information.	cluster name dbasecluster
	backup	Saves a copy of the cluster configuration database in the /etc/cluster.conf.bak file. See Section 8.5, *Backing Up and Restoring the Cluster Database* for information.	cluster backup

clu-admin Com-mand	cluadmin Subcom-mand	Description	Example
	restore	Restores the cluster configuration database from the backup copy in the /etc/cluster.conf.bak file. See Section 8.5, *Backing Up and Restoring the Cluster Database* for information.	cluster restore
	saveas	Saves the cluster configuration database to the specified file. See Section 8.5, *Backing Up and Restoring the Cluster Database* for information.	cluster saveas clus-ter_backup.conf
	restore-from	Restores the cluster configuration database from the specified file. See Section 8.5, *Backing Up and Restoring the Cluster Database* for information.	cluster re-storefrom clus-ter_backup.conf
service	add	Adds a cluster service to the cluster database. The command prompts you for information about service resources and properties. See Section 4.1, *Configuring a Service* for information.	service add
	modify	Modifies the resources or properties of the specified service. You can modify any of the information that you specified when the service was created. See Section 4.5, *Modifying a Service* for information.	service modify dbservice
	show state	Displays the current status of all services or the specified service. See Section 8.1, *Displaying Cluster and Service Status* for information.	service show state dbservice

`cluadmin` Command	**`cluadmin` Subcommand**	**Description**	**Example**
	`relocate`	Causes a service to be stopped on the cluster member its currently running on and restarted on the other. Refer to Section 4.6, *Relocating a Service* for more information.	`service relocate nfs1`
	`show config`	Displays the current configuration for the specified service. See Section 4.2, *Displaying a Service Configuration* for information.	`service show config dbservice`
	`disable`	Stops the specified service. You must enable a service to make it available again. See Section 4.3, *Disabling a Service* for information.	`service disable dbservice`
	`enable`	Starts the specified disabled service. See Section 4.4, *Enabling a Service* for information.	`service enable dbservice`
	`delete`	Deletes the specified service from the cluster configuration database. See Section 4.7, *Deleting a Service* for information.	`service delete dbservice`
`apropos`	None	Displays the `cluadmin` commands that match the specified character string argument or, if no argument is specified, displays all `cluadmin` commands.	`apropos service`
`clear`	None	Clears the screen display.	`clear`
`exit`	None	Exits from `cluadmin`.	`exit`
`quit`	None	Exits from `cluadmin`.	`quit`

While using the `cluadmin` utility, press the [Tab] key to help identify `cluadmin` commands. For example, pressing the [Tab] key at the `cluadmin>` prompt displays a list of all the commands. Entering a letter at the prompt and then pressing the [Tab] key displays the commands that begin with the specified letter. Specifying a command and then pressing the [Tab] key displays a list of all the subcommands that can be specified with that command.

Users can additionally display the history of `cluadmin` commands by pressing the up arrow and down arrow keys at the prompt. The command history is stored in the `.cluadmin_history` file in the user's home directory.

4 Service Configuration and Administration

The following sections describe how to configure, display, enable/disable, modify, relocate, and delete a service, as well as how to handle services which fail to start.

4.1 Configuring a Service

The cluster systems must be prepared before any attempts to configure a service. For example, set up disk storage or applications used in the services. Then, add information about the service properties and resources to the cluster database by using the `cluadmin` utility. This information is used as parameters to scripts that start and stop the service.

To configure a service, follow these steps:

1. If applicable, create a script that will start and stop the application used in the service. See Section 4.1.2, *Creating Service Scripts* for information.
2. Gather information about service resources and properties. See Section 4.1.1, *Gathering Service Information* for information.
3. Set up the file systems or raw devices that the service will use. See Section 4.1.3, *Configuring Service Disk Storage* for information.
4. Ensure that the application software can run on each cluster system and that the service script, if any, can start and stop the service application. See Section 4.1.4, *Verifying Application Software and Service Scripts* for information.
5. Back up the `/etc/cluster.conf` file. See Section 8.5, *Backing Up and Restoring the Cluster Database* for information.
6. Invoke the `cluadmin` utility and specify the `service add` command. The `cluadmin` utility will prompt for information about the service resources and properties obtained in Step 2. If the service passes the configuration checks, it will be started on the user-designated cluster system, unless the user wants to keep the service disabled. For example:

```
cluadmin> service add
```

For more information about adding a cluster service, see the following:

- Section 5.1, *Setting Up an Oracle Service*
- Section 5.3, *Setting Up a MySQL Service*

- Section 5.4, *Setting Up a DB2 Service*
- Section 6.1, *Setting Up an NFS Service*
- Section 6.2, *Setting Up a High Availability Samba Service*
- Section 7.1, *Setting Up an Apache Service*

4.1.1 Gathering Service Information

Before creating a service, gather all available information about the service resources and properties. When adding a service to the cluster database, the `cluadmin` utility will prompt for this information.

In some cases, it is possible to specify multiple resources for a service (for example, multiple IP addresses and disk devices).

The service properties and resources that a user is able to specify are described in the following table.

Table 4–1 Service Property and Resource Information

Service Property or Resource	Description
Service name	Each service must have a unique name. A service name can consist of one to 63 characters and must consist of a combination of letters (either uppercase or lowercase), integers, underscores, periods, and dashes. However, a service name must begin with a letter or an underscore.
Preferred member	Specify the cluster system, if any, on which the service will run unless failover has occurred or unless the service is manually relocated.
Preferred member relocation policy	When enabled, this policy will automatically relocate a service to its preferred member when that system joins the cluster. If this policy is disabled, the service will remain running on the non-preferred member. For example, if an administrator enables this policy and the failed preferred member for the service reboots and joins the cluster, the service will automatically restart on the preferred member.
Script location	If applicable, specify the full path name for the script that will be used to start and stop the service. See Section 4.1.2, *Creating Service Scripts* for more information.

Service Property or Resource	Description
IP address	One or more Internet protocol (IP) addresses may be assigned to a service. This IP address (sometimes called a "floating" IP address) is different from the IP address associated with the host name Ethernet interface for a cluster system, because it is automatically relocated along with the service resources, when failover occurs. If clients use this IP address to access the service, they will not know which cluster system is running the service, and failover is transparent to the clients. Note that cluster members must have network interface cards configured in the IP subnet of each IP address used in a service. Netmask and broadcast addresses for each IP address can also be specified; if they are not, then the cluster uses the netmask and broadcast addresses from the network interconnect in the subnet.
Disk partition	Specify each shared disk partition used in a service.
Mount points, file system types, mount options, NFS export options, and Samba shares	If using a file system, specify the type of file system, the mount point, and any mount options. Mount options available to specify are the standard file system mount options that are described in the `mount`(8) manual page. It is not necessary to provide mount information for raw devices (if used in a service). The ext2 and ext3 file systems are the recommended file systems for a cluster. Although a different file system may be used (such as reiserfs), only ext2 and ext3 have been thoroughly tested and are supported. Specify whether or not to enable forced unmount for a file system. Forced unmount allows the cluster service management infrastructure to unmount a file system even if it is being accessed by an application or user (that is, even if the file system is "busy"). This is accomplished by terminating any applications that are accessing the file system. `cluadmin` will prompt whether or not to NFS export the filesystem and if so, what access permissions should be applied. Refer to Section 6.1, *Setting Up an NFS Service* for details. Specify whether or not to make the filesystem accessible to Windows clients via Samba.

Service Property or Resource	Description
Service Check Interval	Specifies the frequency (in seconds) that the system will check the health of the application associated with the service. For example, it will verify that the necessary NFS or Samba daemons are running. For additional service types, the monitoring consists of examining the return status when calling the "status" clause of the application service script. Specifying a value of 0 for the service check interval will disable checking.
Disable service policy	If a user does not want to automatically start a service after it is added to the cluster, it is possible to keep the new service disabled until the user enables it.

4.1.2 Creating Service Scripts

The cluster infrastructure starts and stops service to specified applications by running service specific scripts. For both NFS and Samba services, the associated scripts are built into the cluster services infrastructure. Consequently, when running `cluadmin` to configure NFS and Samba services, do not enter a service script name. For other application types it is necessary to designate a service script. For example, when configuring a database application in `cluadmin`, specify the fully qualified pathname of the corresponding database start script.

The format of the service scripts conforms to the conventions followed by the System V init scripts. This convention dictates that the scripts have a `start`, `stop`, and `status` clause. These should return an exit status of 0 on success. The cluster infrastructure will stop a cluster service that fails to successfully start. Inability of a service to start will result in the service being placed in a disabled state.

In addition to performing the stop and start functions, service scripts are also used for application service monitoring purposes. This is performed by calling the `status` clause of a service script. To enable service monitoring, specify a nonzero value for the `Status check interval:` prompt in `cluadmin`. If a nonzero exit is returned by a status check request to the service script, then the cluster infrastructure will first attempt to restart the application on the member it was previously running on. Status functions do not have to be fully implemented in service scripts. If no real monitoring is performed by the script, then a stub `status` clause should be present which returns success.

The operations performed within the status clause of an application can be tailored to best meet the application's needs as well as site-specific parameters. For example, a simple status check for a database would consist of verifying that the database process is still running. A more comprehensive check would consist of a database table query.

The /usr/share/cluster/doc/services/examples directory contains a template that can be used to create service scripts, in addition to examples of scripts. See Section 5.1, *Setting Up an Oracle Service*, Section 5.3, *Setting Up a MySQL Service*, Section 7.1, *Setting Up an Apache Service*, and Section 5.4, *Setting Up a DB2 Service* for sample scripts.

4.1.3 Configuring Service Disk Storage

Prior to creating a service, set up the shared file systems and raw devices that the service will use. See Section 2.4.4, *Configuring Shared Disk Storage* for more information.

If employing raw devices in a cluster service, it is possible to use the /etc/sysconfig/rawdevices file to bind the devices at boot time. Edit the file and specify the raw character devices and block devices that are to be bound each time the system boots. See Section 3.1.1, *Editing the rawdevices File* for more information.

Note that software RAID, and host-based RAID are not supported for shared disk storage. Only certified SCSI adapter-based RAID cards can be used for shared disk storage.

Administrators should adhere to the following *service disk storage recommendations*:

- For optimal performance, use a 4 KB block size when creating file systems. Note that some of the mkfs file system build utilities default to a 1 KB block size, which can cause long fsck times.
- To facilitate quicker failover times, it is recommended that the ext3 filesystem be used. Refer to *Creating File Systems* in Section 2.4.4 for more information.
- For large file systems, use the mount command with the nocheck option to bypass code that checks all the block groups on the partition. Specifying the nocheck option can significantly decrease the time required to mount a large file system.

4.1.4 Verifying Application Software and Service Scripts

Prior to setting up a service, install any application that will be used in a service on each system. After installing the application, verify that the application runs and can access shared disk storage. To prevent data corruption, do not run the application simultaneously on both systems.

If using a script to start and stop the service application, install and test the script on both cluster systems, and verify that it can be used to start and stop the application. See Section 4.1.2, *Creating Service Scripts* for information.

4.2 Displaying a Service Configuration

Administrators can display detailed information about the configuration of a service. This information includes the following:

- Service name

- Whether the service was disabled after it was added
- Preferred member system
- Whether the service will relocate to its preferred member when it joins the cluster
- Service Monitoring interval
- Service start script location IP addresses
- Disk partitions
- File system type
- Mount points and mount options
- NFS exports
- Samba shares

To display cluster service status, see Section 8.1, *Displaying Cluster and Service Status*.

To display service configuration information, invoke the `cluadmin` utility and specify the `service show config` command. For example:

```
cluadmin> service show config
  0) dummy
  1) nfs_pref_clu4
  2) nfs_pref_clu3
  3) nfs_nopref
  4) ext3
  5) nfs_eng
  6) nfs_engineering
  c) cancel

Choose service: 6
name: nfs_engineering
disabled: no
preferred node: clu3
relocate: yes
IP address 0: 172.16.33.164
device 0: /dev/sdb11
  mount point, device 0: /mnt/users/engineering
  mount fstype, device 0: ext2
  mount options, device 0: rw,nosuid,sync
  force unmount, device 0: yes
NFS export 0: /mnt/users/engineering/ferris
  Client 0: ferris, rw
NFS export 0: /mnt/users/engineering/denham
  Client 0: denham, rw
```

```
NFS export 0: /mnt/users/engineering/brown
  Client 0: brown, rw
cluadmin>
```

If the name of the service is known, it can be specified with the `service show config` *`service_name`* command.

4.3 Disabling a Service

A running service can be disabled in order to stop the service and make it unavailable. Once disabled, a service can then be re-enabled. See Section 4.4, *Enabling a Service* for information.

There are several situations in which a running service may need to be disabled:

- To modify a service

  A running service must be disabled before it can be modified. See Section 4.5, *Modifying a Service* for more information.

- To temporarily stop a service

  A running service can be disabled, making it unavailable to clients without having to completely delete the service.

To disable a running service, invoke the `cluadmin` utility and specify the `service disable` *`service_name`* command. For example:

```
cluadmin> service disable user_home
Are you sure? (yes/no/?) y
notice: Stopping service user_home  ...
notice: Service user_home is disabled
service user_home disabled
```

4.4 Enabling a Service

A disabled service can be enabled to start the service and make it available.

To enable a disabled service, invoke the `cluadmin` utility and specify the `service enable` *`service_name`* command:

```
cluadmin> service enable user_home
Are you sure? (yes/no/?) y
notice: Starting service user_home ...
notice: Service user_home is running
service user_home enabled
```

4.5 Modifying a Service

All properties that were specified when a service was created can be modified. For example, specified IP addresses can be changed. More resources can also be added to a service (for example, more file systems). See Section 4.1.1, *Gathering Service Information* for information.

A service must be disabled before it can be modified. If an attempt is made to modify a running service, the cluster manager will prompt to disable it. See Section 4.3, *Disabling a Service* for more information.

Because a service is unavailable while being modified, be sure to gather all the necessary service information before disabling it in order to minimize service down time. In addition, back up the cluster database before modifying a service. See Section 8.5, *Backing Up and Restoring the Cluster Database* for more information.

To modify a disabled service, invoke the `cluadmin` utility and specify the `service modify` *`service_name`* command.

```
cluadmin> service modify web1
```

Service properties and resources can also be modified, as needed. The cluster will check the service modifications and allow correction of any mistakes. The cluster will verify the submitted service modification and then start the service, unless prompted to keep the service disabled. If changes are not submitted, the service will be started, if possible, using the original configuration.

4.6 Relocating a Service

In addition to providing automatic service failover, a cluster enables administrators to cleanly stop a service on one cluster system and then start it on the other cluster system. This service relocation functionality allows administrators to perform maintenance on a cluster system while maintaining application and data availability.

To relocate a service by using the `cluadmin` utility, invoke the `service relocate` command. For example:

```
cluadmin> service relocate nfs1
```

If a specific service is not designated, then a menu of running services will appear to choose from.

If an error occurs while attempting to relocate a service, a useful diagnostic approach would be to try to disable the individual service and then enable the service on the other cluster member.

4.7 Deleting a Service

A cluster service can be deleted. Note that the cluster database should be backed up before deleting a service. See Section 8.5, *Backing Up and Restoring the Cluster Database* for information.

To delete a service by using the `cluadmin` utility, follow these steps:

1. Invoke the `cluadmin` utility on the cluster system that is running the service, and specify the `service disable` *`service_name`* command. See Section 4.3, *Disabling a Service* for more information.
2. Specify the `service delete` *`service_name`* command to delete the service.

For example:

```
cluadmin> service disable user_home
Are you sure? (yes/no/?) y
notice: Stopping service user_home  ...
notice: Service user_home is disabled
service user_home disabled

cluadmin> service delete user_home
Deleting user_home, are you sure? (yes/no/?): y
user_home deleted.
cluadmin>
```

4.8 Handling Services that Fail to Start

The cluster puts a service into the **disabled** state if it is unable to successfully start the service. A **disabled** state can be caused by various problems, such as a service start did not succeed, and the subsequent service stop also failed.

Be sure to carefully handle failed services. If service resources are still configured on the owner system, starting the service on the other cluster system may cause significant problems. For example, if a file system remains mounted on the owner system, and you start the service on the other cluster system, the file system will be mounted on both systems, which can cause data corruption. If the enable fails, the service will remain in the **disabled** state.

It is possible to modify a service that is in the **disabled** state. It may be necessary to do this in order to correct the problem that caused the **disabled** state. After modifying the service, it will be enabled on the owner system, if possible, or it will remain in the **disabled** state. The following list details steps to follow in the event of service failure:

1. Modify cluster event logging to log debugging messages. See Section 8.6, *Modifying Cluster Event Logging* for more information.

2. Use the `cluadmin` utility to attempt to enable or disable the service on the cluster system that owns the service. See Section 4.3, *Disabling a Service* and Section 4.4, *Enabling a Service* for more information.
3. If the service does not start or stop on the owner system, examine the `/var/log/messages` log file, and diagnose and correct the problem. You may need to modify the service to fix incorrect information in the cluster database (for example, an incorrect start script), or you may need to perform manual tasks on the owner system (for example, unmounting file systems).
4. Repeat the attempt to enable or disable the service on the owner system. If repeated attempts fail to correct the problem and enable or disable the service, reboot the owner system.
5. If still unable to successfully start the service, verify that the service can be manually restarted outside of the cluster framework. For example, this may include manually mounting the filesystems and manually running the service start script.

5 Database Services

This chapter contains instructions for configuring Red Hat Linux Advanced Server to make database services highly available.

Note

The following descriptions present example database configuration instructions. Be aware that differences may exist in newer versions of each database product. Consequently, this information may not be directly applicable.

5.1 Setting Up an Oracle Service

A database service can serve highly-available data to a database application. The application can then provide network access to database client systems, such as Web servers. If the service fails over, the application accesses the shared database data through the new cluster system. A network-accessible database service is usually assigned an IP address, which is failed over along with the service to maintain transparent access for clients.

This section provides an example of setting up a cluster service for an Oracle database. Although the variables used in the service scripts depend on the specific Oracle configuration, the example may aid in setting up a service for individual environments. See Section 5.2, *Tuning Oracle Services* for information about improving service performance.

In the example that follows:

- The service includes one IP address for the Oracle clients to use.
- The service has two mounted file systems, one for the Oracle software (`/u01`) and the other for the Oracle database (`/u02`), which were set up before the service was added.
- An Oracle administration account with the name **oracle** was created on both cluster systems before the service was added.
- Network access in this example is through Perl DBI proxy.
- The administration directory is on a shared disk that is used in conjunction with the Oracle service (for example, `/u01/app/oracle/admin/db1`).

The Oracle service example uses five scripts that must be placed in `/home/oracle` and owned by the Oracle administration account. The `oracle` script is used to start and stop the Oracle service. Specify this script when you add the service. This script calls the other Oracle example scripts. The `startdb` and `stopdb` scripts start and stop the database. The `startdbi` and `stopdbi` scripts

start and stop a Web application that has been written using Perl scripts and modules and is used to interact with the Oracle database. Note that there are many ways for an application to interact with an Oracle database.

The following is an example of the `oracle` script, which is used to start and stop the Oracle service. Note that the script is run as user oracle, instead of root.

```
#!/bin/sh
#
# Cluster service script to start/stop oracle
#

cd /home/oracle

case $1 in
'start')
    su - oracle -c ./startdbi
    su - oracle -c ./startdb
    ;;
'stop')
    su - oracle -c ./stopdb
    su - oracle -c ./stopdbi
    ;;
esac
```

The following is an example of the `startdb` script, which is used to start the Oracle Database Server instance:

```
#!/bin/sh
#

#
# Script to start the Oracle Database Server instance.
#
###############################################################################
#
# ORACLE_RELEASE
#
# Specifies the Oracle product release.
#
###############################################################################

ORACLE_RELEASE=8.1.6

###############################################################################
#
```

```
# ORACLE_SID
#
# Specifies the Oracle system identifier or "sid", which is the name of
# the Oracle Server instance.
#
###############################################################################

export ORACLE_SID=TESTDB

###############################################################################
#
# ORACLE_BASE
#
# Specifies the directory at the top of the Oracle software product and
# administrative file structure.
#
###############################################################################

export ORACLE_BASE=/u01/app/oracle

###############################################################################
#
# ORACLE_HOME
#
# Specifies the directory containing the software for a given release.
# The Oracle recommended value is $ORACLE_BASE/product/<release>
#
###############################################################################

export ORACLE_HOME=/u01/app/oracle/product/${ORACLE_RELEASE}

###############################################################################
#
# LD_LIBRARY_PATH
#
# Required when using Oracle products that use shared libraries.
#
###############################################################################

export LD_LIBRARY_PATH=/u01/app/oracle/product/${ORACLE_RELEASE}/lib

###############################################################################
#
# PATH
#
```

```
# Verify that the users search path includes $ORCLE_HOME/bin
#
###############################################################################

export PATH=$PATH:/u01/app/oracle/product/${ORACLE_RELEASE}/bin

###############################################################################
#
# This does the actual work.
#
# The oracle server manager is used to start the Oracle Server instance
# based on the initSID.ora initialization parameters file specified.
#
###############################################################################

/u01/app/oracle/product/${ORACLE_RELEASE}/bin/svrmgrl << EOF
spool /home/oracle/startdb.log
connect internal;
startup pfile = /u01/app/oracle/admin/db1/pfile/initTESTDB.ora open;
spool off
EOF

exit 0
```

The following is an example of the `stopdb` script, which is used to stop the Oracle Database Server instance:

```
#!/bin/sh
#
#
# Script to STOP the Oracle Database Server instance.
#
###############################################################################
#
# ORACLE_RELEASE
#
# Specifies the Oracle product release.
#
###############################################################################

ORACLE_RELEASE=8.1.6

###############################################################################
#
# ORACLE_SID
#
```

```
# Specifies the Oracle system identifier or "sid", which is the name
# of the Oracle Server instance.
#
###############################################################################

export ORACLE_SID=TESTDB

###############################################################################
#
# ORACLE_BASE
#
# Specifies the directory at the top of the Oracle software product
# and administrative file structure.
#
###############################################################################

export ORACLE_BASE=/u01/app/oracle

###############################################################################
#
# ORACLE_HOME
#
# Specifies the directory containing the software for a given release.
# The Oracle recommended value is $ORACLE_BASE/product/<release>
#
###############################################################################

export ORACLE_HOME=/u01/app/oracle/product/${ORACLE_RELEASE}

###############################################################################
#
# LD_LIBRARY_PATH
#
# Required when using Oracle products that use shared libraries.
#
###############################################################################

export LD_LIBRARY_PATH=/u01/app/oracle/product/${ORACLE_RELEASE}/lib

###############################################################################
#
# PATH
#
# Verify that the users search path includes $ORCLE_HOME/bin
#
```

```
#########################################################################

export PATH=$PATH:/u01/app/oracle/product/${ORACLE_RELEASE}/bin

#########################################################################
#
# This does the actual work.
#
# The oracle server manager is used to STOP the Oracle Server instance
# in a tidy fashion.
#
#########################################################################

/u01/app/oracle/product/${ORACLE_RELEASE}/bin/svrmgrl << EOF
spool /home/oracle/stopdb.log
connect internal;
shutdown abort;
spool off
EOF

exit 0
```

The following is an example of the `startdbi` script, which is used to start a networking DBI proxy daemon:

```
#!/bin/sh
#
#
#########################################################################
#
# This script allows are Web Server application (perl scripts) to
# work in a distributed environment. The technology we use is
# base upon the DBD::Oracle/DBI CPAN perl modules.
#
# This script STARTS the networking DBI Proxy daemon.
#
#########################################################################

export ORACLE_RELEASE=8.1.6
export ORACLE_SID=TESTDB
export ORACLE_BASE=/u01/app/oracle
export ORACLE_HOME=/u01/app/oracle/product/${ORACLE_RELEASE}
export LD_LIBRARY_PATH=/u01/app/oracle/product/${ORACLE_RELEASE}/lib
export PATH=$PATH:/u01/app/oracle/product/${ORACLE_RELEASE}/bin
```

```
#
# This line does the real work.
#

/usr/bin/dbiproxy --logfile /home/oracle/dbiproxy.log --localport 1100 &

exit 0
```

The following is an example of the `stopdbi` script, which is used to stop a networking DBI proxy daemon:

```
#!/bin/sh
#
#
######################################################################
#
# Our Web Server application (perl scripts) work in a distributed
# environment. The technology we use is base upon the
# DBD::Oracle/DBI CPAN perl modules.
#
# This script STOPS the required networking DBI Proxy daemon.
#
######################################################################


PIDS=$(ps ax | grep /usr/bin/dbiproxy | awk '{print $1}')

for pid in $PIDS
do
        kill -9 $pid
done

exit 0
```

The following example shows how to use `cluadmin` to add an Oracle service.

```
cluadmin> service add oracle

  The user interface will prompt you for information about the service.
  Not all information is required for all services.

  Enter a question mark (?) at a prompt to obtain help.

  Enter a colon (:) and a single-character command at a prompt to do
  one of the following:
```

```
  c - Cancel and return to the top-level cluadmin command
  r - Restart to the initial prompt while keeping previous responses
  p - Proceed with the next prompt

Preferred member [None]: ministor0
Relocate when the preferred member joins the cluster (yes/no/?) \
 [no]: yes
User script (e.g., /usr/foo/script or None) \
 [None]: /home/oracle/oracle

Do you want to add an IP address to the service (yes/no/?): yes

        IP Address Information

IP address: 10.1.16.132
Netmask (e.g. 255.255.255.0 or None) [None]: 255.255.255.0
Broadcast (e.g. X.Y.Z.255 or None) [None]: 10.1.16.255

Do you want to (a)dd, (m)odify, (d)elete or (s)how an IP address,
 or are you (f)inished adding IP addresses: f

Do you want to add a disk device to the service (yes/no/?): yes

        Disk Device Information

Device special file (e.g., /dev/sda1): /dev/sda1
Filesystem type (e.g., ext2, reiserfs, ext3 or None): ext2
Mount point (e.g., /usr/mnt/service1 or None) [None]: /u01
Mount options (e.g., rw, nosuid): [Return]
Forced unmount support (yes/no/?) [no]: yes

Do you want to (a)dd, (m)odify, (d)elete or (s)how devices,
  or are you (f)inished adding device information: a

Device special file (e.g., /dev/sda1): /dev/sda2
Filesystem type (e.g., ext2, reiserfs, ext3 or None): ext2
Mount point (e.g., /usr/mnt/service1 or None) [None]: /u02
Mount options (e.g., rw, nosuid): [Return]
Forced unmount support (yes/no/?) [no]: yes


Do you want to (a)dd, (m)odify, (d)elete or (s)how devices,
  or are you (f)inished adding devices: f
```

```
Disable service (yes/no/?) [no]: no

name: oracle
disabled: no
preferred node: ministor0
relocate: yes
user script: /home/oracle/oracle
IP address 0: 10.1.16.132
  netmask 0: 255.255.255.0
  broadcast 0: 10.1.16.255
device 0: /dev/sda1
  mount point, device 0: /u01
  mount fstype, device 0: ext2
  force unmount, device 0: yes
device 1: /dev/sda2
  mount point, device 1: /u02
  mount fstype, device 1: ext2
  force unmount, device 1: yes

Add oracle service as shown? (yes/no/?) y
notice: Starting service oracle ...
info: Starting IP address 10.1.16.132
info: Sending Gratuitous arp for 10.1.16.132 (00:90:27:EB:56:B8)
notice: Running user script '/home/oracle/oracle start'
notice, Server starting
Added oracle.
cluadmin>
```

5.2 Tuning Oracle Services

The Oracle database recovery time after a failover is directly proportional to the number of outstanding transactions and the size of the database. The following parameters control database recovery time:

- **LOG_CHECKPOINT_TIMEOUT**
- **LOG_CHECKPOINT_INTERVAL**
- **FAST_START_IO_TARGET**
- **REDO_LOG_FILE_SIZES**

To minimize recovery time, set the previous parameters to relatively low values. Note that excessively low values will adversely impact performance. Try different values in order to find the optimal value.

Oracle provides additional tuning parameters that control the number of database transaction retries and the retry delay time. Be sure that these values are large enough to accommodate the failover time

in the cluster environment. This will ensure that failover is transparent to database client application programs and does not require programs to reconnect.

5.3 Setting Up a MySQL Service

A database service can serve highly-available data to a MySQL database application. The application can then provide network access to database client systems, such as Web servers. If the service fails over, the application accesses the shared database data through the new cluster system. A network-accessible database service is usually assigned one IP address, which is failed over along with the service to maintain transparent access for clients.

An example of a MySQL database service is as follows:

- The MySQL server and the database instance both reside on a file system that is located on a disk partition on shared storage. This allows the database data and its run-time state information, which is required for failover, to be accessed by both cluster systems. In the example, the file system is mounted as `/var/mysql`, using the shared disk partition `/dev/sda1`.
- An IP address is associated with the MySQL database to accommodate network access by clients of the database service. This IP address will automatically be migrated among the cluster members as the service fails over. In the example below, the IP address is 10.1.16.12.
- The script that is used to start and stop the MySQL database is the standard System V `init` script, which has been modified with configuration parameters to match the file system on which the database is installed.
- By default, a client connection to a MySQL server will time out after eight hours of inactivity. This connection limit can be modified by setting the `wait_timeout` variable when you start `mysqld`. For example, to set timeouts to 4 hours, start the MySQL daemon as follows:

  ```
  mysqld -O wait_timeout=14400
  ```

  To check if a MySQL server has timed out, invoke the `mysqladmin version` command and examine the uptime. Invoke the query again to automatically reconnect to the server.

  Depending on the Linux distribution, one of the following messages may indicate a MySQL server timeout:

  ```
  CR_SERVER_GONE_ERROR
  CR_SERVER_LOST
  ```

A sample script to start and stop the MySQL database is located in `/usr/share/cluster/doc/services/examples/mysql.server`, and is shown below:

```
#!/bin/sh
# Copyright Abandoned 1996 TCX DataKonsult AB & Monty Program KB & Detron HB
# This file is public domain and comes with NO WARRANTY of any kind
```

```
# Mysql daemon start/stop script.

# Usually this is put in /etc/init.d (at least on machines SYSV R4
# based systems) and linked to /etc/rc3.d/S99mysql. When this is done
# the mysql server will be started when the machine is started.

# Comments to support chkconfig on RedHat Linux
# chkconfig: 2345 90 90
# description: A very fast and reliable SQL database engine.

PATH=/sbin:/usr/sbin:/bin:/usr/bin
basedir=/var/mysql
bindir=/var/mysql/bin
datadir=/var/mysql/var
pid_file=/var/mysql/var/mysqld.pid
mysql_daemon_user=root  # Run mysqld as this user.
export PATH

mode=$1

if test -w /            # determine if we should look at the root config file
then                    # or user config file
  conf=/etc/my.cnf
else
  conf=$HOME/.my.cnf    # Using the users config file
fi

# The following code tries to get the variables safe_mysqld needs from the
# config file.  This isn't perfect as this ignores groups, but it should
# work as the options doesn't conflict with anything else.

if test -f "$conf"      # Extract those fields we need from config file.
then
  if grep "^datadir" $conf > /dev/null
  then
    datadir=`grep "^datadir" $conf | cut -f 2 -d= | tr -d ' '`
  fi
  if grep "^user" $conf > /dev/null
  then
    mysql_daemon_user=`grep "^user" $conf | cut -f 2 -d= | tr -d ' ' | head -1`
  fi
  if grep "^pid-file" $conf > /dev/null
  then
    pid_file=`grep "^pid-file" $conf | cut -f 2 -d= | tr -d ' '`
```

```
  else
    if test -d "$datadir"
    then
      pid_file=$datadir/‘hostname‘.pid
    fi
  fi
  if grep "^basedir" $conf > /dev/null
  then
    basedir=‘grep "^basedir" $conf | cut -f 2 -d= | tr -d ’ ’‘
    bindir=$basedir/bin
  fi
  if grep "^bindir" $conf > /dev/null
  then
    bindir=‘grep "^bindir" $conf | cut -f 2 -d=| tr -d ’ ’‘
  fi
fi


# Safeguard (relative paths, core dumps..)
cd $basedir

case "$mode" in
  ’start’)
    # Start daemon

    if test -x $bindir/safe_mysqld
    then
      # Give extra arguments to mysqld with the my.cnf file. This script may
      # be overwritten at next upgrade.
      $bindir/safe_mysqld –user=$mysql_daemon_user –pid-file=$pid_file –datadir=$datadir &
    else
      echo "Can’t execute $bindir/safe_mysqld"
    fi
    ;;

  ’stop’)
    # Stop daemon. We use a signal here to avoid having to know the
    # root password.
    if test -f "$pid_file"
    then
      mysqld_pid=‘cat $pid_file‘
      echo "Killing mysqld with pid $mysqld_pid"
      kill $mysqld_pid
      # mysqld should remove the pid_file when it exits.
    else
```

```
    echo "No mysqld pid file found. Looked for $pid_file."
   fi
   ;;

 *)
   # usage
   echo "usage: $0 start|stop"
   exit 1
   ;;
esac
```

The following example shows how to use `cluadmin` to add a MySQL service.

```
cluadmin> service add

  The user interface will prompt you for information about the service.
  Not all information is required for all services.

  Enter a question mark (?) at a prompt to obtain help.

  Enter a colon (:) and a single-character command at a prompt to do
  one of the following:

  c - Cancel and return to the top-level cluadmin command
  r - Restart to the initial prompt while keeping previous responses
  p - Proceed with the next prompt

Currently defined services:

  databse1
  apache2
  dbase_home
  mp3_failover

Service name: mysql_1
Preferred member [None]: devel0
Relocate when the preferred member joins the cluster (yes/no/?) [no]: yes
User script (e.g., /usr/foo/script or None) [None]: \
 /etc/rc.d/init.d/mysql.server

Do you want to add an IP address to the service (yes/no/?): yes

        IP Address Information

IP address: 10.1.16.12
Netmask (e.g. 255.255.255.0 or None) [None]: [Return]
```

```
Broadcast (e.g. X.Y.Z.255 or None) [None]: [Return]

Do you want to (a)dd, (m)odify, (d)elete or (s)how an IP address,
 or are you (f)inished adding IP addresses: f

Do you want to add a disk device to the service (yes/no/?): yes

        Disk Device Information

Device special file (e.g., /dev/sda1): /dev/sda1
Filesystem type (e.g., ext2, reiserfs, ext3 or None): ext2
Mount point (e.g., /usr/mnt/service1 or None) [None]: /var/mysql
Mount options (e.g., rw, nosuid): rw
Forced unmount support (yes/no/?) [no]: yes

Do you want to (a)dd, (m)odify, (d)elete or (s)how devices,
  or are you (f)inished adding device information: f

Disable service (yes/no/?) [no]: yes

name: mysql_1
disabled: yes
preferred node: devel0
relocate: yes
user script: /etc/rc.d/init.d/mysql.server
IP address 0: 10.1.16.12
  netmask 0: None
  broadcast 0: None
device 0: /dev/sda1
  mount point, device 0: /var/mysql
  mount fstype, device 0: ext2
  mount options, device 0: rw
  force unmount, device 0: yes

Add mysql_1 service as shown? (yes/no/?) y
Added mysql_1.
cluadmin>
```

5.4 Setting Up a DB2 Service

This section provides an example of setting up a cluster service that will fail over IBM DB2 Enterprise/Workgroup Edition on a cluster. This example assumes that NIS is not running on the cluster systems. To install the software and database on the cluster systems, follow these steps:

1. On both cluster systems, log in as root and add the IP address and host name that will be used to access the DB2 service to `/etc/hosts` file. For example:

```
10.1.16.182     ibmdb2.class.cluster.com        ibmdb2
```

2. Choose an unused partition on a shared disk to use for hosting DB2 administration and instance data, and create a file system on it. For example:

```
# mke2fs /dev/sda3
```

3. Create a mount point on both cluster systems for the file system created in Step 2. For example:

```
# mkdir /db2home
```

4. On the first cluster system, `devel0`, mount the file system created in Step 2 on the mount point created in Step 3. For example:

```
devel0# mount -t ext2 /dev/sda3 /db2home
```

5. On the first cluster system, `devel0`, mount the DB2 cdrom and copy the setup response file included in the distribution to `/root`. For example:

```
devel0% mount -t iso9660 /dev/cdrom /mnt/cdrom
devel0% cp /mnt/cdrom/IBM/DB2/db2server.rsp /root
```

6. Modify the setup response file, `db2server.rsp`, to reflect local configuration settings. Make sure that the UIDs and GIDs are reserved on both cluster systems. For example:

```
-----------Instance Creation Settings------------
-------------------------------------------------
DB2.UID = 2001
DB2.GID = 2001
DB2.HOME_DIRECTORY = /db2home/db2inst1

-----------Fenced User Creation Settings----------
-------------------------------------------------
UDF.UID = 2000
UDF.GID = 2000
UDF.HOME_DIRECTORY = /db2home/db2fenc1

-----------Instance Profile Registry Settings------
-------------------------------------------------
DB2.DB2COMM = TCPIP

----------Administration Server Creation Settings---
-------------------------------------------------
ADMIN.UID = 2002
ADMIN.GID = 2002
```

```
ADMIN.HOME_DIRECTORY = /db2home/db2as

---------Administration Server Profile Registry Settings-
-----------------------------------------------------
ADMIN.DB2COMM = TCPIP

---------Global Profile Registry Settings-------------
-----------------------------------------------------
DB2SYSTEM = ibmdb2
```

7. Start the installation. For example:

```
devel0# cd /mnt/cdrom/IBM/DB2
devel0# ./db2setup -d -r /root/db2server.rsp 1>/dev/null \
     2>/dev/null &
```

8. Check for errors during the installation by examining the installation log file, `/tmp/db2setup.log`. Every step in the installation must be marked as **SUCCESS** at the end of the log file.

9. Stop the DB2 instance and administration server on the first cluster system. For example:

```
devel0# su - db2inst1
devel0# db2stop
devel0# exit
devel0# su - db2as
devel0# db2admin stop
devel0# exit
```

10. Unmount the DB2 instance and administration data partition on the first cluster system. For example:

```
devel0# umount /db2home
```

11. Mount the DB2 instance and administration data partition on the second cluster system, `devel1`. For example:

```
devel1# mount -t ext2 /dev/sda3 /db2home
```

12. Mount the DB2 CDROM on the second cluster system and remotely copy the `db2server.rsp` file to `/root`. For example:

```
devel1# mount -t iso9660 /dev/cdrom /mnt/cdrom
devel1# rcp devel0:/root/db2server.rsp /root
```

13. Start the installation on the second cluster system, `devel1`. For example:

```
devel1# cd /mnt/cdrom/IBM/DB2
devel1# ./db2setup -d -r /root/db2server.rsp 1>/dev/null \
```

```
        2>/dev/null &
```

14. Check for errors during the installation by examining the installation log file. Every step in the installation must be marked as **SUCCESS** except for the following:

```
DB2 Instance Creation                              FAILURE
Update DBM configuration file for TCP/IP           CANCEL
Update parameter DB2COMM                           CANCEL
Auto start DB2 Instance                            CANCEL
DB2 Sample Database                                CANCEL
Start DB2 Instance
Administration Server Creation                     FAILURE
Update parameter DB2COMM                           CANCEL
Start Administration Serve                         CANCEL
```

15. Test the database installation by invoking the following commands, first on one cluster system, and then on the other cluster system:

```
# mount -t ext2 /dev/sda3 /db2home
# su - db2inst1
# db2start
# db2 connect to sample
# db2 select tabname from syscat.tables
# db2 connect reset
# db2stop
# exit
# umount /db2home
```

16. Create the DB2 cluster start/stop script on the DB2 administration and instance data partition. For example:

```
# vi /db2home/ibmdb2
# chmod u+x /db2home/ibmdb2

#!/bin/sh
#
# IBM DB2 Database Cluster Start/Stop Script
#

DB2DIR=/usr/IBMdb2/V6.1

case $1 in
"start")
   $DB2DIR/instance/db2istrt
   ;;
"stop")
   $DB2DIR/instance/db2ishut
```

```
    ;;
esac
```

17. Modify the `/usr/IBMdb2/V6.1/instance/db2ishut` file on both cluster systems to forcefully disconnect active applications before stopping the database. For example:

```
for DB2INST in ${DB2INSTLIST?}; do
  echo "Stopping DB2 Instance "${DB2INST?}"..." >> ${LOGFILE?}
  find_homedir ${DB2INST?}
  INSTHOME="${USERHOME?}"
  su ${DB2INST?} -c " \
    source ${INSTHOME?}/sqllib/db2cshrc   1> /dev/null 2> /dev/null; \
        ${INSTHOME?}/sqllib/db2profile 1> /dev/null 2> /dev/null; \
>>>>>>> db2 force application all; \
    db2stop                    "  1>> ${LOGFILE?} 2>> ${LOGFILE?}
  if [ $? -ne 0 ]; then
    ERRORFOUND=${TRUE?}
  fi
done
```

18. Edit the `inittab` file and comment out the DB2 line to enable the cluster service to handle starting and stopping the DB2 service. This is usually the last line in the file. For example:

```
# db:234:once:/etc/rc.db2 > /dev/console 2>&1 # Autostart DB2 Services
```

Use the `cluadmin` utility to create the DB2 service. Add the IP address from Step 1, the shared partition created in Step 2, and the start/stop script created in Step 16.

To install the DB2 client on a third system, invoke these commands:

```
display# mount -t iso9660 /dev/cdrom /mnt/cdrom
display# cd /mnt/cdrom/IBM/DB2
display# ./db2setup -d -r /root/db2client.rsp
```

To configure a DB2 client, add the service's IP address to the `/etc/hosts` file on the client system:

```
10.1.16.182   ibmdb2.lowell.mclinux.com   ibmdb2
```

Then, add the following entry to the `/etc/services` file on the client system:

```
db2cdb2inst1      50000/tcp
```

Invoke the following commands on the client system:

```
# su - db2inst1
# db2 catalog tcpip node ibmdb2 remote ibmdb2 server db2cdb2inst1
# db2 catalog database sample as db2 at node ibmdb2
# db2 list node directory
# db2 list database directory
```

To test the database from the DB2 client system, invoke the following commands:

```
# db2 connect to db2 user db2inst1 using ibmdb2
# db2 select tabname from syscat.tables
# db2 connect reset
```

6 Network File Sharing Services

This chapter contains instructions for configuring Red Hat Linux Advanced Server to make network file sharing services through NFS and Samba highly available.

6.1 Setting Up an NFS Service

A highly available network filesystem (NFS) are one of the key strengths of the clustering infrastructure. Advantages of clustered NFS services include:

- Ensures that NFS clients maintain uninterrupted access to key data in the event of server failure.
- Facilitates planned maintenance by allowing transparent relocation of NFS services to one cluster member, allowing an administrator to fix or upgrade the other cluster member.
- Allows setup of an active-active configuration to maximize equipment utilization. More details on active-active configurations appear later in this chapter.

6.1.1 NFS Server Requirements

In order to create highly available NFS services, there are a few requirements which must be met by each cluster server. (Note: these requirements do not pertain to NFS client systems.) These requirements are as follows:

- Kernel support for the NFS server must be enabled. NFS can be either configured statically or as a module. Both NFS V2 and NFS V3 are supported.
- The kernel support for NFS provided with Red Hat Linux Advanced Server 2.1 incorporates enhancements (initially developed by Mission Critical Linux Inc.) which allow for transparent relocation of NFS services. These kernel enhancements prevent NFS clients from receiving Stale file handle errors after an NFS service has been relocated. If using kernel sources that do not include these NFS enhancements, then NFS can still be configured and run within the cluster; but warning messages will appear during service start and stop pointing out the absence of these kernel enhancements.
- The NFS daemons must be running on all cluster servers. This is accomplished by enabling the NFS init.d run level script. For example:

```
/sbin/chkconfig --level 345 nfs on
```

- The RPC `portmap` daemon must be enabled. For example:

```
/sbin/chkconfig --level 345 portmap on
```

NFS services will not start unless the following NFS daemons are running: `nfsd`, `rpc.mountd`, and `rpc.statd`.

- Filesystem mounts and their associated exports for clustered NFS services should *not* be included in `/etc/fstab` or `/etc/exports`. Rather, for clustered NFS services, the parameters describing mounts and exports are entered via the `cluadmin` configuration utility.

6.1.2 Gathering NFS Service Configuration Parameters

In preparation of configuring NFS services, it is important to plan how the filesystems will be exported and failed over. The following information is required in order to configure NFS services:

- *Service Name* — A name used to uniquely identify this service within the cluster.
- *Preferred Member* — Defines which system will be the NFS server for this service if more than one cluster member is operational.
- *Relocation Policy* — whether to relocate the service to the preferred member if the preferred member wasn't running at the time the service was initially started. This parameter is useful as a means of load balancing the cluster members as NFS servers by assigning half the load to each.
- *IP Address* — NFS clients access filesystems from an NFS server which is designated by its IP Address (or associated hostname). In order to abstract NFS clients from knowing which specific cluster member is the acting NFS server, the client systems should not use the cluster member's hostname as the IP address by which a service is mounted. Rather, clustered NFS services are assigned floating IP addresses which are distinct from the cluster server's IP addresses. This floating IP address is then configured on whichever cluster member is actively serving the NFS export. Following this approach, the NFS clients are only aware of the floating IP address and are unaware of the fact that clustered NFS server has been deployed. When entering an NFS service's IP address, an administrator will also be prompted to enter an associated netmask and broadcast address. If **None** (which is the default) is selected, then the assigned netmask and broadcast will be the same as what the network interface is currently configured to.
- *Mount Information* — for non-clustered filesystems, the mount information is typically placed in `/etc/fstab`. By contrast, clustered filesystems must not be placed in `/etc/fstab`. This is necessary to ensure that only one cluster member at a time has the filesystem mounted. Failure to do so will result in filesystem corruption and likely system crashes.
  - *Device special file* — The mount information designates the disk's device special file and the directory on which the filesystem will be mounted. In the process of configuring an NFS service, this information will be prompted for.
  - *Mount point directory* — An NFS service can include more than one filesystem mount. In this manner, the filesystems will be grouped together as a single failover unit.

  - *Mount options* — The mount information also designates the mount options. Note: by default, the Linux NFS server does not guarantee that all write operations are synchronously written to disk. In order to ensure synchronous writes, specify the `sync` mount option. Specifying the `sync` mount option favors data integrity at the expense of performance. Refer to `mount`(8) for detailed descriptions of the mount related parameters.
  - *Forced unmount* — As part of the mount information, there will be a prompt as to whether forced unmount should be enabled or not. When forced unmount is enabled, if any applications running on the cluster server have the designated filesystem mounted when the service is being disabled or relocated, then that application will be killed to allow the unmount to proceed.

- *Export Information* — for non-clustered NFS services, export information is typically placed in `/etc/exports`. In contrast, clustered NFS services should *not* place export information in `/etc/exports`; rather there will be a prompt for this information during service configuration. Export information includes:
  - *Export directory* — the export directory can be the same as the mount point specified with the mount information. In this case, the entire filesystem is accessible through NFS. Alternatively, a specified portion (subdirectory) of a mounted filesystem can be mounted instead of the entire filesystem. By exporting subdirectories of a mountpoint, different access rights can be allocated to different sets of NFS clients.
  - *Export client names* — this parameter defines which systems will be allowed to access the file system as NFS clients. Under this method, individual systems can be designated (e.g. fred), as well as wildcards to allow groups of systems (e.g. *.wizzbang.com). Entering a client name of * allows any client to mount the filesystem.
  - *Export client options* — this parameter defines the access rights afforded to the corresponding client(s). Examples include **ro** (read only), and **rw** (read write). Unless explicitly specified otherwise, the default export options are `ro`,`async`,`wdelay`,`root_squash`.

  Refer to `exports`(5) for detailed descriptions of the export parameter syntax.

When running the `cluadmin` utility to configure NFS services:

- Take extra care to correctly enter the service parameters. The validation logic associated with NFS parameters is currently insufficient.
- In response to most of the prompts, you can enter the [?] character to obtain descriptive help text.

6.1.3 Example NFS Service Configuration

In order to illustrate the configuration process for an NFS service, an example configuration is described in this section. This example consists of setting up a single NFS export which houses the home directories of 4 members of an accounting department. NFS client access will be restricted to these four user's systems.

The following are the service configuration parameters which will be used as well as some descriptive commentary.

Note

Prior to configuring an NFS service using `cluadmin`, it is required that the cluster daemons are running.

- *Service Name* — **nfs_accounting**. This name was chosen as a reminder of the service's intended function to provide exports to the members of the accounting department.
- *Preferred Member* — **clu4**. In this example cluster, the member names are clu3 and clu4.
- User Script — The cluster infrastructure includes support for NFS services. Consequently, there is no need to create a User Script when configuring an NFS service. For this reason, when prompted to specify a User Script, the default value of **None** should be selected.
- *IP Address* — **10.0.0.10**. There is a corresponding hostname of clunfsacct associated with this IP address, by which NFS clients mount the filesystem. Note that this IP address is distinct from that of both cluster members (clu3 and clu4). The default netmask and broadcast address will be used.
- *Mount Information* — `/dev/sdb10`, which refers to the partition on the shared storage RAID box on which the file system will be physically stored. **ext3** — referring to the file system type which was specified when the file system was created. `/mnt/users/accounting` — specifies the file system mount point. `rw`,`nosuid`,`sync` — are the mount options.
- *Export Information* - for this example, the entire mounted file system will be made accessible on a read write basis by four members of the accounting department. The names of the systems used by these four members are `burke`, `stevens`, `needle` and `dwalsh`.

The following is an excerpt of the /etc/hosts file used to represent IP addresses and associated hostnames used within the cluster:

```
10.0.0.3     clu3        # cluster member

10.0.0.4     clu4        # second cluster member

10.0.0.10    clunfsacct  # floating IP address associated with accounting dept. NFS service

10.0.0.11    clunfseng   # floating IP address associated with engineering dept. NFS service
```

The following is excerpted from running `cluadmin` to configure this example NFS service:

```
cluadmin> service add
```

```
Service name: nfs_accounting
Preferred member [None]: clu4
Relocate when the preferred member joins the cluster (yes/no/?) \
     [no]: yes
Status check interval [0]: 30
User script (e.g., /usr/foo/script or None) [None]:
Do you want to add an IP address to the service (yes/no/?) [no]: yes

        IP Address Information

IP address: 10.0.0.10
Netmask (e.g. 255.255.255.0 or None) [None]:
Broadcast (e.g. X.Y.Z.255 or None) [None]:
Do you want to (a)dd, (m)odify, (d)elete or (s)how an IP address, or
are you (f)inished adding IP addresses [f]: f
Do you want to add a disk device to the service (yes/no/?) [no]: yes

Disk Device Information

Device special file (e.g., /dev/sdb4): /dev/sdb10
Filesystem type (e.g., ext2, ext3 or None): ext3
Mount point (e.g., /usr/mnt/service1) [None]: /mnt/users/accounting
Mount options (e.g., rw,nosuid,sync): rw,nosuid,sync
Forced unmount support (yes/no/?) [yes]:
Would you like to allow NFS access to this filesystem (yes/no/?)  [no]: yes

You will now be prompted for the NFS export configuration:

Export directory name: /mnt/users/accounting

Authorized NFS clients

Export client name [*]: burke
Export client options [None]: rw
Do you want to (a)dd, (m)odify, (d)elete or (s)how NFS CLIENTS, or
are you (f)inished adding CLIENTS [f]: a

Export client name [*]: stevens
Export client options [None]: rw
Do you want to (a)dd, (m)odify, (d)elete or (s)how NFS CLIENTS, or
are you (f)inished adding CLIENTS [f]: a

Export client name [*]: needle
Export client options [None]: rw
Do you want to (a)dd, (m)odify, (d)elete or (s)how NFS CLIENTS, or
```

```
are you (f)inished adding CLIENTS [f]: a

Export client name [*]: dwalsh
Export client options [None]: rw
Do you want to (a)dd, (m)odify, (d)elete or (s)how NFS CLIENTS, or
are you (f)inished adding CLIENTS [f]: f
Do you want to (a)dd, (m)odify, (d)elete or (s)how NFS EXPORTS, or
are you (f)inished adding EXPORTS [f]:
Do you want to (a)dd, (m)odify, (d)elete or (s)how DEVICES,
or are you (f)inished adding DEVICES [f]:
Disable service (yes/no/?) [no]:
name: nfs_eng
disabled: no
preferred node: clu4
relocate: yes
user script: None
monitor interval: 30
IP address 0: 10.0.0.10
  netmask 0: None
  broadcast 0: None
device 0: /dev/sdb10
  mount point, device 0: /mnt/users/accounting
  mount fstype, device 0: ext3
  mount options, device 0: rw,nosuid,sync
  force unmount, device 0: yes
NFS export 0: /mnt/users/accounting
  Client 0: burke, rw
  Client 1: stevens, rw
  Client 2: needle, rw
  Client 3: dwalsh, rw
Add nfs_eng service as shown? (yes/no/?) yes
Added nfs_eng.
cluadmin>
```

6.1.4 NFS Client Access

The NFS usage model for clients is completely unchanged from its normal approach. Following the prior example, if a client system wishes to mount the highly available NFS service, it simply needs to have an entry like the following in its `/etc/fstab` file:

```
clunfsacct:/mnt/users/accounting  /mnt/users/  nfs bg 0 0
```

6.1.5 Active-Active NFS Configuration

In the previous section, an example configuration of a simple NFS service was discussed. This section describes how to setup a more complex NFS service.

The example in this section involves configuring a pair of highly available NFS services. In this example, suppose two separate teams of users will be accessing NFS filesystems served by the cluster. To serve these users, two separate NFS services will be configured. Each service will have its own separate IP address and be preferred to distinct cluster members. In this manner, under normal operating circumstances, when both cluster members are running, each will be NFS exporting one of the filesystems. This enables an administrator to most effectively utilize the capacity of the two server systems. In the event of a failure (or planned maintenance) on either of the cluster members, both NFS services will be running on the active cluster member.

This example configuration will expand upon the NFS service created in the prior section by adding in a second service. The following service configuration parameters apply to this second service:

- *Service Name* — **nfs_engineering**. This name was chosen as a reminder of the service's intended function to provide NFS exports to the members of the engineering department.
- *Preferred Member* — **clu3**. In this example cluster, the member names are clu3 and clu4. Note that here clu3 is specified because the other cluster service (nfs_accounting) has clu4 specified as its preferred server.
- *IP Address* — **10.0.0.11**. There is a corresponding hostname of clunfseng associated with this IP address, by which NFS clients mount the filesystem. Note that this IP address is distinct from that of both cluster members (clu3 and clu4). Also note that this IP address is different from the one associated with the other NFS service (nfs_accounting). The default netmask and broadcast address will be used.
- *Mount Information* — `/dev/sdb11`, which refers to the partition on the shared storage RAID box on which the filesystem will be physically stored. **ext2** — referring to the filesystem type which was specified when the filesystem was created. `/mnt/users/engineering` — specifies the filesystem mount point. `rw`,`nosuid`,`sync` — are the mount options.
- *Export Information* — for this example, individual subdirectories of the mounted filesystem will be made accessible on a read-write (rw) basis by three members of the engineering department. The names of the systems used by these three team members are `ferris`, `denham`, and `brown`. To make this example more illustrative, notice that each team member will only be able to NFS mount their specific subdirectory.

The following is an example output from running `cluadmin` to create this second NFS service on the same cluster as used in the prior example when the service nfs_accounting was created.

```
cluadmin> service add
```

```
Service name: nfs_engineering
Preferred member [None]: clu3
Relocate when the preferred member joins the cluster (yes/no/?) [no]: yes
Status check interval [0]: 30
User script (e.g., /usr/foo/script or None) [None]:
Do you want to add an IP address to the service (yes/no/?) [no]: yes

        IP Address Information

IP address: 10.0.0.11
Netmask (e.g. 255.255.255.0 or None) [None]:
Broadcast (e.g. X.Y.Z.255 or None) [None]:
Do you want to (a)dd, (m)odify, (d)elete or (s)how an IP address, or
are you (f)inished adding IP addresses [f]: f
Do you want to add a disk device to the service (yes/no/?) [no]: yes

Disk Device Information

Device special file (e.g., /dev/sdb4): /dev/sdb11
Filesystem type (e.g., ext2, ext3 or None): ext2
Mount point (e.g., /usr/mnt/service1) [None]: /mnt/users/engineering
Mount options (e.g., rw,nosuid,sync): rw,nosuid,sync
Forced unmount support (yes/no/?) [yes]:
Would you like to allow NFS access to this filesystem (yes/no/?) \
     [no]: yes

You will now be prompted for the NFS export configuration:

Export directory name: /mnt/users/engineering/ferris

Authorized NFS clients

Export client name [*]: ferris
Export client options [None]: rw
Do you want to (a)dd, (m)odify, (d)elete or (s)how NFS CLIENTS, or
are you (f)inished adding CLIENTS [f]: f
Do you want to (a)dd, (m)odify, (d)elete or (s)how NFS EXPORTS, or
are you (f)inished adding EXPORTS [f]: a

Export directory name: /mnt/users/engineering/denham

Authorized NFS clients

Export client name [*]: denham
Export client options [None]: rw
```

```
Do you want to (a)dd, (m)odify, (d)elete or (s)how NFS CLIENTS, or
are you (f)inished adding CLIENTS [f]:
Do you want to (a)dd, (m)odify, (d)elete or (s)how NFS EXPORTS, or
are you (f)inished adding EXPORTS [f]: a

Export directory name: /mnt/users/engineering/brown

Authorized NFS clients

Export client name [*]: brown
Export client options [None]: rw
Do you want to (a)dd, (m)odify, (d)elete or (s)how NFS CLIENTS, or
are you (f)inished adding CLIENTS [f]: f
Do you want to (a)dd, (m)odify, (d)elete or (s)how NFS EXPORTS, or
are you (f)inished adding EXPORTS [f]: a
Do you want to (a)dd, (m)odify, (d)elete or (s)how DEVICES, or
are you (f)inished adding DEVICES [f]:
Disable service (yes/no/?) [no]:
name: nfs_engineering
disabled: no
preferred node: clu3
relocate: yes
user script: None
monitor interval: 30
IP address 0: 10.0.0.11
  netmask 0: None
  broadcast 0: None
device 0: /dev/sdb11
  mount point, device 0: /mnt/users/engineering
  mount fstype, device 0: ext2
  mount options, device 0: rw,nosuid,sync
  force unmount, device 0: yes
NFS export 0: /mnt/users/engineering/ferris
  Client 0: ferris, rw
NFS export 0: /mnt/users/engineering/denham
  Client 0: denham, rw
NFS export 0: /mnt/users/engineering/brown
  Client 0: brown, rw
Add nfs_engineering service as shown? (yes/no/?) yes
Added nfs_engineering.
cluadmin>
```

6.1.6 NFS Caveats

The following points should be taken into consideration when clustered NFS services are configured.

Avoid using `exportfs -r`

File systems being NFS exported by cluster members do not get specified in the conventional `/etc/exports` file. Rather, the NFS exports associated with cluster services are specified in the cluster configuration file (as established by `cluadmin`).

The command `exportfs -r` removes any exports which are not explicitly specified in the `/etc/exports` file. Running this command will cause the clustered NFS services to become unavailable until the service is restarted. For this reason, it is recommended to avoid using the `exportfs -r` command on a cluster on which highly available NFS services are configured. To recover from unintended usage of `exportfs -r`, the NFS cluster service must be stopped and then restarted.

NFS File Locking

NFS file locks are *not* preserved across a failover or service relocation. This is due to the fact that the Linux NFS implementation stores file locking information in system files. These system files representing NFS locking state are not replicated across the cluster. The implication is that locks may be regranted subsequent to the failover operation.

6.2 Setting Up a High Availability Samba Service

Highly available network file services are one of the key strengths of the clustering infrastructure. Advantages of high availibility Samba services include:

- Heterogeneous file serving capabilities to Microsft® Windows™ clients using the CIFS/SMB protocol.
- Allows the same set of filesystems to be simultaneously network served to both NFS and Windows based clients.
- Ensures that Windows-based clients maintain access to key data, or allowed to quickly reestablish connection in the event of server failure.
- Facilitates planned maintenance by allowing the transparent relocation of Samba services to one cluster member, enabling administrators to fix or upgrade the other cluster member.
- Allows the setup of an active-active configuration to maximize equipment utilization. More details on active-active configurations appear below.

Note

A complete explanation of Samba configuration is beyond the scope of this document. Rather, this documentation highlights aspects which are crucial for clustered operation. Refer to *The Official Red Hat Linux Customization Guide* for more details on Samba configuration. Additionally, refer to the following URL for more information on Samba configuration http://www.redhat.com/support/resources/print_file/samba.html. To configure high availability Samba services, a prerequisite would be to know how to configure conventional non-clustered Samba fileserving.

6.2.1 Samba Server Requirements

If you intend to create highly available Samba services, then there are a few requirements which must be met by each cluster server. These requirements include:

- The Samba RPM packages must be installed. Red Hat Linux Advanced Server ships with the following Samba-related packages: `samba` and `samba-common`. Note that there have been no modifications to the Samba RPMs to support high- availability.
- The Samba daemons will be started and stopped by the cluster infrastructure on a per-service basis. Consequently, the Samba configuration information should *not* be specified in the conventional `/etc/samba/smb.conf`. The automated system startup of the Samba daemons `smbd` and `nmbd` should be disabled in `init.d` run levels. For example: `chkconfig --del smb`.
- Since the cluster infrastructure stops the cluster related Samba daemons appropriately, system administrators should not manually run the conventional samba stop script (e.g. `service smb stop`) as this will terminate all cluster related samba daemons.
- File system mounts for clustered Samba services should not be included in `/etc/fstab`. Rather, for clustered services, the parameters describing mounts are entered via the `cluadmin` configuration utility.
- Failover of samba printer shares is not currently supported.

6.2.2 Samba Operating Model

This section provides background information describing the implementation model in support of Samba high availability services. Knowledge of this information will provide the context for understanding the configuration requirements of clustered Samba services.

The conventional, non-clustered Samba configuration model consists of editing the `/etc/samba/smb.conf` file to designate which filesystems are to be made network accessible to

the specified Windows clients. It also designates access permissions and other mapping capabilities. In the single system model, a single instance of each of the `smbd` and `nmbd` daemons are automatically started up by the `/etc/rc.d/init.d/smb` runlevel script.

In order to implement high availibility Samba services, rather than having a single `/etc/samba/smb.conf` file; there is an individual per-service samba configuration file. These files are called `/etc/samba/smb.conf.`*`sharename`*; where *sharename* is the specific name of the individual configuration file associated with a Samba service. For example, one share could be called *eng* and another share *acct*, the corresponding Samba configuration files would be `/etc/samba/smb.conf.`*`eng`* and `/etc/samba/smb.conf.`*`acct`*, respectively.

The format of the `smb.conf.`*`sharename`* file is identical to the conventional `smb.conf` format. No additional fields have been created for clustered operation. There are several fields within the `smb.conf.`*`sharename`* file which are required for correct cluster operation; these fields will be described in an upcoming section. When a new Samba service is created using the `cluadmin` utility, a default template `smb.conf.`*`sharename`* file will be created based on the service specific parameters. This file should be used as a starting point from which the system administrator should then adjust to add in the appropriate Windows client systems, specific directories to share as well as permissions.

The system administrator is required to copy the `/etc/samba/smb.conf.`*`sharename`* files onto both cluster members. After the initial configuration time, should any changes be made to any `smb.conf.`*`sharename`* file, it is necessary to also copy this updated version to the other cluster member.

To facilitate high-availability Samba functionality, each individual Samba service configured within the cluster (via `cluadmin`) will have its own individual pair of `smbd`/`nmbd` daemons. Consequently, if there are more than one Samba services configured with the cluster, you may see multiple instances of these daemon pairs running on an individual cluster server. These Samba daemons `smbd`/`nmbd` are not initiated via the conventional `init.d` run level scripts; rather they are initiated by the cluster infrastructure based on whichever node is the active service provider.

In order to allow a single system to run multiple instances of the Samba daemons, each pair of daemons is required to have its own locking directory. Consequently, there will be a separate per-service Samba daemon locking directory. This directory is given the name `/var/cache/samba/`*`sharename`*; where *sharename* is replaced by the Samba share name specified within the service configuration information (via `cluadmin`). Following the prior example, the corresponding lock directories would be `/var/cache/samba/`*`eng`* and `/var/cache/samba/`*`acct`*.

When the `cluadmin` utility is used to configure a Samba service, the `/var/cache/samba/`*`sharename`* directory will be automatically created on the system on which the `cluadmin` utility is running. At this time a reminder will be displayed that you need to manually create this lock directory on the other cluster member. For example: `mkdir /var/cache/samba/`*`eng`*

6.2.3 Gathering Samba Service Configuration Parameters

When preparing to configure Samba services, determine configuration information such as which filesystems will be presented as shares to Windows based clients. The following information is required in order to configure NFS services:

- *Service Name* — A name used to uniquely identify this service within the cluster.
- *Preferred Member* — Defines which system will be the Samba server for this service when more than one cluster member is operational.
- *Relocation Policy* — whether to relocate the service to the preferred member if the preferred member was not running at the time the service was initially started. This parameter is useful as a means of load balancing the cluster members as Samba servers by assigning half the load to each.
- *Status Check Interval* — specifies how often (in seconds) the cluster subsystem should verify that the pair of Samba daemons `smbd`/`nmbd` which are associated with this service are running. In the event that either of these daemons have unexpectedly exited, they will be automatically restarted to resume services. If a value of 0 is specified, then no monitoring will be performed. For example, designating an interval of 90 seconds will result in monitoring at that interval.
- *IP Address* — Windows clients access file shares from a server as designated by its IP Address (or associated hostname). In order to abstract Windows clients from knowing which specific cluster member is the acting Samba server, the client systems should not use the cluster member's hostname as the IP address by which a service is accessed. Rather, clustered Samba services are assigned floating IP addresses which are distinct from the cluster server's IP addresses. This floating IP address is then configured on which ever cluster member is actively serving the share. Following this approach, the Windows clients are only aware of the floating IP address and are unaware of the fact that clustered Samba services have been deployed. When you enter a Samba service's IP address, you will also be prompted to enter an associated netmask and broadcast address. If you select the default of None, then the assigned netmask and broadcast will be the same as what the network interface is currently configured to.
- *Mount Information* — for non-clustered filesystems, the mount information is typically placed in `/etc/fstab`. In contrast, clustered filesystems must not be placed in `/etc/fstab`. This is necessary to ensure that only one cluster member at a time has the filesystem mounted. Failure to do so will result in filesystem corruption and likely system crashes.
  - *Device special file* — The mount information designates the disk's device special file and the directory on which the filesystem will be mounted. In the process of configuring a Samba service you will be prompted for this information.
  - *Mount point directory* — A Samba service can include more than one filesystem mount. In this manner, the filesystems will be grouped together as a single failover unit.
  - *Mount options* — The mount information also designates the mount options.

  - *Forced unmount* — As part of the mount information, you will be prompted as to whether forced unmount should be enabled or not. When forced unmount is enabled, if any applications running on the cluster server have the designated filesystem mounted when the service is being disabled or relocated, then that application will be killed off to allow the unmount to proceed.

- *Export Information* — this information is required for NFS services only. If you are only performing file serving to Windows based clients, answer no when prompted regarding NFS exports. Alternatively, you can configure a service to perform heterogeneous file serving by designating both NFS exports parameters and the Samba share parameter.

- *Samba Share Name* — In the process of configuring a service you will be asked if you wish to share the filesystem to Windows clients. If you answer yes to this question, you will then be prompted for the Samba share name. Based on the name you specify here, there will be a corresponding `/etc/samba/smb.conf.`*`sharename`* file and lock directory `/var/cache/samba/`*`sharename`*. By convention the actual Windows share name specified within the `smb.conf.`*`sharename`* will be set in accordance with this parameter. In practice, you can designate more than one Samba share within an individual `smb.conf.`*`sharename`* file. There can be at most 1 samba configuration specified per service; which must be specified with the first device. For example, if you have multiple disk devices (and corresponding file system mounts) within a single service, then specify a single *sharename* for the service. Then within the `/etc/samab/smb.conf.`*`sharename`* file, designate multiple individual samba shares to share directories from the multiple devices. To disable samba sharing of a service, the share name should be set to **None**.

When running the `cluadmin` utility to configure Samba services:

- Please take care that you correctly enter the service parameters. The validation logic associated with Samba parameters is currently not very robust.
- In response to most of the prompts, you can enter the [?] character to obtain descriptive help text.
- After configuring a Samba service via `cluadmin`, remember to tune the `/etc/samba/smb.conf.`*`sharename`* file for each service in accordance with the clients and authorization scheme you desire.
- Remember to copy the `smb.conf.`*`sharename`* file over to the other cluster member.
- Perform the recommended step to create the Samba daemon's lock directory on the other cluster member, for example: `mkdir /var/cache/samba/`*`acct`*.
- If you delete a Samba service, be sure to manually remove the `/etc/samba/smb.conf/`*`sharename`* file. The `cluadmin` utility does not automatically delete this file in order to preserve your site specific configuration parameters for possible later usage.

6.2.4 Example Samba Service Configuration

In order to illustrate the configuration process for a Samba service, an example configuration is described in this section. This example consists of setting up a single Samba share which houses the home directories of four members of the accounting team. The accounting team will then access this share from their Windows based systems.

The following are the service configuration parameters which will be used as well as some descriptive commentary.

- *Service Name* — `samba_acct`. This name was chosen as a reminder of the service's intended function to provide exports to the members of the accounting team.
- *Preferred Member* — **clu4**. In this example cluster, the member names are clu3 and clu4.
- User Script — The cluster infrastructure includes support for Samba services. Consequently, there is no need to create a User Script when configuring a Samba service. For this reason, when prompted to specify a User Script, the default value of **None** should be selected.
- *Monitoring Interval* — **90** seconds.
- *IP Address* — **10.0.0.10**. There is a corresponding hostname of cluacct associated with this IP address, by which Windows based clients access the share. Note that this IP address is distinct from that of both cluster members (clu3 and clu4). The default netmask and broadcast address will be used.
- *Mount Information* — `/dev/sdb10` refers to the partition on the shared storage RAID box on which the filesystem will be physically stored. **ext2** refers to the filesystem type which was specified when the filesystem was created. `/mnt/users/accounting` specifies the filesystem mount point. `rw,nosuid,sync` are the mount options.
- *Export Information* — for simplicity in this example, the filesystem is not being NFS exported.
- *Share Name* — `acct`. This is the share name by which Windows based clients will access this Samba share, e.g. `\\10.0.0.10\acct`.

The following is an excerpt of the `/etc/hosts` file used to represent IP addresses and associated hostnames used within the cluster:

```
10.0.0.3      clu3      # cluster member
10.0.0.4      clu4      # second cluster member
10.0.0.10     cluacct   # floating IP address associated with accounting team NFS service
```

The following is an excerpt from running `cluadmin` to configure this example Samba service:

```
Service name: samba_acct
Preferred member [None]: clu4
Relocate when the preferred member joins the cluster (yes/no/?) [no]: yes
User script (e.g., /usr/foo/script or None) [None]:
Status check interval [0]: 90
Do you want to add an IP address to the service (yes/no/?) [no]: yes

        IP Address Information

IP address: 10.0.0.10
Netmask (e.g. 255.255.255.0 or None) [None]:
Broadcast (e.g. X.Y.Z.255 or None) [None]:
Do you want to (a)dd, (m)odify, (d)elete or (s)how an IP address, or
are you (f)inished adding IP addresses [f]:
Do you want to add a disk device to the service (yes/no/?) [no]: yes

Disk Device Information

Device special file (e.g., /dev/sdb4): /dev/sdb12
Filesystem type (e.g., ext2, ext3 or None): ext2
Mount point (e.g., /usr/mnt/service1) [None]: /mnt/users/accounting
Mount options (e.g., rw,nosuid,sync): rw,nosuid,sync
Forced unmount support (yes/no/?) [yes]:
Would you like to allow NFS access to this filesystem (yes/no/?)\
     [no]: no
Would you like to share to Windows clients (yes/no/?)  [no]: yes

You will now be prompted for the Samba configuration:
Samba share name: acct

The samba config file /etc/samba/smb.conf.acct does not exist.

Would you like a default config file created (yes/no/?)  [no]: yes

Successfully created daemon lock directory /var/cache/samba/acct.
Please run ‘mkdir /var/cache/samba/acct‘ on the other cluster member.

Successfully created /etc/samba/smb.conf.acct.
Please remember to make necessary customizations and then copy the file
over to the other cluster member.

Do you want to (a)dd, (m)odify, (d)elete or (s)how DEVICES, or
are you (f)inished adding DEVICES [f]: f
name: samba_acct
preferred node: clu4
```

```
relocate: yes
user script: None
monitor interval: 90
IP address 0: 10.0.0.10
  netmask 0: None
  broadcast 0: None
device 0: /dev/sdb12
  mount point, device 0: /mnt/users/accounting
  mount fstype, device 0: ext2
  mount options, device 0: rw,nosuid,sync
  force unmount, device 0: yes
  samba share, device 0: acct
Add samba_acct service as shown? (yes/no/?) yes
```

After running cluadmin as shown above to configure the service, remember to:

- Customize `/etc/samba/smb.conf.`*`sharename`* accordingly.
- Copy `/etc/samba/smb.conf.`*`sharename`* over to the other cluster member.
- Create the suggested lock directory on the other cluster member, e.g. `mkdir /var/cache/samba/`*`acct`*

6.2.5 `smb.conf.`*`sharename`* File Fields

This section describes the fields within the `smb.conf.`*`sharename`* file which are most relevant to the correct operation of highly available Samba services. It is beyond the scope of this document to completely describe all of the fields within a Samba configuration file. There have been no additional field names added in support of clustering, and the file format follows the normal Samba conventions.

Shown below is an example `smb.conf.`*`sharename`* file which was automatically generated by `cluadmin` in response to the service specific parameters. This example file matches the above `cluadmin` service configuration example.

```
# Template samba service configuration file - please modify to specify
# subdirectories and client access permissions.
# Remember to copy this file over to other cluster member, and create
# the daemon lock directory /var/cache/samba/acct.
#
# From a cluster perspective, the key fields are:
# lock directory - must be unique per samba service.
# bind interfaces only - must be present set to yes.
# interfaces - must be set to service floating IP address.
# path - must be the service mountpoint or subdirectory thereof.
# Refer to the cluster documentation for details.

[global]
```

```
        workgroup = RHCLUSTER
        lock directory = /var/cache/samba/acct
        log file = /var/log/samba/%m.log
        encrypt passwords = yes
        bind interfaces only = yes
        interfaces = 10.0.0.10

[acct]
        comment = High Availability Samba Service
        browsable = yes
        writable = no
        public = yes
        path = /mnt/service12
```

The following are descriptions of the most relevant fields, from a clustering perspective, in the `/etc/samba/smb.conf.`*`sharename`* file. In this example, the file is named `/etc/samba/smb.conf.`*`acct`* in accordance with the share name being specified as *acct* while running `cluadmin`. Only the cluster specific fields are described below. The remaining fields follow standard Samba convention and should be tailored accordingly.

Global Parameters

These parameters pertain to all shares which are specified in the `smb.conf.`*`sharename`* file. Note that it is possible to designate more than one share within this file, provided that the directories described within it are within the service's filesystem mounts.

lock directory

Dictates the name of the directory in which the Samba daemons `smbd`/`nmbd` will place their locking files. This must be set to `/var/cache/samba/`*`sharename`*, where *sharename* varies based on the parameter specified in `cluadmin`. Specification of a lock directory is required in order to allow a separate per-service instance of `smbd`/`nmbd`.

bind interfaces only

This parameter must be set to **yes** in order to allow each `smbd`/`nmbd` pair to bind to the floating IP address associated with this clustered Samba service.

interfaces

Specifies the IP address associated with the Samba service. If a netmask is specified within the service, this field would appear like the following example: *interfaces = 10.0.0.10/255.255.254.0*

Share specific parameters

These parameters pertain to a specific Samba share.

writable

By default, the share access permissions are conservatively set as non-writable. Tune this parameter according to your site-specific preferences.

path

Defaults to the first filesystem mount point specified within the service configuration. This should be adjusted to match the specific directory or subdirectory intended to be available as a share to Windows clients.

6.2.6 Windows Client Access to Samba Shares

Windows clients see no discernible difference when accessing shares that are being served by a high availability cluster. From the Windows client's perspective the only requirement is that they access the Samba share via its floating IP address (or associated hostname) which was configured using `cluadmin`, e.g. 10.0.0.10. The Windows clients should not directly access the share from either of the cluster member system's IP address (e.g. clu3 or clu4).

Depending upon the authorization scheme to be utilized in the cluster environment, the `smbpasswd` command may have to be used to establish Windows account information on the cluster servers. When establishing these accounts, it is required that the same Samba related account information be setup on both cluster members. This can be accomplished either by running `smbpassword` similarly on both cluster members, or by running the command on one system and then copying over the resulting `/etc/samba/smbpasswd` file to the other system. For example, to enable a Windows client system named `sarge` to access a Samba share served by the cluster members, run the following command on both cluster members, taking care to specify the same username and password each time:

```
smbpasswd -a sarge
```

On a Windows client, the Samba share can then be accessed in the conventional manner. For example, it is possible to click on the **Start** button on the main taskbar, followed by selecting **Run**. This brings up a dialog box where the clustered Samba share name can be specified. For example: `\\10.0.0.10\acct` or `\\cluacct\acct`. To access the samba share from a Windows client it is also possible to use the **Map Network Drive** feature. Ensure that the hostname portion of the share name refers to the floating service IP address. Following the hostname / IP addresses from the above `/etc/hosts` excerpt; the correct name to refer to this highly available cluster share is `\\cluacct\acct`. The share should not be accessed by referring to the name of the cluster server itself. For example, do not access this share as either `\\clu3\acct` or `\\clu4\acct`. If a share is incorrectly referred to by the cluster server name (e.g. `\\clu3\acct`), then the Windows client will only be able to access the share while it is being actively served by `clu3`, thereby negating any high availability benefits.

Unlike the NFS protocol, the Windows based CIFS/SMB protocol is more stateful. As a consequence, in the Windows environment, it is the responsibility of the individual application to take appropriate

measures to respond to the lack of immediate response from the Samba server. In the case of a planned service relocation or a true failover scenario, there is a period of time where the Windows clients will not get immediate response from the Samba server. Robust Windows applications will retry requests which timeout during this interval.

Well-behaved applications will correctly retry for a service response, resulting in Windows clients being completely unaware of service relocations or failover operations. In contrast, poorly behaved Windows applications will result in error messages in the event of a failover or relocation indicating the inability to access the share. It may be necessary to retry the operation or restart the application in order to enable Windows client systems to reattach to a Samba share for applications that do not correctly behave during failover or service relocation.

The behavior of a Windows based client in response to either failover or relocation of a samba service also varies on which release of windows is installed on each client system. For example, Windows 98 based systems often encounter errors such as, "The network path was not found". Whereas, later versions such as Windows 2000 transparently recover under the same set of circumstances.

7 Apache Services

This chapter contains instructions for configuring Red Hat Linux Advanced Server to make the Apache Web server highly available.

7.1 Setting Up an Apache Service

This section provides an example of setting up a cluster service that will fail over an Apache Web server. Although the actual variables used in the service depend on the specific configuration, the example may assist in setting up a service for a particular environment.

To set up an Apache service, you must configure both cluster systems as Apache servers. The cluster software ensures that only one cluster system runs the Apache software at one time. The Apache configuration will consist of installing the apache RPM packages on both cluster members and configuring a shared filesystem to house the web site's content.

When installing the Apache software on the cluster systems, do not configure the cluster systems so that Apache automatically starts when the system boots by performing the following command: `chkconfig --del httpd`. Rather than having the system startup scripts spawn httpd, the cluster infrastructure will do that on the active cluster server for the Apache service. This will ensure that the corresponding IP address and filesystem mounts are active on only one cluster member at a time.

When adding an Apache service, a "floating" IP address must be assigned to it. The cluster infrastructure binds this IP address to the network interface on the cluster system that is currently running the Apache service. This IP address ensures that the cluster system running the Apache software is transparent to the HTTP clients accessing the Apache server.

The file systems that contain the Web content must not be automatically mounted on shared disk storage when the cluster systems boot. Instead, the cluster software must mount and unmount the file systems as the Apache service is started and stopped on the cluster systems. This prevents both cluster systems from accessing the same data simultaneously, which may result in data corruption. Therefore, do not include the file systems in the `/etc/fstab` file.

Setting up an Apache service involves the following four steps:

1. Set up the shared file system for the service. This filesystem is used to house the website's content.
2. Install the Apache software on both cluster systems.
3. Configure the Apache software on both cluster systems.
4. Add the service to the cluster database.

To set up the shared file systems for the Apache service, perform the following tasks as root user on one cluster system:

1. On a shared disk, use the interactive fdisk utility to create a partition that will be used for the Apache document root directory. Note that it is possible to create multiple document root directories on different disk partitions. See *Partitioning Disks* in Section 2.4.4 for more information.
2. Use the `mkfs` command to create an ext2 file system on the partition you created in the previous step. Specify the drive letter and the partition number. For example:

   ```
   mkfs /dev/sde3
   ```

3. Mount the file system that will contain the Web content on the Apache document root directory. For example:

   ```
   mount /dev/sde3 /var/www/html
   ```

   Do not add this mount information to the `/etc/fstab` file, because only the cluster software can mount and unmount file systems used in a service.
4. Copy all the required files to the document root directory.
5. If you have CGI files or other files that must be in different directories or is separate partitions, repeat these steps, as needed.

Apache must be installed on both cluster systems. Note that the basic Apache server configuration must be the same on both cluster systems in order for the service to fail over correctly. The following example shows a basic Apache Web server installation, with no third-party modules or performance tuning. To install Apache with modules, or to tune it for better performance, see the Apache documentation that is located in the Apache installation directory, or on the Apache Web site, http://httpd.apache.org/docs-project/.

On both cluster systems, install the Apache RPMs. For example:

```
rpm -Uvh apache-1.3.20-16.i386.rpm
```

To configure the cluster systems as Apache servers, customize the `httpd.conf` Apache configuration file, and create a script that will start and stop the Apache service. Then, copy the files to the other cluster system. The files must be identical on both cluster systems in order for the Apache service to fail over correctly.

On one system, perform the following tasks:

1. Edit the `/etc/httpd/conf/httpd.conf` Apache configuration file and customize the file according to your configuration. For example:
   - Specify the directory that will contain the HTML files. You will specify this mount point when adding the Apache service to the cluster database. It is only required to change this field if the mountpoint for the web site's content differs from the default setting of `/var/www/html`. For example:

     ```
     DocumentRoot "/mnt/apacheservice/html"
     ```

- If the script directory resides in a non-standard location, specify the directory that will contain the CGI programs. For example:

```
ScriptAlias /cgi-bin/ "/mnt/apacheservice/cgi-bin/"
```

- Specify the path that was used in the previous step, and set the access permissions to default to that directory. For example:

```
<Directory mnt/apacheservice/cgi-bin">
AllowOverride None
Options None
Order allow,deny
Allow from all
</Directory>
```

  Additional changes may need to be made in order to tune Apache or add third-party module functionality,. For information on setting up other options, refer to the Apache project documentation on the Apache Web site, http://httpd.apache.org/docs-project/.

2. The standard Apache start script, `/etc/rc.d/init.d/httpd` will also be used within the cluster framework to start and stop the Apache server on the active cluster member. Accordingly, when configuring the service, specify that script when prompted for the **User script**.

Note

Depending on the release version, the default Apache service script `/etc/rc.d/init.d/httpd` may not correctly return the status of the httpd daemons in response to `service httpd status` commands. This precludes correct operation of the cluster's monitoring of this service (if optionally designating a monitoring interval when configuring the service). The status clause of this script may appear as follows:

```
status)
       status $httpd
       ;;
```

If this is the case, in order to make service monitoring of Apache work properly, add an additional line as follows to the status clause:

```
status)
       status $httpd
       RETVAL=$?
       ;;
```

Before the Apache service is added to the cluster database, ensure that the Apache directories are not mounted. Then, on one cluster system, add the service. Specify an IP address, which the cluster infrastructure will bind to the network interface on the cluster system that runs the Apache service.

The following is an example of using the `cluadmin` utility to add an Apache service.

```
cluadmin> service add apache

  The user interface will prompt you for information about the service.
  Not all information is required for all services.

  Enter a question mark (?) at a prompt to obtain help.

  Enter a colon (:) and a single-character command at a prompt to do
  one of the following:

  c - Cancel and return to the top-level cluadmin command
  r - Restart to the initial prompt while keeping previous responses
  p - Proceed with the next prompt

Preferred member [None]: devel0
Relocate when the preferred member joins the cluster (yes/no/?) \
 [no]: yes
User script (e.g., /usr/foo/script or None) [None]: \
 /etc/rc.d/init.d/httpd

Do you want to add an IP address to the service (yes/no/?): yes

        IP Address Information

IP address: 10.1.16.150
Netmask (e.g. 255.255.255.0 or None) [None]: 255.255.255.0
Broadcast (e.g. X.Y.Z.255 or None) [None]: 10.1.16.255

Do you want to (a)dd, (m)odify, (d)elete or (s)how an IP address,
 or are you (f)inished adding IP addresses: f

Do you want to add a disk device to the service (yes/no/?): yes

        Disk Device Information

Device special file (e.g., /dev/sda1): /dev/sdb3
Filesystem type (e.g., ext2, reiserfs, ext3 or None): ext3
Mount point (e.g., /usr/mnt/service1 or None) [None]: /var/www/html
Mount options (e.g., rw, nosuid): rw
Forced unmount support (yes/no/?) [no]: yes
```

```
Do you want to (a)dd, (m)odify, (d)elete or (s)how devices,
  or are you (f)inished adding device information: f

Disable service (yes/no/?) [no]: no

name: apache
disabled: no
preferred node: node1
relocate: yes
user script: /etc/rc.d/init/httpd
IP address 0: 10.1.16.150
 netmask 0: 255.255.255.0
 broadcast 0: 10.1.16.255
device 0: /dev/sde3
 mount point, device 0: /var/www/html
 mount fstype, device 0: ext3
 mount options, device 0: rw,sync
 force unmount, device 0: yes
 owner, device 0: nobody
 group, device 0: nobody
Add apache service as shown? (yes/no/?) y

Added apache.
cluadmin>
```

Note

The Red Hat Cluster Manager GUI can not be used on clusters where high-availability Apache services are configured. Refer to Chapter 9, *Configuring and using the Red Hat Cluster Manager GUI* for more information.

---

8 Cluster Administration

The following chapter describes the various administrative tasks involved in maintaining a cluster after it has been installed and configured.

8.1 Displaying Cluster and Service Status

Monitoring cluster and service status can help identify and resolve problems in the cluster environment. The following tools assist in displaying cluster status:

- The `clustat` command
- Log file messages
- The cluster monitoring GUI

Note that status is always from the point of view of the cluster system on which an administrator is running a tool. To obtain comprehensive cluster status, run a tool on all cluster systems.

Cluster and service status includes the following information:

- Cluster member system status
- Power switch status
- Heartbeat channel status
- Service status and which cluster system is running the service or owns the service
- Service monitoring status of the cluster system

The following tables describe how to analyze the status information shown by the `clustat` command and the cluster GUI.

Table 8–1 Member Status

Member Status	Description
UP	The member system is communicating with the other member system and accessing the quorum partitions.
DOWN	The member system is unable to communicate with the other member system.

Table 8–2 Power Switch Status

Power Switch Status	Description
OK	The power switch is operating properly.
Wrn	Could not obtain power switch status.
Err	A failure or error has occurred.
Good	The power switch is operating properly.
Unknown	The other cluster member is **DOWN**.
Timeout	The power switch is not responding to power daemon commands, possibly because of a disconnected serial cable.
Error	A failure or error has occurred.
None	The cluster configuration does not include power switches.
Initializing	The switch is in the process of being initialized and its definitive status has not been concluded.

Table 8–3 Heartbeat Channel Status

Heartbeat Channel Status	Description
OK	The heartbeat channel is operating properly.
Wrn	Could not obtain channel status.
Err	A failure or error has occurred.
ONLINE	The heartbeat channel is operating properly.
OFFLINE	The other cluster member appears to be **UP**, but it is not responding to heartbeat requests on this channel.
UNKNOWN	Could not obtain the status of the other cluster member system over this channel, possibly because the system is **DOWN** or the cluster daemons are not running.

Table 8–4 Service Status

Service Status	Description
running	The service resources are configured and available on the cluster system that owns the service. The **running** state is a persistent state. From this state, a service can enter the **stopping** state (for example, if the preferred member rejoins the cluster)
disabled	The service has been disabled, and does not have an assigned owner. The **disabled** state is a persistent state. From this state, the service can enter the **starting** state (if a user initiates a request to start the service).
starting	The service is in the process of being started. The **starting** state is a transient state. The service remains in the **starting** state until the service start succeeds or fails. From this state, the service can enter the **running** state (if the service start succeeds), the **stopped** state (if the service stop fails), or the **error** state (if the status of the service resources cannot be determined).
stopping	The service is in the process of being stopped. The **stopping** state is a transient state. The service remains in the **stopping** state until the service stop succeeds or fails. From this state, the service can enter the **stopped** state (if the service stop succeeds), the **running** state (if the service stop failed and the service can be started).
stopped	The service is not running on any cluster system, does not have an assigned owner, and does not have any resources configured on a cluster system. The **stopped** state is a persistent state. From this state, the service can enter the **disabled** state (if a user initiates a request to disable the service), or the **starting** state (if the preferred member joins the cluster).

To display a snapshot of the current cluster status, invoke the `clustat` utility. For example:

```
clustat
Cluster Status Monitor (Fileserver Test Cluster)
07:46:05
Cluster alias: clu1alias.boston.redhat.com

====================== M e m b e r   S t a t u s ======================
  Member         Status     Node Id    Power Switch
  -------------- ---------- ---------- ------------
  clu1           Up         0          Good
```

```
  clu2           Up         1          Good

================== H e a r t b e a t   S t a t u s ==================
  Name                           Type       Status
  ------------------------------ ---------- ------------
  clu1         <--> clu2         network    ONLINE

================== S e r v i c e   S t a t u s ======================
                                       Last             Monitor
Restart
Service          Status   Owner          Transition       Interval Count

  ------------- -------- ------------- ---------------- ------------
  nfs1           started  clu1          16:07:42 Feb 27  15       0
  nfs2           started  clu2          00:03:52 Feb 28  2        0
  nfs3           started  clu1          07:43:54 Feb 28  90       0
```

To monitor the cluster and display status at specific time intervals, invoke `clustat` with the `-i` *`time`* command-line option, where *time* specifies the number of seconds between status snapshots.

8.2 Starting and Stopping the Cluster Software

Start the cluster software on a cluster system by invoking the `cluster start` command located in the System V `/etc/rc.d/init` directory. For example:

```
/sbin/service cluster start
```

Stop the cluster software on a cluster system by invoking the `cluster stop` command located in the System V `/etc/rc.d/init` directory. For example:

```
 /sbin/service cluster stop
```

The previous command will cause the cluster system's services to "failover" to the other cluster system.

8.3 Removing a Cluster Member

It may become necessary to temporarily remove a member system from the cluster. For example, if a cluster system experiences a hardware failure, that system will have to be rebooted but prevented from rejoining the cluster in order to perform maintenance on the system.

Use the `/sbin/chkconfig` utility to be able to boot a cluster system, without allowing it to rejoin the cluster. For example:

```
/sbin/chkconfig --del cluster
```

When the system is able to rejoin the cluster, use the following command:

```
/sbin/chkconfig --add cluster
```

Then reboot the system or run the `cluster start` command located in the System V `init` directory. For example:

```
/sbin/service cluster start
```

8.4 Modifying the Cluster Configuration

It may be necessary at some point to modify the cluster configuration. For example, it may be necessary to correct heartbeat channel or quorum partition entries in the cluster database, a copy of which is located in the `/etc/cluster.conf` file.

Use the cluconfig and `cluadmin` utility to modify the cluster configuration. Do not modify the `cluster.conf` file manually. To modify the cluster configuration, stop the cluster software on one cluster system, as described in Section 8.2, *Starting and Stopping the Cluster Software*.

Then, invoke the cluconfig utility, and specify the correct information at the prompts. After running the utility, restart the cluster software.

8.5 Backing Up and Restoring the Cluster Database

It is recommended to regularly back up the cluster database, especially before making any significant changes to the cluster configuration.

To back up the cluster database to the `/etc/cluster.conf.bak` file, invoke the `cluadmin` utility, and specify the `cluster backup` command. For example:

```
cluadmin> cluster backup
```

You can also save the cluster database to a different file by invoking the `cluadmin` utility and specifying the `cluster saveas` *`filename`* command.

To restore the cluster database, follow these steps:

1. Stop the cluster software on one system by invoking the `cluster stop` command located in the System V `init` directory. For example:

   ```
   /sbin/service cluster stop
   ```

   The previous command will cause the cluster system's services to fail over to the other cluster system.

2. On the remaining cluster system, invoke the `cluadmin` utility and restore the cluster database. To restore the database from the `/etc/cluster.conf.bak` file, specify the `cluster restore` command. To restore the database from a different file, specify the `cluster restorefrom` *`file_name`* command.

   The cluster will disable all running services, delete all the services, and then restore the database.

3. Restart the cluster software on the stopped system by invoking the `cluster start` command located in the System V `init` directory. For example:

   ```
   /sbin/service cluster start
   ```

4. Restart each cluster service by invoking the `cluadmin` utility on the cluster system on which you want to run the service and specifying the `service enable` *`service_name`* command.

8.6 Modifying Cluster Event Logging

It is possible to modify the severity level of the events that are logged by the `clupowerd`, `cluquorumd`, `cluhbd`, and `clusvcmgrd` daemons. This is done so that the daemons on the cluster systems will log messages at the same level.

To change a cluster daemon's logging level on all the cluster systems, invoke the `cluadmin` utility, and specify the `cluster loglevel` command, the name of the daemon, and the severity level. Specify the severity level by using the name or the number that corresponds to the severity level. The values 0 to 7 refer to the following severity levels:

```
0 — emerg
1 — alert
2 — crit
3 — err
4 — warning
5 — notice
6 — info
7 — debug
```

Note that the cluster logs messages with the designated severity level and also messages of a higher severity. For example, if the severity level for quorum daemon messages is 2 (**crit**), then the cluster logs messages for **crit**, **alert**, and **emerg** severity levels. Note that setting the logging level to a low severity level, such as 7 (**debug**), will result in large log files over time.

The following example enables the cluquorumd daemon to log messages of all severity levels:

```
 cluadmin
cluadmin> cluster loglevel cluquorumd 7
cluadmin>
```

8.7 Updating the Cluster Software

Before upgrading Red Hat Cluster Manager, be sure to install all of the required software, as described in Section 2.3.1, *Kernel Requirements*. The cluster software can be updated while preserving the existing cluster database. Updating the cluster software on a system can take from 10 to 20 minutes.

To update the cluster software while minimizing service downtime, follow these steps:

1. On a cluster system in need of an update, run the `cluadmin` utility and back up the current cluster database. This will preserve the existing cluster configuration database. For example, at the `cluadmin>` prompt, perform the following command:

   ```
   cluster backup
   ```

2. Stop the cluster software on the first cluster system to be updated by invoking the `cluster stop` command located in the System V `init` directory. For example:

   ```
   /sbin/service cluster stop
   ```

3. Install the latest cluster software on the first cluster system to be updated by invoking. However, when prompted by the cluconfig utility whether to use the existing cluster database, specify **`yes`**.

4. Stop the cluster software on the second cluster system to be update by invoking the `cluster stop` command located in the System V `init` directory. At this point, no services are available.

5. Run cluconfig on the first updated cluster system. When prompted whether to use the existing cluster database, specify **`yes`**. The cluster configuration prompts will be displayed with default parameters set to those of the current configuration. If no changes are necessary, just press [Enter] to accept the existing value[s].

6. Start the cluster software on the first updated cluster system by invoking the `cluster start` command located in the System V `init` directory. At this point, services may become available. For example:

   ```
   /sbin/service cluster restart
   ```

7. Install the latest cluster software on the second cluster system to be updated by invoking the following command:

   ```
   rpm --upgrade clumanager-x.rpm
   ```

   Replace *x* with the version of Red Hat Cluster Manager currently available.

8. On the second updated cluster system,run the `/sbin/cluconfig --init=`*`raw_file`* command, where *raw_file* specifies the primary quorum partition. The script will use the information specified for the first cluster system as the default. For example:

```
cluconfig --init=/dev/raw/raw1
```

9. Start the cluster software on the second cluster system by invoking the `cluster start` command located in the System V `init` directory. For example:

```
/sbin/service cluster start
```

8.8 Reloading the Cluster Database

Invoke the `cluadmin` utility and use the `cluster reload` command to force the cluster to re-read the cluster database. For example:

```
cluadmin> cluster reload
```

8.9 Changing the Cluster Name

Invoke the `cluadmin` utility and use the `cluster name` *`cluster_name`* command to specify a name for the cluster. The cluster name is used in the display of the `clustat` command. For example:

```
cluadmin> cluster name Accounting Team Fileserver
Accounting Team Fileserver
```

8.10 Reinitializing the Cluster

In rare circumstances, you may want to reinitialize the cluster systems, services, and database. Be sure to back up the cluster database before reinitializing the cluster. See Section 8.5, *Backing Up and Restoring the Cluster Database* for information.

To completely reinitialize the cluster, follow these steps:

1. Disable all the running cluster services.
2. Stop the cluster daemons on both cluster systems by invoking the `cluster stop command` located in the System V `init` directory on both cluster systems. For example:

```
/sbin/service cluster stop
```

3. Install the cluster software on both cluster systems. See Section 3.1, *Steps for Installing and Initializing the Cluster Software* for information.
4. On one cluster system, run the cluconfig utility. When prompted whether to use the existing cluster database, specify **no**. This will delete any state information and cluster database from the quorum partitions.
5. After cluconfig completes, follow the utility's instruction to run the `cluconfig` command on the other cluster system. For example:

```
/sbin/cluconfig --init=/dev/raw/raw1
```

6. Start the cluster daemons by invoking the `cluster start` command located in the System V `init` directory on both cluster systems. For example:

```
/sbin/service cluster start
```

8.11 Disabling the Cluster Software

It may become necessary to temporarily disable the cluster software on a member system. For example, if a cluster system experiences a hardware failure, an administrator may want to reboot the system, but prevent it from rejoining the cluster in order to perform maintenance on the system.

Use the `/sbin/chkconfig` utility to be able to boot a cluster system, without allowing it to rejoin the cluster. For example:

```
/sbin/chkconfig --del cluster
```

When you want the system to rejoin the cluster, use the following command:

```
/sbin/chkconfig --add cluster
```

You can then reboot the system or run the `cluster start` command located in the System V `init` directory. For example:

```
/sbin/service cluster start
```

8.12 Diagnosing and Correcting Problems in a Cluster

To ensure the proper diagnosis of any problems in a cluster, event logging must be enabled. In addition, if problems arise in a cluster, be sure to set the severity level to **debug** for the cluster daemons. This will log descriptive messages that may help solve problems. Once any issues have been resolved, reset the debug level back down to its default value of **info** to avoid excessively large log message files from being generated.

If problems occur while running the `cluadmin` utility (for example, problems enabling a service), set the severity level for the clusvcmgrd daemon to **debug**. This will cause debugging messages to be displayed while running the `cluadmin` utility. See Section 8.6, *Modifying Cluster Event Logging* for more information.

Use Table 8–5, *Diagnosing and Correcting Problems in a Cluster* to troubleshoot issues in a cluster.

Table 8–5 Diagnosing and Correcting Problems in a Cluster

Problem	Symptom	Solution
SCSI bus not terminated	SCSI errors appear in the log file	Each SCSI bus must be terminated only at the beginning and end of the bus. Depending on the bus configuration, it might be necessary to enable or disable termination in host bus adapters, RAID controllers, and storage enclosures. To support hot plugging, external termination is required to terminate a SCSI bus. In addition, be sure that no devices are connected to a SCSI bus using a stub that is longer than 0.1 meter. See Section 2.4.4, *Configuring Shared Disk Storage* and Section A.3, *SCSI Bus Termination* for information about terminating different types of SCSI buses.
SCSI bus length greater than maximum limit	SCSI errors appear in the log file	Each type of SCSI bus must adhere to restrictions on length, as described in Section A.4, *SCSI Bus Length*. In addition, ensure that no single-ended devices are connected to the LVD SCSI bus, because this will cause the entire bus to revert to a single-ended bus, which has more severe length restrictions than a differential bus.

Problem	Symptom	Solution
SCSI identification numbers not unique	SCSI errors appear in the log file	Each device on a SCSI bus must have a unique identification number. See Section A.5, *SCSI Identification Numbers* for more information.
SCSI commands timing out before completion	SCSI errors appear in the log file	The prioritized arbitration scheme on a SCSI bus can result in low-priority devices being locked out for some period of time. This may cause commands to time out, if a low-priority storage device, such as a disk, is unable to win arbitration and complete a command that a host has queued to it. For some workloads, this problem can be avoided by assigning low-priority SCSI identification numbers to the host bus adapters. See Section A.5, *SCSI Identification Numbers* for more information.

Problem	Symptom	Solution
Mounted quorum partition	Messages indicating checksum errors on a quorum partition appear in the log file	Be sure that the quorum partition raw devices are used only for cluster state information. They cannot be used for cluster services or for non-cluster purposes, and cannot contain a file system. See *Configuring Quorum Partitions* in Section 2.4.4 for more information. These messages could also indicate that the underlying block device special file for the quorum partition has been erroneously used for non-cluster purposes.
Service file system is unclean	A disabled service cannot be enabled	Manually run a checking program such as `fsck`. Then, enable the service. Note that the cluster infrastructure does by default run `fsck` with the `-p` option to automatically repair file system inconsistencies. For particularly egregious error types you may be required to manually initiate filesystem repair options.

Problem	Symptom	Solution
Quorum partitions not set up correctly	Messages indicating that a quorum partition cannot be accessed appear in the log file	Run the `cludiskutil -t` command to check that the quorum partitions are accessible. If the command succeeds, run the `cludiskutil -p` command on both cluster systems. If the output is different on the systems, the quorum partitions do not point to the same devices on both systems. Check to make sure that the raw devices exist and are correctly specified in the `/etc/sysconfig/rawdevices` file. See *Configuring Quorum Partitions* in Section 2.4.4 for more information. These messages could also indicate that **yes** was not chosen when prompted by the cluconfig utility to initialize the quorum partitions. To correct this problem, run the utility again.
Cluster service operation fails	Messages indicating the operation failed to appear on the console or in the log file	There are many different reasons for the failure of a service operation (for example, a service stop or start). To help identify the cause of the problem, set the severity level for the cluster daemons to **debug** in order to log descriptive messages. Then, retry the operation and examine the log file. See Section 8.6, *Modifying Cluster Event Logging* for more information.

Problem	Symptom	Solution
Cluster service stop fails because a file system cannot be unmounted	Messages indicating the operation failed appear on the console or in the log file	Use the `fuser` and `ps` commands to identify the processes that are accessing the file system. Use the `kill` command to stop the processes. Use the `lsof -t` *`file_system`* command to display the identification numbers for the processes that are accessing the specified file system. If needed, Pipe the output to the `kill` command. To avoid this problem, be sure that only cluster-related processes can access shared storage data. In addition, modify the service and enable forced unmount for the file system. This enables the cluster service to unmount a file system even if it is being accessed by an application or user.
Incorrect entry in the cluster database	Cluster operation is impaired	The `cluadmin` utility can be used to examine and modify service configuration. Additionally, the cluconfig utility is used to modify cluster parameters.
Incorrect Ethernet heartbeat entry in the cluster database or `/etc/hosts` file	Cluster status indicates that a Ethernet heartbeat channel is **OFFLINE** even though the interface is valid	Examine and modify the cluster configuration by running the cluconfig utility, as specified in Section 8.4, *Modifying the Cluster Configuration*, and correct the problem. In addition, be sure to use the `ping` command to send a packet to all the network interfaces used in the cluster.

Problem	Symptom	Solution
Loose cable connection to power switch	Power switch status is **Timeout**	Check the serial cable connection.
Power switch serial port incorrectly specified in the cluster database	Power switch status indicates a problem	Examine the current settings and modify the cluster configuration by running the cluconfig utility, as specified in Section 8.4, *Modifying the Cluster Configuration*, and correct the problem.
Heartbeat channel problem	Heartbeat channel status is **OFFLINE**	Examine the current settings and modify the cluster configuration by running the cluconfig utility, as specified in Section 8.4, *Modifying the Cluster Configuration*, and correct the problem. Verify that the correct type of cable is used for each heartbeat channel connection. Run the command `ping` to each cluster system over the network interface for each Ethernet heartbeat channel.

9 Configuring and using the Red Hat Cluster Manager GUI

Red Hat Cluster Manager includes a graphical user interface (GUI) which allows an administrator to graphically monitor cluster status. The GUI does not allow configuration changes or management of the cluster, however.

9.1 Setting up the JRE

The Red Hat Cluster Manager GUI can be run directly on a cluster member, or from a non-cluster member to facilitate remote web based monitoring. The GUI itself is implemented as a java applet that runs in a Web browser. For this reason, it is required that all systems on which the GUI is intended to be run must have the Java Runtime Environment (JRE) installed and configured as a browser plug-in. The cluster manager GUI can be run using either the IBM JRE or the Sun JRE.

WARNING

The IBM JRE is included and installed by default on Red Hat Linux Advanced Server. The installation and use of the Sun JRE with Red Hat Linux Advanced Server is *not* supported. The information in Section 9.1.2, *Setting up the Sun JRE* is provided only as a convenience to users who wish to deploy it.

9.1.1 Setting up the IBM JRE

The IBM JRE is automatically installed on the cluster members in the `IBMJava2-JRE-1.3.<version>` RPM package (where *<version>* is the version of the IBM JRE currently available). This package places the JRE in `/opt/IBMJava2-13/`.

The RPM installation of the JRE will automatically setup the required plugin link as required by the Mozilla Web browser.

To enable the IBM JRE for usage with the Netscape Navigator version 4.x, follow the instructions supplied with the JRE. For example, as specified in `/opt/IBMJava2-131/docs/README-EN.JRE.HTML` of IBM JRE v.1.3.1-3,the instructions specifiy the following commands:

```
cd /usr/lib/netscape/plugins
ln -s /opt/IBMJava2-131/jre/bin/javaplugin.so
```

9.1.2 Setting up the Sun JRE

If the cluster GUI is to be installed on a non-cluster member, it may be necessary to download and install the JRE. The JRE can be obtained from Sun's java.sun.com site. For example, at the time of publication, the specific page is http://java.sun.com/j2se/1.3/jre/download-linux.html

After downloading the JRE, run the downloaded program (for example, `j2re-1_3_1_02-linux-i386-rpm.bin`) and confirm the license agreement. This results in the extraction of the JRE's RPM, `jre-1.3.1_02.i386.rpm`, which is installed using `rpm`.

After installing the JRE, enable the browser that is intended to run the GUI applet with Java support. The procedure needed to enable java support is dependent on the specific browser and browser version used. Refer to the installation instructions for java plugins found on the JRE download page.

For example, to enable java for usage with release 4 of Netscape Navigator/Communicator, add the following in the `~/.bash_profile` file:

```
export NPX_PLUGIN_PATH=/usr/java/jre1.3.1_02/plugin/i386/ns4:/usr/lib/netscape/plugins
```

The specific directory path may vary. Also, note that the JRE's installation instructions are different for release 6 of Netscape Communicator.

The following example describes the setup step necessary to configure the Mozilla browser to enable the java plugin:

```
ln -s /usr/java/jre1.3.1_02/plugin/i386/ns600/libjavaplugin_oji.so \
  /usr/lib/mozilla/plugins/
```

9.2 Configuring Cluster Monitoring Parameters

When the cluconfig utility is run to configure the cluster, it will prompt for configuration information which relates to operation of the Cluster Manager GUI.

The first GUI related parameter asks whether or not to configure a cluster alias. For example:

```
Enter IP address for cluster alias [NONE]: 172.16.33.128
```

A cluster alias consists of a floating IP address which will be active on either of the cluster members. For the purposes of this example, the IP is set to 172.16.33.128. It is useful to use this IP address (or associated hostname) within the browser when pointing it at the cluster member to monitor. If electing not to configure a cluster alias, then it is required to designate individual cluster members in order to monitor the cluster status using the GUI. The benefit of specifying the cluster alias is that the GUI will continue to be responsive as long as at least one cluster member is active.

The second GUI related parameter prompted for in cluconfig asks whether or not to allow remote monitoring. For example:

```
Do you wish to allow remote monitoring of the cluster? yes/no [yes]:
```

If **no** is answered, the Cluster Manager GUI can still be run locally on either of the cluster members; but, it is not possible to monitor the cluster from non-cluster systems.

9.3 Enabling the Web Server

In order to enable usage of the Cluster Manager GUI, all cluster members must be running a web server. For example, the HTTP daemon must be running for the Apache web server to operate.

Note

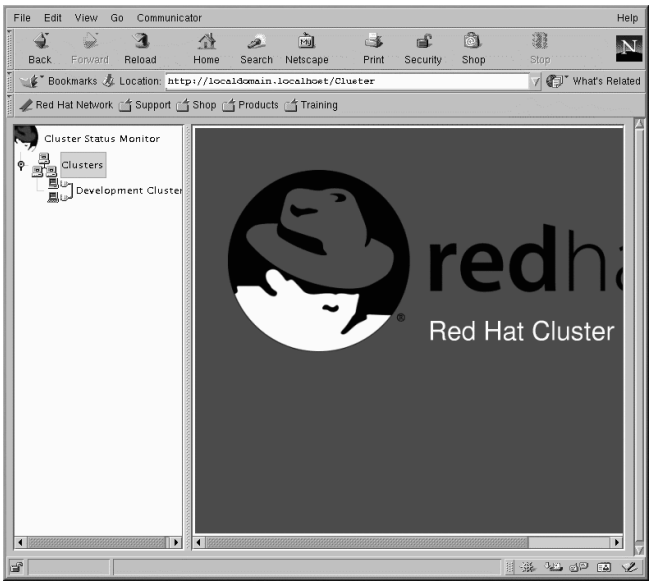
If intending to utilize the Cluster Manager GUI, a highly available Apache service cannot be configured as described in Chapter 7, *Apache Services*. This restriction exists because a highly available Apache service causes the web server to be running on only one cluster member at a time.

Correct operation of the Cluster Manager GUI requires that Apache's document root remain at the default setting of `/var/www/html` as this is where the directory cluster and its corresponding web content is installed.

9.4 Starting the Red Hat Cluster Manager GUI

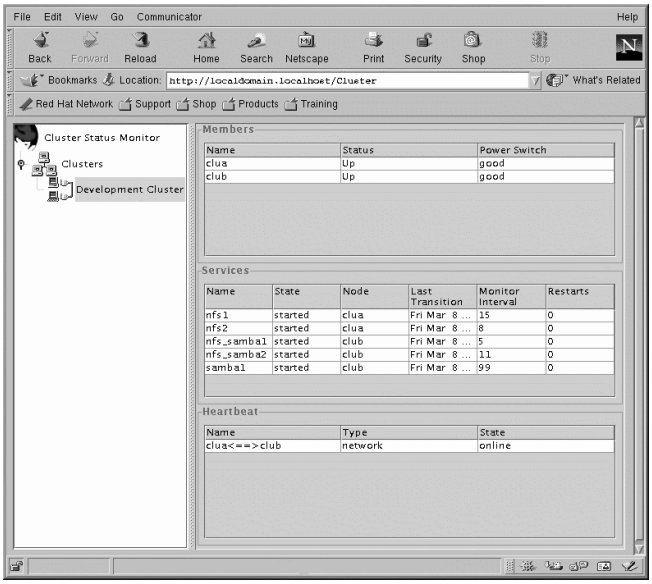
After configuring the java browser plugin, start the Cluster Manager GUI by specifying the appropriate URL to the browser. The GUI's URL consists of either the cluster member name or cluster alias, followed by "/Cluster". For example, using the cluster alias from the preceding example, the corresponding URL would be **`http://clu2alias/Cluster`**. When the GUI applet starts up, a splash screen will appear on the right and a tree view on the left. To begin cluster monitoring, double-click the **Clusters** label within the tree view, which will reveal the cluster name (as initially configured using cluconfig).

Figure 9–1 Red Hat Cluster Manager GUI Splashscreen



By double-clicking on the cluster name within the tree view, the right side of the GUI will then fill with cluster statistics, as shown in Figure 9–2, *Red Hat Cluster Manager GUI Main Screen.* These statistics depict the status of the cluster members, the services running on each member, and the heartbeat channel status.

Figure 9–2 Red Hat Cluster Manager GUI Main Screen

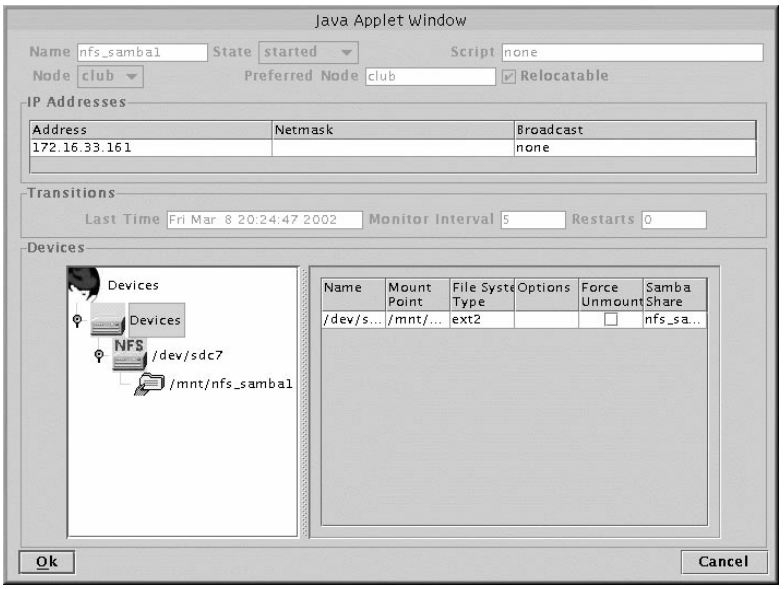


By default, the cluster statistics will be refreshed every 5 seconds. Clicking the right mouse button on the cluster name within the tree view will load a dialog allowing modification of the default update interval.

9.4.1 Viewing Configuration Details

After initiating cluster monitoring, it is possible to obtain detailed configuration information by double-clicking on any of the cluster status items. Following the prior example, double click on the **nfs_samba1** service and the Service Information window will appear as seen in Figure 9–3, *Red Hat Cluster Manager GUI Configuration Details Screen*:

Figure 9–3 Red Hat Cluster Manager GUI Configuration Details Screen



In Figure 9–3, *Red Hat Cluster Manager GUI Configuration Details Screen*, notice that the detailed device information appears after clicking on the individual device parameters.

In addition to obtaining detailed configuration information related to cluster services, it is also possible to view the configuration of individual cluster members and heartbeat channels by double-clicking within the relevant section of the GUI.

A Supplementary Hardware Information

The information in the following sections can help you set up a cluster hardware configuration. In some cases, the information is vendor specific.

A.1 Setting Up Power Switches

A.1.1 Setting up RPS-10 Power Switches

If an RPS-10 Series power switch is used as a part of a cluster, be sure of the following:

- Set the rotary address on both power switches to 0. Be sure that the switch is positioned correctly and is not between settings.
- Toggle the four Setup switches on both power switches, as follows:

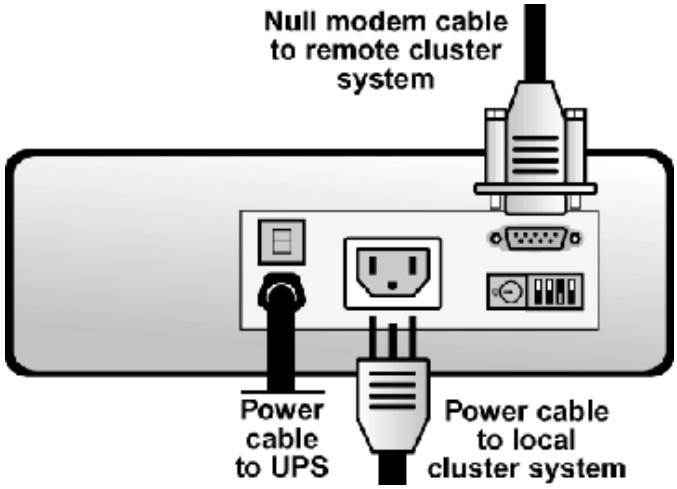
Table A–1 Setting up RPS-10 Power Switches

Switch	Function	Up Position	Down Position
1	Data rate		X
2	Toggle delay		X
3	Power up default	X	
4	Unused		X

- Ensure that the serial port device special file (for example, `/dev/ttyS1`) that is specified in the `/etc/cluster.conf` file corresponds to the serial port to which the power switch's serial cable is connected.
- Connect the power cable for each cluster system to its own power switch.
- Use null modem cables to connect each cluster system to the serial port on the power switch that provides power to the other cluster system.

Figure A–1, *RPS-10 Power Switch Hardware Configuration* shows an example of an RPS-10 Series power switch configuration.

Figure A–1 RPS-10 Power Switch Hardware Configuration



See the RPS-10 documentation supplied by the vendor for additional installation information. Note that the information provided in this document supersedes the vendor information.

A.1.2 Setting up WTI NPS Power Switches

The WTI NPS-115 and NPS-230 power switch is a network attached device. Essentially it is a power strip with network connectivity enabling power cycling of individual outlets. Only 1 NPS is needed within the cluster (unlike the RPS-10 model where a separate switch per cluster member is required).

Since there is no independent means whereby the cluster software can verify that each cluster member system has been plugged into the appropriate plug on the back of the NPS power switch, please take care to ensure correct setup. Failure to do so will cause the cluster software to incorrectly conclude that a successful power cycle has occurred.

When setting up the NPS switch the following configuration guidelines should be followed.

When configuring the power switch itself:

- Assign a **System Password** (under the **General Parameters** menu). Note: this password is stored in clear text in the cluster configuration file, so choose a password which differs from the system's password. (Although, the file permissions for that file `/etc/cluster.conf` are only readable by root.)
- Do not assign a password under the **Plug Parameters**.

- Assign system names to the Plug Parameters, (for example, *clu1* to plug 1, *clu2* to plug 2 — assuming these are the cluster member names).

When running cluconfig to specify power switch parameters:

- Specify a switch type of **WTI_NPS**.
- Specify the password you assigned to the NPS switch (refer to Step 1 in prior section).
- When prompted for the plug/port number, specify the same name as assigned in Step 3 in the prior section.

Note

It has been observed that the NPS power switch may become unresponsive when placed on networks which have high occurrences of broadcast or multicast packets. In these cases isolating the power switch to a private subnet may be needed.

The NPS-115 power switch has a very useful feature which can accommodate power cycling cluster members with dual power supplies. The NPS-115 consists of 2 banks of power outlets, each of which is independently powered and has 4 plugs. Each power plug of the NPS-115 gets plugged into a separate power source (presumably a separate UPS). For cluster members with dual power supplies, plug their power cords into an outlet in each bank. Then, when configuring the NPS-115 and assigning ports, simply assign the same name to outlets in each bank that has been plugged the corresponding cluster member. For example, suppose the cluster members were *clu3* and *clu4*, where *clu3* is plugged into outlets 1 and 5, and *clu4* is plugged into outlets 2 and 6:

```
Plug | Name           | Status  | Boot Delay | Password         | Default |
-----+----------------+---------+------------+------------------+---------+
 1   | clu3           |   ON    |   5  sec   | (undefined)      |   ON    |
 2   | clu4           |   ON    |   5  sec   | (undefined)      |   ON    |
 3   | (undefined)    |   ON    |   5  sec   | (undefined)      |   ON    |
 4   | (undefined)    |   ON    |   5  sec   | (undefined)      |   ON    |
 5   | clu3           |   ON    |   5  sec   | (undefined)      |   ON    |
 6   | clu4           |   ON    |   5  sec   | (undefined)      |   ON    |
 7   | (undefined)    |   ON    |   5  sec   | (undefined)      |   ON    |
 8   | (undefined)    |   ON    |   5  sec   | (undefined)      |   ON    |
-----+----------------+---------+------------+------------------+---------+
```

By specifying the same name to multiple outlets, in response to a power cycle command, all outlets with the same name will be power cycled. In this manner, a cluster member with dual power supplies can be successfully power cycled. Under this dual configuration, the parameters specified to cluconfig are the same as the single configuration described above.

A.1.3 Setting up Baytech Power Switches

The following information pertains to the RPC-3 and PRC-5 power switches.

The Baytech power switch is a network attached device. Essentially, it is a power strip with network connectivity enabling power cycling of individual outlets. Only 1 Baytech switch is needed within the cluster (unlike the RPS-10 model where a separate switch per cluster member is required).

Since there is no independent means whereby the cluster software can verify that you have plugged each cluster member system into the appropriate plug on the back of the Baytech power switch, please take care to ensure correct setup. Failure to do so will cause the cluster software to incorrectly conclude a successful power cycle has occurred.

Note

As shipped from the manufacturer, all of the outlets of a Baytech switch are set to off. To power on the outlets into which the cluster members are plugged, use the Baytech's configuration menus by starting from the main menu, then selecting **Outlet Control**. From there, it is possible to turn on individual outlets, for example **on 1**, **on 2**, etc.

When setting up the Baytech switch the following configuration guidelines should be followed.

When configuring the Baytech power switch itself:

1. Using a serial connection, assign the IP address related parameters.
2. Under the **Access** => **Network access** menu, ensure that both **Prompt for user name** and **Prompt for password** are enabled.
3. Assign a user name and password under the **Manage Users** menu or use the default "admin" account with an assigned password. Note: this password is stored in clear text in the cluster configuration file, so choose a password which differs from the system's password (even though the file permissions for the file `/etc/cluster.conf` are only readable by root).
4. To assign the system names to the corresponding outlets, go to the **Configuration** menu, followed by the **Outlets** menu, and finally **Name Outlets** (for example, *clu1* to outlet 1, *clu2* to outlet 2 — assuming these are the cluster member names).

When running cluconfig to specify power switch parameters:

- Specify a switch type of **BAYTECH**.
- Specify the username and password assigned to the Baytech switch (refer to Step 3 in prior section).

- When prompted for the plug/port number, specify the same name as assigned in Step 4 in prior section.

The following is an example screen output from configuring the Baytech switch which shows that the outlets have been named according to the example cluster names *clu1* and *clu2*.

```
Outlet Operation Configuration Menu
  Enter request, CR to exit.

    1)...Outlet Status Display: enabled
    2)...Command Confirmation : enabled
    3)...Current Alarm Level (amps):  4.1
    4)...Name Outlets
    5)...Outlet Power-up Delay
    6)...Display Outlet Users
   Enter request>4

  Enter number of outlet to name, CR to exit.
    1)...clu1
    2)...clu2
    3)...Outlet 3
    4)...Outlet 4
    5)...Outlet 5
    6)...Outlet 6
    7)...Outlet 7
    8)...Outlet 8
```

A.1.4 Setting up Watchdog Power Switches

A description of the usage model for watchdog timers as a cluster data integrity provision appears in Section 2.1.3, *Choosing the Type of Power Controller*. As described in that section, there are two variants of watchdog timers: Hardware-based and software-based.

The following details the configuration tasks required in order to setup watchdog timer usage in a cluster hardware configuration.

Regardless of which type of watchdog timer is employed, it is necessary to create the device special file appropriate for the watchdog timer. This can be accomplished as follows:

```
# cd /dev
# ./MAKEDEV watchdog
```

When running the cluconfig utility, where it prompts for the power switch type, specify a type of SW_WATCHDOG, regardless of the specific type of watchdog timer in use.

Configuring the Software Watchdog Timer

Any cluster system can utilize the software watchdog timer as a data integrity provision, as no dedicated hardware components are required. If you have specified a power switch type of **SW_WATCHDOG** while using the cluconfig utility, the cluster software will automatically load the corresponding loadable kernel module called `softdog`.

If the cluster is configured to utilize the software watchdog timer, the cluster quorum daemon (`cluquorumd`) will periodically reset the timer interval. Should `cluqourumd` fail to reset the timer, the failed cluster member will reboot itself.

When using the software watchdog timer, there is a small risk that the system will hang in such a way that the software watchdog thread will not be executed. In this unlikely scenario, the other cluster member may takeover services of the apparently hung cluster member. Generally, this is a safe operation; but in the unlikely event that the hung cluster member resumes, data corruption could occur. To further lessen the chance of this vulnerability occurring when using the software watchdog timer, administrators should also configure the NMI watchdog timer.

Enabling the NMI Watchdog Timer

If you are using the software watchdog timer as a data integrity provision, it is also recommended to enable the Non-Maskable Interrupt (NMI) watchdog timer to enhance the data integrity guarantees. The NMI watchdog timer is a different mechanism for causing the system to reboot in the event of a hang scenario where interrupts are blocked. This NMI watchdog can be used in conjunction with the software watchdog timer.

Unlike the software watchdog timer which is reset by the cluster quorum daemon (`cluquorumd`), the NMI watchdog timer counts system interrupts. Normally, a healthy system will receive hundreds of device and timer interrupts per second. If there are no interrupts in a 5 second interval, a system hang has occurred and the NMI watchdog timer will expire, initiating a system reboot.

A robust data integrity solution can be implemented by combining the health monitoring of the the cluster quorum daemon with the software watchdog timer along with the low-level system status checks of the NMI watchdog.

Correct operation of the NMI watchdog timer mechanism requires that the cluster members contain an APIC chip on the main system board. The majority of contemporary systems do include the APIC component. Generally, Intel-based SMP systems and Intel-based uniprocessor systems with SMP system boards (2+ cpu slots/sockets, but only one CPU) are known the support the NMI watchdog.

Note

There may be other server types that support NMI watchdog timers aside from ones with Intel-based SMP system boards. Unfortunately, there is no simple way to test for this functionality other than simple trial and error.

The NMI watchdog is enabled on supported systems by adding *nmi_watchdog=1* to the kernel's command line. Here is an example `/etc/grub.conf`:

```
#
# grub.conf
#
default=0
timeout=10
splashimage=(hd0,0)/grub/splash.xpm.gz
title HA Test Kernel (2.4.9-10smp)
        root (hd0,0)
        # This is the kernel's command line.
        kernel /vmlinuz-2.4.9-10smp ro root=/dev/hda2 nmi_watchdog=1

# end of grub.conf
```

On systems using lilo instead of grub, add **`nmi_watchdog=1`** to the "append" section in `/etc/lilo.conf`. For example:

```
#
# lilo.conf
#
prompt
timeout=50
default=linux
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
lba32

image=/boot/vmlinuz-2.4.9-10smp
        label=linux
        read-only
        root=/dev/hda2
        append="nmi_watchdog=1"

# end of lilo.conf
```

In order to determine if the server supports the NMI watchdog timer, first try adding "nmi_watchdog=1" to the kernel command line as described above. After the system has booted, log in as root and type:

```
cat /proc/interrupts
```

The output should appear similar to the following:

```
           CPU0
  0:    5623100          XT-PIC  timer
  1:         13          XT-PIC  keyboard
  2:          0          XT-PIC  cascade
  7:          0          XT-PIC  usb-ohci
  8:          1          XT-PIC  rtc
  9:     794332          XT-PIC  aic7xxx, aic7xxx
 10:     569498          XT-PIC  eth0
 12:         24          XT-PIC  PS/2 Mouse
 14:          0          XT-PIC  ide0
NMI:    5620998
LOC:    5623358
ERR:          0
MIS:          0
```

The relevant portion of the above output is to verify that the `NMI` id appears on the left side. If NMI value (in the middle column) is non-zero, the server supports the NMI watchdog.

If this approach fails, that is, `NMI` is zero, try passing *nmi_watchdog=2* to the kernel instead of *nmi_watchdog=1* in the manner described previously. Again, check /proc/interrupts after the system boots. If `NMI` is non-zero, the NMI watchdog has been configured properly. If `NMI` is zero, your system does not support the NMI watchdog timer.

Configuring a Hardware Watchdog Timer

The kernel provides driver support for various types of hardware watchdog timers. Some of these timers are implemented directly on the system board, whereas others are separate hardware components such as PCI cards. Hardware based watchdog timers provide excellent data integrity provisions in the cluster because they operate independently of the system processor and are therefore fully operational in rebooting a system in the event of a system hang.

Due to a lack of uniformity among low-level hardware watchdog components, it is difficult to make generalizations describing how to know if a particular system contains such components. Many low-level hardware watchdog components are not self-identifying.

The kernel provides support for the following hardware watchdog variants listed in Table A–2, *Hardware Watchdog Timers* :

Table A–2 Hardware Watchdog Timers

Card/Timer	Driver
Acquire SBC	acquirewdt
Advantech SBC	advantechwdt
Intel-810 based TCO WDT	i810-tco
Eurotech CPU-1220/1410 WDT	eurotech
IB700 WDT	ib700
60xx SBC WDT	sbc60xxwdt
W83877F WDT	w83877f
Netwinder W83977AF	wdt977
Industrial Computer WDT500	wdt
Industrial Computer WDT501	wdt
Industrial Computer WDT500PCI	wdt_pci
Industrial Computer WDT501PCI	wdt_pci

In order to configure any of the above watchdog timers into the kernel, it is necessary to place a corresponding entry into the /etc/modules.conf file. For example, if an Intel-810 based TCO WDT is to be used, the following line should be added to /etc/modules.conf:

```
alias watchdog i810-tco
```

A.1.5 Other Network Power Switches

The cluster software includes support for a range of power switch types. This range of power switch module support originated from developers at Mission Critical Linux, Inc. and as part of the open source Linux-HA project. Time and resource constraints did not allow for comprehensive testing of the complete range of switch types. As such, the associated power switch STONITH modules are considered latent features. Examples of these other power switch modules include:

- APC Master Switch: http://www.apc.com

Note

It has been observed that the Master Switch may become unresponsive when placed on networks which have high occurrences of broadcast or multi-cast packets. In these cases, isolate the power switch to a private subnet.

- APC Serial On/Off Switch (partAP9211): http://www.apc.com

Note

This switch type does not provide a means for the cluster to query its status. Therefore the cluster always assumes it is connected and operational.

A.1.6 Setting up Power Switch type None

It is possible to configure a cluster that does not include any power switch functionality. As described in Section 2.1.3, *Choosing the Type of Power Controller*, configuring a cluster without any power switch provisions is not recommended due to data corruption vulnerabilities under certain failover scenarios.

In order to setup a cluster that does not include any power switch provisions, simply select the type **NONE** when prompted for power switch type from the cluconfig utility.

Note

Usage of power switch type **NONE** is discouraged because it does not protect data integrity in the event of system hang. If your cluster configuration does not include hardware power switches, then the software watchdog type is recommended.

A.2 SCSI Bus Configuration Requirements

SCSI buses must adhere to a number of configuration requirements in order to operate correctly. Failure to adhere to these requirements will adversely affect cluster operation and application and data availability.

The following are **SCSI bus configuration requirements**:

- Buses must be terminated at each end. See Section A.3, *SCSI Bus Termination* for more information.
- Buses must not extend beyond the maximum length restriction for the bus type. Internal cabling must be included in the length of the SCSI bus. See Section A.4, *SCSI Bus Length* for more information.
- All devices (host bus adapters and disks) on a bus must have unique SCSI identification numbers. See Section A.5, *SCSI Identification Numbers* for more information.
- The Linux device name for each shared SCSI device must be the same on each cluster system. For example, a device named `/dev/sdc` on one cluster system must be named `/dev/sdc` on the other cluster system. One way to ensure that devices are named the same is by using identical hardware for both cluster systems.

Use the system's configuration utility to set SCSI identification numbers and enable host bus adapter termination. When the system boots, a message is displayed describing how to start the utility. For example, the utility will prompt the user to press [Ctrl]-[A], and follow the prompts to perform a particular task. To set storage enclosure and RAID controller termination, refer to the vendor documentation. See Section A.3, *SCSI Bus Termination* and Section A.5, *SCSI Identification Numbers* for more information.

See http://www.scsita.org and the following sections for detailed information about SCSI bus requirements.

A.3 SCSI Bus Termination

A SCSI bus is an electrical path between two terminators. A device (host bus adapter, RAID controller, or disk) attaches to a SCSI bus by a short **stub**, which is an unterminated bus segment that usually must be less than 0.1 meter in length.

Buses must have only two terminators located at opposing ends of the bus. Additional terminators, terminators that are not at the ends of the bus, or long stubs will cause the bus to operate incorrectly. Termination for a SCSI bus can be provided by the devices connected to the bus or by external terminators, if the internal (onboard) device termination can be disabled.

Testing has shown that external termination on HBAs that run at speeds greater than 80MB/sec. does not work reliably.

When disconnecting a device from a single-initiator SCSI bus follow these guidelines:

- Unterminated SCSI cables must not be connected to an operational host bus adapter or storage device.
- Connector pins must not bend or touch an electrical conductor while the SCSI cable is disconnected.

- To disconnect a host bus adapter from a single-initiator bus, you must disconnect the SCSI cable first from the RAID controller and then from the adapter. This ensures that the RAID controller is not exposed to any erroneous input.
- Protect connector pins from electrostatic discharge while the SCSI cable is disconnected by wearing a grounded anti-static wrist guard and physically protecting the cable ends from contact with other objects.
- Do not remove a device that is currently participating in any SCSI bus transactions.

To enable or disable an adapter's internal termination, use the system BIOS utility. When the system boots, a message is displayed describing how to start the utility. For example, many utilities will prompt users to press Ctrl-A. Follow the prompts for setting the termination. At this point, it is also possible to set the SCSI identification number, as needed, and disable SCSI bus resets. See Section A.5, *SCSI Identification Numbers* for more information.

To set storage enclosure and RAID controller termination, see the vendor documentation.

A.4 SCSI Bus Length

A SCSI bus must adhere to length restrictions for the bus type. Buses that do not adhere to these restrictions will not operate properly. The length of a SCSI bus is calculated from one terminated end to the other, and must include any cabling that exists inside the system or storage enclosures.

A cluster supports LVD (low voltage differential) buses. The maximum length of a single-initiator LVD bus is 25 meters. The maximum length of a multi-initiator LVD bus is 12 meters. According to the SCSI standard, a single-initiator LVD bus is a bus that is connected to only two devices, each within 0.1 meter from a terminator. All other buses are defined as multi-initiator buses.

Do not connect any single-ended devices to an LVD bus, or the bus will convert to a single-ended bus, which has a much shorter maximum length than a differential bus.

A.5 SCSI Identification Numbers

Each device on a SCSI bus must have a unique SCSI identification number. Devices include host bus adapters, RAID controllers, and disks.

The number of devices on a SCSI bus depends on the data path for the bus. A cluster supports wide SCSI buses, which have a 16-bit data path and support a maximum of 16 devices. Therefore, there are sixteen possible SCSI identification numbers that can be assigned to the devices on a bus.

In addition, SCSI identification numbers are prioritized. Use the following priority order to assign SCSI identification numbers:

```
7 - 6 - 5 - 4 - 3 - 2 - 1 - 0 - 15 - 14 - 13 - 12 - 11 - 10 - 9 - 8
```

The previous order specifies that 7 is the highest priority, and 8 is the lowest priority. The default SCSI identification number for a host bus adapter is 7, because adapters are usually assigned the highest priority. It is possible to assign identification numbers for logical units in a RAID subsystem by using the RAID management interface.

To modify an adapter's SCSI identification number, use the system BIOS utility. When the system boots, a message is displayed describing how to start the utility. For example, a user may be prompted to press [Ctrl]-[A], and follow the prompts for setting the SCSI identification number. At this point, it is possible to enable or disable the adapter's internal termination, as needed, and disable SCSI bus resets. See Section A.3, *SCSI Bus Termination* for more information.

The prioritized arbitration scheme on a SCSI bus can result in low-priority devices being locked out for some period of time. This may cause commands to time out, if a low-priority storage device, such as a disk, is unable to win arbitration and complete a command that a host has queued to it. For some workloads, it is possible to avoid this problem by assigning low-priority SCSI identification numbers to the host bus adapters.

A.6 Host Bus Adapter Features and Configuration Requirements

The following table describes some recommended SCSI and Fibre Channel host bus adapters. It includes information about adapter termination and how to use the adapters in single initiator SCSI buses and Fibre Channel interconnects.

The specific product devices listed in the table have been tested. However, other devices may also work well in a cluster. It is possible to use a host bus adapter other than a recommended one. The information in the table can help to determine if the device has the features and characteristics that will enable it to work in a cluster.

Table A–3 Host Bus Adapter Features and Configuration Requirements

Host Bus Adapter	Features	Single-Initiator Configuration
Adaptec 2940U2W	Ultra2, wide, LVD. HD68 external connector. One channel, with two bus segments. Set the onboard termination by using the BIOS utility. Onboard termination is disabled when the power is off.	Set the onboard termination to automatic (the default). Use the internal SCSI connector for private (non-cluster) storage.
Qlogic QLA1080	Ultra2, wide, LVD VHDCI external connector One channel Set the onboard termination by using the BIOS utility. Onboard termination is disabled when the power is off, unless jumpers are used to enforce termination.	Set the onboard termination to automatic (the default). Use the internal SCSI connector for private (non-cluster) storage.

Host Bus Adapter	Features	Single-Initiator Configuration
Tekram DC-390U2W	Ultra2, wide, LVD HD68 external connector One channel, two segments Onboard termination for a bus segment is disabled if internal and external cables are connected to the segment. Onboard termination is enabled if there is only one cable connected to the segment. Termination is disabled when the power is off.	Use the internal SCSI connector for private (non-cluster) storage.
Adaptec 29160	Ultra160 HD68 external connector One channel, with two bus segments Set the onboard termination by using the BIOS utility. Termination is disabled when the power is off, unless jumpers are used to enforce termination.	Set the onboard termination to automatic (the default). Use the internal SCSI connector for private (non-cluster) storage.

Host Bus Adapter	Features	Single-Initiator Configuration
Adaptec 29160LP	Ultra160 VHDCI external connector One channel Set the onboard termination by using the BIOS utility. Termination is disabled when the power is off, unless jumpers are used to enforce termination.	Set the onboard termination to automatic (the default). Use the internal SCSI connector for private (non-cluster) storage.
Adaptec 39160 Qlogic QLA12160	Ultra160 Two VHDCI external connectors Two channels Set the onboard termination by using the BIOS utility. Termination is disabled when the power is off, unless jumpers are used to enforce termination.	Set onboard termination to automatic (the default). Use the internal SCSI connectors for private (non-cluster) storage.

Host Bus Adapter	Features	Single-Initiator Configuration
LSI Logic SYM22915	Ultra160 Two VHDCI external connectors Two channels Set the onboard termination by using the BIOS utility. The onboard termination is automatically enabled or disabled, depending on the configuration, even when the module power is off. Use jumpers to disable the automatic termination.	Set onboard termination to automatic (the default). Use the internal SCSI connectors for private (non-cluster) storage.
Adaptec AIC-7896 on the Intel L440GX+ motherboard (as used on the VA Linux 2200 series)	One Ultra2, wide, LVD port, and one Ultra, wide port. Onboard termination is permanently enabled, so the adapter must be located at the end of the bus.	Termination is permanently enabled, so no action is needed in order to use the adapter in a single-initiator bus.

Table A–4 QLA2200 Features and Configuration Requirements

Host Bus Adapter	Features	Single-Initiator Configuration	Multi-Initiator Configuration
QLA2200 (minimum driver: QLA2x00 V2.23	Fibre Channel arbitrated loop and fabric One channel	Can be implemented with point-to-point links from the adapter to a multi-ported storage device. Hubs are required to connect an adapter to a dual-controller RAID array or to multiple RAID arrays.	Can be implemented with FC hubs or switches

A.7 Tuning the Failover Interval

This section describes how to tune configurable parameters associated with the `cluqourumd` daemon. These parameters control the amount of time that a healthy cluster member will grant a failed cluster member prior to concluding that a failure has occurred. After this time interval has elapsed, the healthy cluster member will power cycle the failed cluster member (depending on the type of power switch in use) and resume services previously running on the failed member.

There are several parameters governing the amount of time prior to initiating failover, including the following:

Table A–5 Failover Interval Parameters in `cluquorumd`

Name	Default (sec.)	Description
pingInterval	2	The frequency at which `cluquorumd` updates its on-disk state information and reads in the state of the other cluster member.

Name	Default (sec.)	Description
sameTimeNetdown	7	The number of intervals that must elapse before concluding a cluster member has failed when the `cluhbd` heartbeat daemon is unable to communicate with the other cluster member
sameTimeNetup	12	The number of intervals that must elapse before concluding a cluster member to have failed, when the `cluhbd` heartbeat daemon is able to communicate with the other cluster member. The value of this parameter must be greater than the sameTimeNetdown parameter.

For example, suppose one cluster member has a hardware fault and ceases to function. In this case, both `cluquorumd` and `cluhbd` will agree that the other cluster member has failed. After a period of (*pingInterval* * *sameTimeNetdown*), (which by default is 14 seconds total), the failover will commence.

In order to tune this failover interval, the `cludb` utility is used. For example, to modify the failover interval down to a duration of 10 seconds, rather than the default of 14 seconds, the value of the *sameTimeNetdown* parameter should be set to 5 as follows:

```
cludb -p cluquorumd%sameTimeNetdown 5
```

Note

Caution should be used when tuning these parameters. If a specified failover interval is too small, there is risk of incorrectly concluding that a member has failed during spikes in activity.

Also note that if the power switch type "watchdog" is in use, it is necessary that the watchdog expiration interval be less than the failover interval. It is recommended that the watchdog interval be set to approximately 2/3 the duration of the failover interval.

B Supplementary Software Information

The information in the following sections can assist in the management of the cluster software configuration.

B.1 Cluster Communication Mechanisms

A cluster uses several intra-cluster communication mechanisms to ensure data integrity and correct cluster behavior when a failure occurs. The cluster uses these mechanisms to:

- Control when a system can become a cluster member
- Determine the state of the cluster systems
- Control the behavior of the cluster when a failure occurs

The cluster communication mechanisms are as follows:

- Quorum disk partitions

  Periodically, each cluster system writes a timestamp and system status (**UP** or **DOWN**) to the primary and backup quorum partitions, which are raw partitions located on shared storage. Each cluster system reads the system status and timestamp that were written by the other cluster system and determines if they are up to date. The cluster systems attempt to read the information from the primary quorum partition. If this partition is corrupted, the cluster systems read the information from the backup quorum partition and simultaneously repair the primary partition. Data consistency is maintained through checksums and any inconsistencies between the partitions are automatically corrected.

  If a cluster system reboots but cannot write to both quorum partitions, the system will not be allowed to join the cluster. In addition, if an existing cluster system can no longer write to both partitions, it removes itself from the cluster by shutting down.

- Remote power switch monitoring

  Periodically, each cluster system monitors the health of the remote power switch connection, if any. The cluster system uses this information to help determine the status of the other cluster system. The complete failure of the power switch communication mechanism does not automatically result in a failover.

- Ethernet and serial heartbeats

  The cluster systems are connected together by using point-to-point Ethernet and serial lines. Periodically, each cluster system issues heartbeats (`ping`s) across these lines. The cluster uses this information to help determine the status of the systems and to ensure correct cluster operation.

The complete failure of the heartbeat communication mechanism does not automatically result in a failover.

If a cluster system determines that the quorum timestamp from the other cluster system is not up-to-date, it will check the heartbeat status. If heartbeats to the system are still operating, the cluster will take no action at this time. If a cluster system does not update its timestamp after some period of time, and does not respond to heartbeat `ping`s, it is considered down.

Note that the cluster will remain operational as long as one cluster system can write to the quorum disk partitions, even if all other communication mechanisms fail.

B.2 Cluster Daemons

The cluster daemons are as follows:

- Quorum daemon

  On each cluster system, the `cluquorumd` quorum daemon periodically writes a timestamp and system status to a specific area on the primary and backup quorum disk partitions. The daemon also reads the other cluster system's timestamp and system status information from the primary quorum partition or, if the primary partition is corrupted, from the backup partition.

- Heartbeat daemon

  On each cluster system, the cluhbd heartbeat daemon issues pings across the point-to-point Ethernet and serial lines to which both cluster systems are connected.

- Power daemon

  On each cluster system, the clupowerd power daemon monitors the remote power switch connection, if any. You will notice that there are two separate clupowerd processes running. One is the *master* process which responds to message requests (e.g. status and power cycle); the other process does periodic polling of the power switch status.

- Service manager daemon

  On each cluster system, the clusvcmgrd service manager daemon responds to changes in cluster membership by stopping and starting services. You may notice, at times, that there may be more than one `clusvcmgrd` process running. This occurs due to the fact that `clusvcmgrd` spawns separate processes for *start*, *stop*, and *monitoring* operations.

- System monitoring daemon

  On each cluster system, the `clumibd` and `xmproxyd` daemons respond to cluster monitoring requests. The Red Hat Cluster Manager GUI is the principal user of these services.

B.3 Failover and Recovery Scenarios

Understanding cluster behavior when significant events occur can assist in the proper management of a cluster. Note that cluster behavior depends on whether power switches are employed in the configuration. Power switches enable the cluster to maintain complete data integrity under all failure conditions.

The following sections describe how the system will respond to various failure and error scenarios.

B.3.1 System Hang

In a cluster configuration that uses power switches, if a system hangs, the cluster behaves as follows:

1. The functional cluster system detects that the hung cluster system is not updating its timestamp on the quorum partitions and is not communicating over the heartbeat channels.
2. The functional cluster system power-cycles the hung system. Alternatively, if watchdog timers are in use, a failed system will reboot itself.
3. The functional cluster system restarts any services that were running on the hung system.
4. If the previously hung system reboots, and can join the cluster (that is, the system can write to both quorum partitions), services are re-balanced across the member systems, according to each service's placement policy.

In a cluster configuration that does not use power switches, if a system hangs, the cluster behaves as follows:

1. The functional cluster system detects that the hung cluster system is not updating its timestamp on the quorum partitions and is not communicating over the heartbeat channels.
2. Optionally, if watchdog timers are used, the failed system will reboot itself.
3. The functional cluster system sets the status of the hung system to **DOWN** on the quorum partitions, and then restarts the hung system's services.
4. If the hung system becomes active, it notices that its status is **DOWN**, and initiates a system reboot.

   If the system remains hung, manually power-cycle the hung system in order for it to resume cluster operation.
5. If the previously hung system reboots, and can join the cluster, services are re-balanced across the member systems, according to each service's placement policy.

B.3.2 System Panic

A system panic (crash) is a controlled response to a software-detected error. A panic attempts to return the system to a consistent state by shutting down the system. If a cluster system panics, the following occurs:

1. The functional cluster system detects that the cluster system that is experiencing the panic is not updating its timestamp on the quorum partitions and is not communicating over the heartbeat channels.
2. The cluster system that is experiencing the panic initiates a system shut down and reboot.
3. If power switches are used, the functional cluster system power-cycles the cluster system that is experiencing the panic.
4. The functional cluster system restarts any services that were running on the system that experienced the panic.
5. When the system that experienced the panic reboots, and can join the cluster (that is, the system can write to both quorum partitions), services are re-balanced across the member systems, according to each service's placement policy.

B.3.3 Inaccessible Quorum Partitions

Inaccessible quorum partitions can be caused by the failure of a SCSI (or Fibre Channel) adapter that is connected to the shared disk storage, or by a SCSI cable becoming disconnected to the shared disk storage. If one of these conditions occurs, and the SCSI bus remains terminated, the cluster behaves as follows:

1. The cluster system with the inaccessible quorum partitions notices that it cannot update its timestamp on the quorum partitions and initiates a reboot.
2. If the cluster configuration includes power switches, the functional cluster system power-cycles the rebooting system.
3. The functional cluster system restarts any services that were running on the system with the inaccessible quorum partitions.
4. If the cluster system reboots, and can join the cluster (that is, the system can write to both quorum partitions), services are re-balanced across the member systems, according to each service's placement policy.

B.3.4 Total Network Connection Failure

A total network connection failure occurs when all the heartbeat network connections between the systems fail. This can be caused by one of the following:

- All the heartbeat network cables are disconnected from a system.
- All the serial connections and network interfaces used for heartbeat communication fail.

If a total network connection failure occurs, both systems detect the problem, but they also detect that the SCSI disk connections are still active. Therefore, services remain running on the systems and are not interrupted.

If a total network connection failure occurs, diagnose the problem and then do one of the following:

- If the problem affects only one cluster system, relocate its services to the other system. Then, correct the problem and relocate the services back to the original system.
- Manually stop the services on one cluster system. In this case, services do not automatically fail over to the other system. Instead, restart the services manually on the other system. After the problem is corrected, it is possible to re-balance the services across the systems.
- Shut down one cluster system. In this case, the following occurs:
  1. Services are stopped on the cluster system that is shut down.
  2. The remaining cluster system detects that the system is being shut down.
  3. Any services that were running on the system that was shut down are restarted on the remaining cluster system.
  4. If the system reboots, and can join the cluster (that is, the system can write to both quorum partitions), services are re-balanced across the member systems, according to each service's placement policy.

B.3.5 Remote Power Switch Connection Failure

If a query to a remote power switch connection fails, but both systems continue to have power, there is no change in cluster behavior unless a cluster system attempts to use the failed remote power switch connection to power-cycle the other system. The power daemon will continually log high-priority messages indicating a power switch failure or a loss of connectivity to the power switch (for example, if a cable has been disconnected).

If a cluster system attempts to use a failed remote power switch, services running on the system that experienced the failure are stopped. However, to ensure data integrity, they are not failed over to the other cluster system. Instead, they remain stopped until the hardware failure is corrected.

B.3.6 Quorum Daemon Failure

If a quorum daemon fails on a cluster system, the system is no longer able to monitor the quorum partitions. If power switches are not used in the cluster, this error condition may result in services being run on more than one cluster system, which can cause data corruption.

If a quorum daemon fails, and power switches are used in the cluster, the following occurs:

1. The functional cluster system detects that the cluster system whose quorum daemon has failed is not updating its timestamp on the quorum partitions, although the system is still communicating over the heartbeat channels.
2. After a period of time, the functional cluster system power-cycles the cluster system whose quorum daemon has failed. Alternatively, if watchdog timers are in use, the failed system will reboot itself.
3. The functional cluster system restarts any services that were running on the cluster system whose quorum daemon has failed.
4. If the cluster system reboots and can join the cluster (that is, it can write to the quorum partitions), services are re-balanced across the member systems, according to each service's placement policy.

If a quorum daemon fails, and neither power switches nor watchdog timers are used in the cluster, the following occurs:

1. The functional cluster system detects that the cluster system whose quorum daemon has failed is not updating its timestamp on the quorum partitions, although the system is still communicating over the heartbeat channels.
2. The functional cluster system restarts any services that were running on the cluster system whose quorum daemon has failed. Under the unlikely event of catastrophic failure, both cluster systems may be running services simultaneously, which can cause data corruption.

B.3.7 Heartbeat Daemon Failure

If the heartbeat daemon fails on a cluster system, service failover time will increase because the quorum daemon cannot quickly determine the state of the other cluster system. By itself, a heartbeat daemon failure will not cause a service failover.

B.3.8 Power Daemon Failure

If the power daemon fails on a cluster system and the other cluster system experiences a severe failure (for example, a system panic), the cluster system will not be able to power-cycle the failed system. Instead, the cluster system will continue to run its services, and the services that were running on the failed system will not fail over. Cluster behavior is the same as for a remote power switch connection failure.

B.3.9 Service Manager Daemon Failure

If the service manager daemon fails, services cannot be started or stopped until you restart the service manager daemon or reboot the system. The simplest way to restart the service manager is to first stop the cluster software and then restart it. For example, to stop the service, perform the following command:

```
/sbin/service cluster stop
```

Then, to restart the cluster software, perform the following:

```
/sbin/service cluster start
```

B.3.10 Monitoring Daemon Failure

If the cluster monitoring daemon (`clumibd`) fails, it is not possible to use the cluster GUI to monitor status. Note, to enable the cluster GUI to remotely monitor cluster status from non-cluster systems, enable this compatibility when prompted in cluconfig.

B.4 Cluster Database Fields

A copy of the cluster database is located in the `/etc/opt/cluster/cluster.conf` file. It contains detailed information about the cluster members and services. *Do not* manually edit the configuration file. Instead, use cluster utilities to modify the cluster configuration.

When you run cluconfig, the site-specific information is entered into fields within the *[members]* section of the database. The following is each cluster member field and its subsequent description:

```
start member0
start chan0
  device = serial_port
  type = serial
end chan0
```

Specifies the tty port that is connected to a null model cable for a serial heartbeat channel. For example, the **`serial_port`** could be **`/dev/ttyS1`**.

```
start chan1
  name = interface_name
  type = net
end chan1
```

Specifies the network interface for one Ethernet heartbeat channel. The **`interface_name`** is the host name to which the interface is assigned (for example, **`storage0`**).

```
start chan2
  device = interface_name
  type = net
end chan2
```

Specifies the network interface for a second Ethernet heartbeat channel. The **`interface_name`** is the host name to which the interface is assigned (for example, **`cstorage0`**). This field can specify the point-to-point dedicated heartbeat network.

```
id = id
name = system_name
```

Specifies the identification number (either 0 or 1) for the cluster system and the name that is returned by the `hostname` command (for example, **`storage0`**).

```
powerSerialPort = serial_port
```

Specifies the device special file for the serial port to which the power switches are connected, if any (for example, **`/dev/ttyS0`**).

```
powerSwitchType = power_switch
```

Specifies the power switch type, either **`RPS10`**, **`APC`**, or **`None`**.

```
quorumPartitionPrimary = raw_disk
quorumPartitionShadow = raw_disk

end member0
```

Specifies the raw devices for the primary and backup quorum partitions (for example, **`/dev/raw/raw1`** and **`/dev/raw/raw2`**).

When you add a cluster service, the service-specific information you specify is entered into the fields within the *[services]* section in the database. The following details each cluster service field and its subsequent description:

```
start service0
name = service_name
disabled = yes_or_no
userScript = path_name
```

Specifies the name of the service, whether the service should be disabled after it is created, and the full path name of any script used to start and stop the service.

```
preferredNode = member_name
relocateOnPreferredNodeBoot = yes_or_no
```

Specifies the name of the cluster system on which you prefer to run the service, and whether the service should relocate to that system when it reboots and joins the cluster.

```
start network0
ipAddress = aaa.bbb.ccc.ddd
netmask = aaa.bbb.ccc.ddd
broadcast = aaa.bbb.ccc.ddd
end network0
```

Specifies the IP address, if any, and accompanying netmask and broadcast addresses used by the service. Note that is is possible to specify multiple IP addresses for a service.

```
start device0
  name = device_file
```

Specifies the special device file, if any, that is used in the service (for example, **/dev/sda1**). Note that it is possible to specify multiple device files for a service.

```
  start mount
  name = mount_point
  fstype = file_system_type
  options = mount_options
  forceUnmount = yes_or_no
```

Specifies the directory mount point, if any, for the device, the type of file system, the mount options, and whether forced unmount is enabled for the mount point.

```
owner = user_name
 group = group_name
 mode = access_mode
end device0
end service0
```

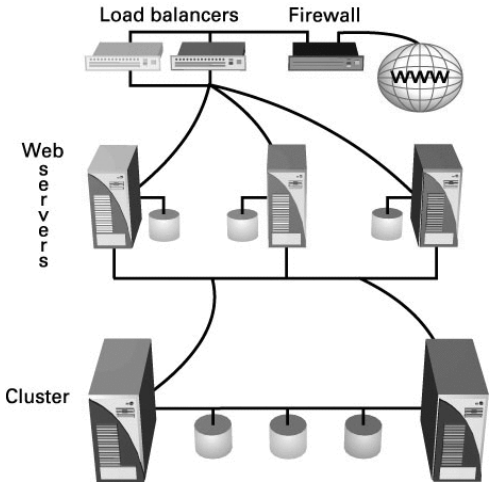
Specifies the owner of the device, the group to which the device belongs, and the access mode for the device.

B.5 Using Red Hat Cluster Manager with Piranha

A cluster can be used in conjunction with the Piranha load-balancing features to deploy a highly available e-commerce site that has complete data integrity and application availability, in addition to load balancing capabilities.

Figure B–1, *Cluster in an LVS Environment* shows how you could use Red Hat Cluster Manager with Piranha. The figure shows a cluster with a three-tier architecture, where the top tier consists of Piranha load-balancing systems to distribute Web requests. The second tier consists of a set of Web servers to serve the requests. The third tier consists of a cluster to serve data to the Web servers.

Figure B–1 Cluster in an LVS Environment



In a Piranha configuration, client systems issue requests on the World Wide Web. For security reasons, these requests enter a Web site through a firewall, which can be a Linux system serving in that capacity or a dedicated firewall device. For redundancy, you can configure firewall devices in a failover configuration. Behind the firewall are Piranha load-balancing systems, which can be configured in an active-standby mode. The active load-balancing system forwards the requests to a set of Web servers.

Each Web server can independently process an HTTP request from a client and send the response back to the client. Piranha enables an administrator to expand a Web site's capacity by adding Web servers to the load-balancing systems' set of active Web servers. In addition, if a Web server fails, it can be removed from the set.

This Piranha configuration is particularly suitable if the Web servers serve only static Web content, which consists of small amounts of infrequently changing data, such as corporate logos, that can be easily duplicated on the Web servers. However, this configuration is not suitable if the Web servers serve dynamic content, which consists of information that changes frequently. Dynamic content could include a product inventory, purchase orders, or customer database, which must be consistent on all the Web servers to ensure that customers have access to up-to-date and accurate information.

To serve dynamic Web content in a Piranha configuration, add a cluster behind the Web servers, as shown in the previous figure. This combination of Piranha and Red Hat Cluster Manager allows for the configuration of a high-integrity, no-single-point-of-failure e-commerce site. The cluster can run a highly-available instance of a database or a set of databases that are network-accessible to the Web servers.

For example, the figure could represent an e-commerce site used for online merchandise ordering through a URL. Client requests to the URL pass through the firewall to the active Piranha load-balancing system, which then forwards the requests to one of the three Web servers. The Red Hat Cluster Manager systems serve dynamic data to the Web servers, which forward the data to the requesting client system.

Index

A

C

D

E

F

H

I

J

K

Q

R

S

T

U

W