Linux FailSafe™ Administrator's Guide

October 05, 2000

**Written by Joshua Rodman of SuSE, Inc.
and Steven Levine and Jenn Byrnes of SGI**

Illustrated by Dany Galgani and Chris Wengelski

Production by Adrian Daley, Glen Traefald

**Engineering contributions by
Scott Henry, Daniel Hurtubise,
Vidula Iyer, Ashwinee Khaladkar, Herbert Lewis,
Michael Nishimoto, Wesley Smith, Bill Sparks, Paddy Sreenivasan,
Dan Stekloff, Rebecca Underwood, Mayank Vasa, Manish Verma**

NOTICE

Contact information: Silicon Graphics, Inc., 1600 Amphitheatre Pkwy, Mountain View, CA 94043, or:

`http://www.sgi.com`

LIMITED RIGHTS LEGEND

TRADEMARKS

007-4322-001 August 2000
Original publication
007-4322-002 October 2000
Clarifications

Contents

About This Guide

This guide describes the configuration and administration of a Linux FailSafe™ highly available system.

This guide was prepared in conjunction with Release 1.0 of the Linux FailSafe product.

Audience

The *Linux FailSafe Administrator's Guide* is written for the person who administers the Linux FailSafe system. The Linux FailSafe administrator must be familiar with the operation of the appropriate storage subsystem configurations, such as the configuration of any raid systems or fibre channel systems which will be used in the Linux FailSafe configuration. Good knowledge of mirroring, the filesystems used, and any volume management system to be used is also required.

Structure of This Guide

Linux FailSafe configuration and administration information is presented in the following chapters and appendices:

- Chapter 1, *Overview of the Linux FailSafe System*, introduces the components of the FailSafe system and explains its hardware and software architecture.
- Chapter 2, *Planning Linux FailSafe Configuration*, describes how to plan the configuration of a FailSafe cluster.
- Chapter 3, *Installing Linux FailSafe Software and Preparing the System*, describes several procedures that must be performed on nodes in a Linux FailSafe cluster to prepare them for high availability setup.
- Chapter 4, *Linux FailSafe Administration Tools*, describes the cluster manager tools you can use to administer a FailSafe system.
- Chapter 5, *Linux FailSafe Cluster Configuration*, explains how to perform the administrative tasks to configure a FailSafe system.
- Chapter 7, *Linux FailSafe System Operation*, explains how to perform the administrative tasks to operate and monitor a FailSafe system.
- Chapter 8, *Testing Linux FailSafe Configuration*, describes how to test the configured FailSafe system.
- Chapter 9, *Linux FailSafe Recovery*, describes the log files used by FailSafe and how to evaluate problems in a FailSafe system.
- Chapter 10, *Upgrading and Maintaining Active Clusters*, describes some procedures you may need to perform without shutting down a FailSafe cluster.

Related Documentation

Besides this guide, other documentation for the Linux FailSafe system includes the following

- *Linux FailSafe Programmer's Guide*

System man pages for referenced commands are as follows:

- `cbeutil`
- `cdbBackup`
- `cdbRestore`
- `cdbutil`
- `cluster_mgr`
- `crsd`
- `cdbd`
- `ha_cilog`
- `ha_cmsd`
- `ha_exec2`
- `ha_fsd`
- `ha_gcd`
- `ha_ifd`
- `ha_ifdadmin`
- `ha_macconfig2`
- `ha_srmd`
- `ha_statd2`
- `haStatus`
- `failsafe`

Conventions Used in This Guide

These type conventions and symbols are used in this guide:

`command`

Function names, literal command-line arguments (options/flags)

`filename`

Name of a file or directory

`command -o option`

Commands and text that you are to type literally in response to shell and command prompts

term

New terms

Book Title

Manual or book title

variable

Command-line arguments, filenames, and variables to be supplied by the user in examples, code, and syntax statements

`literal text`

Code examples, error messages, prompts, and screen text

`#`

System shell prompt for the superuser (`root`)

1 Overview of the Linux FailSafe System

This chapter provides an overview of the components and operation of the Linux FailSafe system. It contains these major sections:

- Section 1.1, *High Availability and Linux FailSafe*
- Section 1.2, *Concepts*
- Section 1.3, *Additional Linux FailSafe Features*
- Section 1.4, *Linux FailSafe Administration*
- Section 1.5, *Hardware Components of a Linux FailSafe Cluster*
- Section 1.6, *Linux FailSafe Disk Connections*
- Section 1.7, *Linux FailSafe Supported Configurations*
- Section 1.8, *Highly Available Resources*
- Section 1.9, *Highly Available Applications*
- Section 1.10, *Failover and Recovery Processes*
- Section 1.11, *Overview of Configuring and Testing a New Linux FailSafe Cluster*

1.1 High Availability and Linux FailSafe

In the world of mission critical computing, the availability of information and computing resources is extremely important. The availability of a system is affected by how long it is unavailable after a failure in any of its components. Different degrees of availability are provided by different types of systems:

- Fault-tolerant systems (continuous availability). These systems use redundant components and specialized logic to ensure continuous operation and to provide complete data integrity. On these systems the degree of availability is extremely high. Some of these systems can also tolerate outages due to hardware or software upgrades (continuous availability). This solution is very expensive and requires specialized hardware and software.
- Highly available systems. These systems survive single points of failure by using redundant off-the-shelf components and specialized software. They provide a lower degree of availability than the fault-tolerant systems, but at much lower cost. Typically these systems provide high availability only for client/server applications, and base their redundancy on cluster architectures with shared resources.

The Silicon Graphics® Linux FailSafe product provides a general facility for providing highly available services. Linux FailSafe provides highly available services for a cluster that contains multiple nodes (*N*-node configuration). Using Linux FailSafe, you can configure a highly available system in any of the following topologies:

- Basic two-node configuration
- Ring configuration
- Star configuration, in which multiple applications running on multiple nodes are backed up by one node
- Symmetric pool configuration

These configurations provide redundancy of processors and I/O controllers. Redundancy of storage can either be obtained through the use of multi-hosted RAID disk devices and mirrored disks, or with redundant disk systems which are kept in synchronization.

If one of the nodes in the cluster or one of the nodes' components fails, a different node in the cluster restarts the highly available services of the failed node. To clients, the services on the replacement node are indistinguishable from the original services before failure occurred. It appears as if the original node has crashed and rebooted quickly. The clients notice only a brief interruption in the highly available service.

In a Linux FailSafe highly available system, nodes can serve as backup for other nodes. Unlike the backup resources in a fault-tolerant system, which serve purely as redundant hardware for backup in case of failure, the resources of each node in a highly available system can be used during normal operation to run other applications that are not necessarily highly available services. All highly available services are owned and accessed by one node at a time.

Highly available services are monitored by the Linux FailSafe software. During normal operation, if a failure is detected on any of these components, a **failover** process is initiated. Using Linux FailSafe, you can define a failover policy to establish which node will take over the services under what conditions. This process consists of resetting the failed node (to ensure data consistency), doing any recovery required by the failed over services, and quickly restarting the services on the node that will take them over.

Linux FailSafe supports **selective failover** in which individual highly available applications can be failed over to a backup node independent of the other highly available applications on that node.

Linux FailSafe highly available services fall into two groups: highly available resources and highly available applications. Highly available resources include network interfaces, logical volumes, and filesystems such as ext2f or reiserfs that have been configured for Linux FailSafe. Silicon Graphics has also developed Linux FailSafe NFS. Highly available applications can include applications such as NFS, Apache, etc.

Linux FailSafe provides a framework for making additional applications into highly available services. If you want to add highly available applications on a Linux FailSafe cluster, you must write scripts to handle application monitoring functions. Information on developing these scripts is described in the *Linux FailSafe Programmer's Guide*. If you need assistance in this regard, contact SGI Global Services, which offers custom Linux FailSafe agent development and HA integration services.

1.2 Concepts

In order to use Linux FailSafe, you must understand the concepts in this section.

1.2.1 Cluster Node (or Node)

A **cluster node** is a single Linux execution environment. In other words, a single physical or virtual machine. In current Linux environments this will always be an individual computer. The term **node** is used to indicate this meaning in this guide for brevity, as opposed to any meaning such as a network node.

1.2.2 Pool

A **pool** is the entire set of nodes having membership in a group of clusters. The clusters are usually close together and should always serve a common purpose. A replicated cluster configuration database is stored on each node in the pool.

1.2.3 Cluster

A **cluster** is a collection of one or more nodes coupled to each other by networks or other similar interconnections. A cluster belongs to one pool and only one pool. A cluster is identified by a simple name; this name must be unique within the pool. A particular node may be a member of only one cluster. All nodes in a cluster are also in the pool; however, all nodes in the pool are not necessarily in the cluster.

1.2.4 Node Membership

A **node membership** is the list of nodes in a cluster on which Linux FailSafe can allocate resource groups.

1.2.5 Process Membership

A **process membership** is the list of process instances in a cluster that form a process group. There can be multiple process groups per node.

1.2.6 Resource

A **resource** is a single physical or logical entity that provides a service to clients or other resources. For example, a resource can be a single disk volume, a particular network address, or an application such as a web server. A resource is generally available for use over time on two or more nodes in a cluster, although it can only be allocated to one node at any given time.

Resources are identified by a resource name and a resource type. One resource can be dependent on one or more other resources; if so, it will not be able to start (that is, be made available for use) unless the dependent resources are also started. Dependent resources must be part of the same resource group and are identified in a resource dependency list.

1.2.7 Resource Type

A **resource type** is a particular class of resource. All of the resources in a particular resource type can be handled in the same way for the purposes of failover. Every resource is an instance of exactly one resource type.

A resource type is identified by a simple name; this name should be unique within the cluster. A resource type can be defined for a specific node, or it can be defined for an entire cluster. A resource type definition for a specific node overrides a clusterwide resource type definition with the same name; this allows an individual node to override global settings from a clusterwide resource type definition.

Like resources, a resource type can be dependent on one or more other resource types. If such a dependency exists, at least one instance of each of the dependent resource types must be defined. For example, a resource type named `Netscape_web` might have resource type dependencies on resource types named `IP_address` and `volume`. If a resource named `web1` is defined

with the Netscape_web resource type, then the resource group containing web1 must also contain at least one resource of the type IP_address and one resource of the type volume.

The Linux FailSafe software includes some predefined resource types. If these types fit the application you want to make highly available, you can reuse them. If none fit, you can create additional resource types by using the instructions in the *Linux FailSafe Programmer's Guide*.

1.2.8 Resource Name

A **resource name** identifies a specific instance of a resource type. A resource name must be unique for a given resource type.

1.2.9 Resource Group

A **resource group** is a collection of interdependent resources. A resource group is identified by a simple name; this name must be unique within a cluster. Table 1–1, *Example Resource Group* shows an example of the resources and their corresponding resource types for a resource group named WebGroup.

Table 1–1 Example Resource Group

Resource	Resource Type
10.10.48.22	IP_address
/fs1	filesystem
vol1	volume
web1	Netscape_web

If any individual resource in a resource group becomes unavailable for its intended use, then the entire resource group is considered unavailable. Therefore, a resource group is the unit of failover.

Resource groups cannot overlap; that is, two resource groups cannot contain the same resource.

1.2.10 Resource Dependency List

A **resource dependency list** is a list of resources upon which a resource depends. Each resource instance must have resource dependencies that satisfy its resource type dependencies before it can be added to a resource group.

1.2.11 Resource Type Dependency List

A **resource type dependency list** is a list of resource types upon which a resource type depends. For example, the filesystem resource type depends upon the volume resource type, and the Netscape_web resource type depends upon the filesystem and IP_address resource types.

For example, suppose a file system instance fs1 is mounted on volume vol1. Before fs1 can be added to a resource group, fs1 must be defined to depend on vol1. Linux FailSafe only knows that a file system instance must have one volume instance in its dependency list. This requirement is inferred from the resource type dependency list.

1.2.12 Failover

A **failover** is the process of allocating a resource group (or application) to another node, according to a failover policy. A failover may be triggered by the failure of a resource, a change in the node membership (such as when a node fails or starts), or a manual request by the administrator.

1.2.13 Failover Policy

A **failover policy** is the method used by Linux FailSafe to determine the destination node of a failover. A failover policy consists of the following:

- Failover domain
- Failover attributes
- Failover script

Linux FailSafe uses the failover domain output from a failover script along with failover attributes to determine on which node a resource group should reside.

The administrator must configure a failover policy for each resource group. A failover policy name must be unique within the pool. Linux FailSafe includes predefined failover policies, but you can define your own failover algorithms as well.

1.2.14 Failover Domain

A **failover domain** is the ordered list of nodes on which a given resource group can be allocated. The nodes listed in the failover domain must be within the same cluster; however, the failover domain does not have to include every node in the cluster.

The administrator defines the initial failover domain when creating a failover policy. This list is transformed into a run-time failover domain by the failover script; Linux FailSafe uses the run-time failover domain along with failover attributes and the node membership to determine the node on which a resource group should reside. Linux FailSafe stores the run-time failover domain and uses it as input to the next failover script invocation. Depending on the run-time conditions and contents of the failover script, the initial and run-time failover domains may be identical.

In general, Linux FailSafe allocates a given resource group to the first node listed in the run-time failover domain that is also in the node membership; the point at which this allocation takes place is affected by the failover attributes.

1.2.15 Failover Attribute

A **failover attribute** is a string that affects the allocation of a resource group in a cluster. The administrator must specify system attributes (such as `Auto_Failback` or `Controlled_Failback`), and can optionally supply site-specific attributes.

1.2.16 Failover Scripts

A **failover script** is a shell script that generates a run-time failover domain and returns it to the Linux FailSafe process. The Linux FailSafe process `ha_fsd` applies the failover attributes and then selects the first node in the returned failover domain that is also in the current node membership.

The following failover scripts are provided with the Linux FailSafe release:

- `ordered`, which never changes the initial failover domain. When using this script, the initial and run-time failover domains are equivalent.
- `round-robin`, which selects the resource group owner in a round-robin (circular) fashion. This policy can be used for resource groups that can be run in any node in the cluster.

If these scripts do not meet your needs, you can create a new failover script using the information in this guide.

1.2.17 Action Scripts

The **action scripts** are the set of scripts that determine how a resource is started, monitored, and stopped. There must be a set of action scripts specified for each resource type.

The following is the complete set of action scripts that can be specified for each resource type:

- `exclusive`, which verifies that a resource is not already running
- `start`, which starts a resource
- `stop`, which stops a resource
- `monitor`, which monitors a resource
- `restart`, which restarts a resource on the same server after a monitoring failure occurs

The release includes action scripts for predefined resource types. If these scripts fit the resource type that you want to make highly available, you can reuse them by copying them and modifying them as needed. If none fits, you can create additional action scripts by using the instructions in the *Linux FailSafe Programmer's Guide*.

1.3 Additional Linux FailSafe Features

Linux FailSafe provides the following features to increase the flexibility and ease of operation of a highly available system:

- Dynamic management
- Fine grain failover
- Local restarts

These features are summarized in the following sections.

1.3.1 Dynamic Management

Linux FailSafe allows you to perform a variety of administrative tasks while the system is running:

- Dynamically managed application monitoring

  Linux FailSafe allows you to turn monitoring of an application on and off while other highly available applications continue to run. This allows you to perform online application upgrades without bringing down the Linux FailSafe system.

- Dynamically managed Linux FailSafe resources

Linux FailSafe allows you to add resources while the Linux FailSafe system is online.

- Dynamically managed Linux FailSafe upgrades

  Linux FailSafe allows you to upgrade Linux FailSafe software on one node at a time without taking down the entire Linux FailSafe cluster.

1.3.2 Fine Grain Failover

Using Linux FailSafe, you can specify **fine-grain failover**. Fine-grain failover is a process in which a specific resource group is failed over from one node to another node while other resource groups continue to run on the first node, where possible. Fine-grain failover is possible in Linux FailSafe because the unit of failover is the resource group, and not the entire node.

1.3.3 Local Restarts

Linux FailSafe allows you to fail over a resource group onto the same node. This feature enables you to configure a single-node system, where backup for a particular application is provided on the same machine, if possible. It also enables you to indicate that a specified number of local restarts be attempted before the resource group fails over to a different node.

1.4 Linux FailSafe Administration

You can perform all Linux FailSafe administrative tasks by means of the Linux FailSafe Cluster Manager Graphical User Interface (GUI). The Linux FailSafe GUI provides a guided interface to configure, administer, and monitor a Linux FailSafe-controlled highly available cluster. The Linux FailSafe GUI also provides screen-by-screen help text.

If you wish, you can perform Linux FailSafe administrative tasks directly by means of the Linux FailSafe Cluster Manager CLI, which provides a command-line interface for the administration tasks.

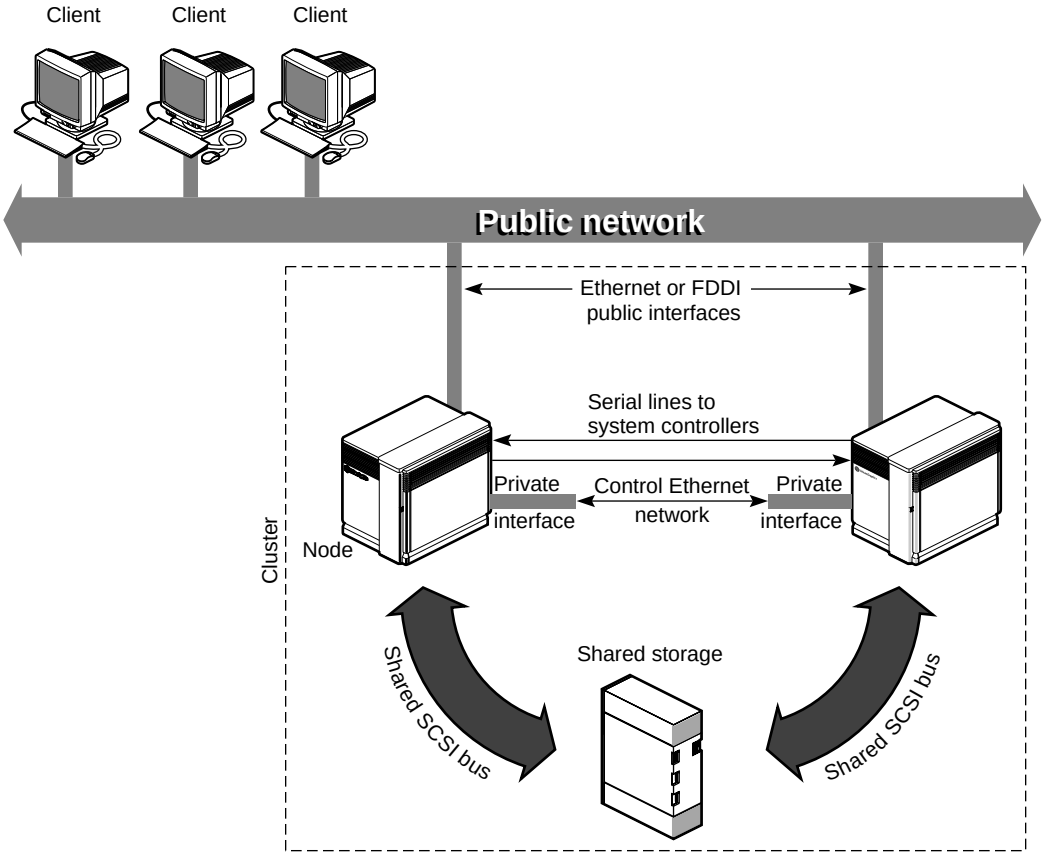
For information on Linux FailSafe Cluster Manager tools, see Chapter 4, *Linux FailSafe Administration Tools*.

For information on Linux FailSafe configuration and administration tasks, see Chapter 5, *Linux FailSafe Cluster Configuration*, and Chapter 7, *Linux FailSafe System Operation*.

1.5 Hardware Components of a Linux FailSafe Cluster

Figure 1–1, *Sample Linux FailSafe System Components*, shows an example of Linux FailSafe hardware components, in this case for a two-node system.

Figure 1–1 Sample Linux FailSafe System Components



The hardware components of the Linux FailSafe system are as follows:

- Up to eight Linux nodes
- Two or more interfaces on each node to control networks (Ethernet, FDDI, or any other available network interface)

  At least two network interfaces on each node are required for the control network **heartbeat** connection, by which each node monitors the state of other nodes. The Linux FailSafe software also uses this connection to pass **control** messages between nodes. These interfaces have distinct IP addresses.
- A mechanism for remote reset of nodes

  A reset ensures that the failed node is not using the shared disks when the replacement node takes them over.
- Disk storage and SCSI bus shared by the nodes in the cluster

  The nodes in the Linux FailSafe system can share dual-hosted disk storage over a shared fast and wide SCSI bus where this is supported by the SCSI controller and Linux driver.

Note that few Linux drivers are currently known to implement this correctly. Please check hardware compatibility lists if this is a configuration you plan to use. Fibre Channel solutions should universally support this.

The Linux FailSafe system is designed to survive a single point of failure. Therefore, when a system component fails, it must be restarted, repaired, or replaced as soon as possible to avoid the possibility of two or more failed components.

1.6 Linux FailSafe Disk Connections

A Linux FailSafe system supports the following disk connections:

- RAID support
  - Single controller or dual controllers
  - Single or dual hubs
  - Single or dual pathing
- JBOD support
  - Single or dual vaults
  - Single or dual hubs
- Network-mirrored support
  - Clustered filesystems such as GFS
  - Network mirroring block devices such as with DRBD

Network mirrored devices are not discussed in the examples within this guide. However, the Linux FailSafe configuration items that are set for shared storage apply validly to network-duplicated storage.

SCSI disks can be connected to two machines only. Fibre channel disks can be connected to multiple machines.

1.7 Linux FailSafe Supported Configurations

Linux FailSafe supports the following highly available configurations:

- Basic two-node configuration
- Star configuration of multiple primary and 1 backup node

- Ring configuration

You can use the following reset models when configuring a Linux FailSafe system:

- Server-to-server. Each server is directly connected to another for reset. May be unidirectional.
- Network. Each server can reset any other by sending a signal over the control network to a multiplexer.

The following sections provide descriptions of the different Linux FailSafe configurations.

1.7.1 Basic Two-Node Configuration

In a basic two-node configuration, the following arrangements are possible:

- All highly available services run on one node. The other node is the backup node. After failover, the services run on the backup node. In this case, the backup node is a hot standby for failover purposes only. The backup node can run other applications that are not highly available services.
- Highly available services run concurrently on both nodes. For each service, the other node serves as a backup node. For example, both nodes can be exporting different NFS filesystems. If a failover occurs, one node then exports all of the NFS filesystems.

1.8 Highly Available Resources

This section discusses the highly available resources that are provided on a Linux FailSafe system.

1.8.1 Nodes

If a node crashes or hangs (for example, due to a parity error or bus error), the Linux FailSafe software detects this. A different node, determined by the failover policy, takes over the failed node's services after resetting the failed node.

If a node fails, the interfaces, access to storage, and services also become unavailable. See the succeeding sections for descriptions of how the Linux FailSafe system handles or eliminates these points of failure.

1.8.2 Network Interfaces and IP Addresses

Clients access the highly available services provided by the Linux FailSafe cluster using IP addresses. Each highly available service can use multiple IP addresses. The IP addresses are not tied to a particular highly available service; they can be shared by all the highly available services in the cluster.

Linux FailSafe uses the IP aliasing mechanism to support multiple IP addresses on a single network interface. Clients can use a highly available service that uses multiple IP addresses even when there is only one network interface in the server node.

The IP aliasing mechanism allows a Linux FailSafe configuration that has a node with multiple network interfaces to be backed up by a node with a single network interface. IP addresses configured on multiple network interfaces are moved to the single interface on the other node in case of a failure.

Linux FailSafe requires that each network interface in a cluster have an IP address that does not failover. These IP addresses, called **fixed IP addresses**, are used to monitor network interfaces.

Each fixed IP address must be configured to a network interface at system boot up time. All other IP addresses in the cluster are configured as **highly available IP addresses**.

Highly available IP addresses are configured on a network interface. During failover and recovery processes they are moved to another network interface in the other node by Linux FailSafe. Highly available IP addresses are specified when you configure the Linux FailSafe system. Linux FailSafe uses the `ifconfig` command to configure an IP address on a network interface and to move IP addresses from one interface to another.

In some networking implementations, IP addresses cannot be moved from one interface to another by using only the `ifconfig` command. Linux FailSafe uses **re-MACing** (**MAC address impersonation**) to support these networking implementations. Re-MACing moves the physical (MAC) address of a network interface to another interface. It is done by using the `macconfig` command. Re-MACing is done in addition to the standard `ifconfig` process that Linux FailSafe uses to move IP addresses. To do RE-MACing in Linux FailSafe, a resource of type MAC_Address is used.

Re-MACing can be used only on Ethernet networks. It cannot be used on FDDI networks.

Re-MACing is required when packets called gratuitous ARP packets are not passed through the network. These packets are generated automatically when an IP address is added to an interface (as in a failover process). They announce a new mapping of an IP address to MAC address. This tells clients on the local subnet that a particular interface now has a particular IP address. Clients then update their internal ARP caches with the new MAC address for the IP address. (The IP address just moved from interface to interface.) When gratuitous ARP packets are not passed through the network, the internal ARP caches of subnet clients cannot be updated. In these cases, re-MACing is used. This moves the MAC address of the original interface to the new interface. Thus, both the IP address and the MAC address are moved to the new interface and the internal ARP caches of clients do not need updating.

Re-MACing is not done by default; you must specify that it be done for each pair of primary and secondary interfaces that requires it. A procedure in the section Section 2.5.1, *Planning Network Interface and IP Address Configuration* describes how you can determine whether re-MACing is required. In general, routers and PC/NFS clients may require re-MACing interfaces.

A side effect of re-MACing is that the original MAC address of an interface that has received a new MAC address is no longer available for use. Because of this, each network interface has to be backed up by a dedicated backup interface. This backup interface cannot be used by clients as a primary interface. (After a failover to this interface, packets sent to the original MAC address are ignored by every node on the network.) Each backup interface backs up only one network interface.

1.8.3 Disks

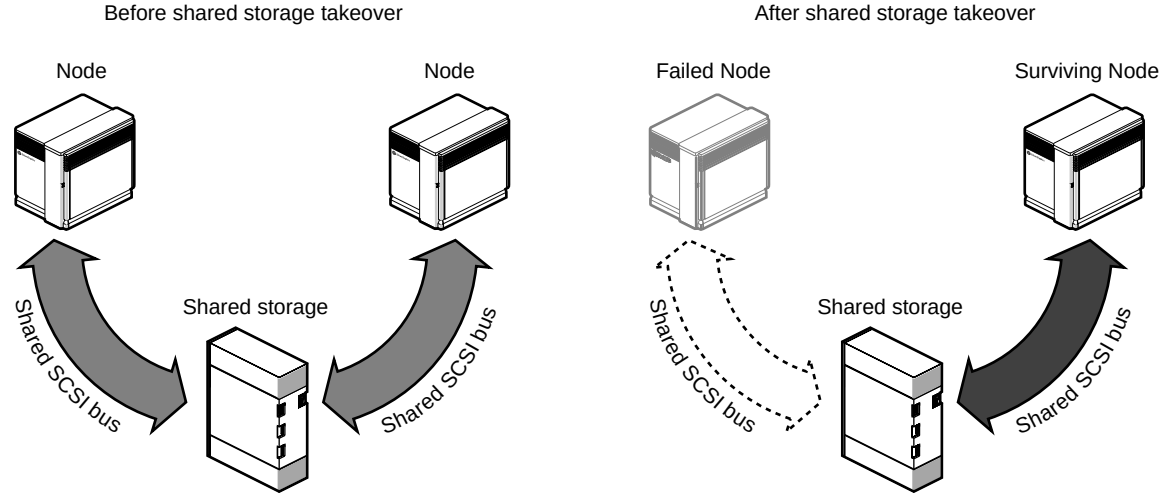
The Linux FailSafe cluster can include shared SCSI-based storage in the form of individual disks, RAID systems, or Fibre Channel storage systems.

With mirrored volumes on the disks in a RAID or Fibre Channel system, the device system should provide redundancy. No participation of the Linux FailSafe system software is required

for a disk failure. If a disk controller fails, the Linux FailSafe system software initiates the failover process.

Figure 1–2, *Disk Storage Failover on a Two-Node System*, shows disk storage takeover on a two-node system. The surviving node takes over the shared disks and recovers the logical volumes and filesystems on the disks. This process is expedited by a filesystem such as ReiserFS or XFS, because of journaling technology that does not require the use of the `fsck` command for filesystem consistency checking.

Figure 1–2 Disk Storage Failover on a Two-Node System



1.9 Highly Available Applications

Each application has a primary node and up to seven additional nodes that you can use as a backup node, according to the failover policy you define. The primary node is the node on which the application runs when Linux FailSafe is in **normal state**. When a failure of any highly available resources or highly available application is detected by Linux FailSafe software, all highly available resources in the affected resource group on the failed node are failed over to a different node and the highly available applications on the failed node are stopped. When these operations are complete, the highly available applications are started on the backup node.

All information about highly available applications, including the primary node, components of the resource group, and failover policy for the application and monitoring, is specified when you configure your Linux FailSafe system with the Cluster Manager GUI or with the Cluster Manager CLI. Information on configuring the system is provided in Chapter 5, *Linux FailSafe Cluster Configuration*. Monitoring scripts detect the failure of a highly available application.

The Linux FailSafe software provides a framework for making applications highly available services. By writing scripts and configuring the system in accordance with those scripts, you can turn client/server applications into highly available applications. For information, see the *Linux FailSafe Programmer's Guide*.

1.10 Failover and Recovery Processes

When a failure is detected on one node (the node has crashed, hung, or been shut down, or a highly available service is no longer operating), a different node performs a failover of the highly

available services that are being provided on the node with the failure (called the **failed node**). Failover allows all of the highly available services, including those provided by the failed node, to remain available within the cluster.

A failure in a highly available service can be detected by Linux FailSafe processes running on another node. Depending on which node detects the failure, the sequence of actions following the failure is different.

If the failure is detected by the Linux FailSafe software running on the same node, the failed node performs these operations:

- Stops the highly available resource group running on the node
- Moves the highly available resource group to a different node, according to the defined failover policy for the resource group
- Sends a message to the node that will take over the services to start providing all resource group services previously provided by the failed node

When it receives the message, the node that is taking over the resource group performs these operations:

- Transfers ownership of the resource group from the failed node to itself
- Starts offering the resource group services that were running on the failed node

If the failure is detected by Linux FailSafe software running on a different node, the node detecting the failure performs these operations:

- Using the serial connection between the nodes, reboots the failed node to prevent corruption of data
- Transfers ownership of the resource group from the failed node to the other nodes in the cluster, based on the resource group failover policy.
- Starts offering the resource group services that were running on the failed node

When a failed node comes back up, whether the node automatically starts to provide highly available services again depends on the failover policy you define. For information on defining failover policies, see Section 5.5.12, *Defining a Failover Policy*.

Normally, a node that experiences a failure automatically reboots and resumes providing highly available services. This scenario works well for transient errors (as well as for planned outages for equipment and software upgrades). However, if there are persistent errors, automatic reboot can cause recovery and an immediate failover again. To prevent this, the Linux FailSafe software checks how long the rebooted node has been up since the last time it was started. If the interval is less than five minutes (by default), the Linux FailSafe software automatically disables Linux FailSafe from booting on the failed node and does not start up the Linux FailSafe software on this node. It also writes error messages to `/var/log/failsafe` and to the appropriate log file.

1.11 Overview of Configuring and Testing a New Linux FailSafe Cluster

After the Linux FailSafe cluster hardware has been installed, follow this general procedure to configure and test the Linux FailSafe system:

1. Become familiar with Linux FailSafe terms by reviewing this chapter.
2. Plan the configuration of highly available applications and services on the cluster using Chapter 2, *Planning Linux FailSafe Configuration*.
3. Perform various administrative tasks, including the installation of prerequisite software, that are required by Linux FailSafe, as described in Chapter 3, *Installing Linux FailSafe Software and Preparing the System*.
4. Define the Linux FailSafe configuration as explained in Chapter 5, *Linux FailSafe Cluster Configuration*.
5. Test the Linux FailSafe system in three phases: test individual components prior to starting Linux FailSafe software, test normal operation of the Linux FailSafe system, and simulate failures to test the operation of the system after a failure occurs.

1.12 Linux FailSafe System Software

This section describes the software layers, communication paths, and cluster configuration database.

1.12.1 Layers

A Linux FailSafe system has the following software layers:

- Plug-ins, which create highly available services. If the application plug-in you want is not available, you can hire the Silicon Graphics Global Services group to develop the required software, or you can use the *Linux FailSafe Programmer's Guide* to write the software yourself.
- Linux FailSafe base, which includes the ability to define resource groups and failover policies
- High-availability cluster infrastructure that lets you define clusters, resources, and resource types (this consists of the `cluster_services` installation package)
- Cluster software infrastructure, which lets you do the following:
  - Perform node logging
  - Administer the cluster
  - Define nodes

  The cluster software infrastructure consists of the `cluster_admin` and `cluster_control` subsystems).

Figure 1–3, *Software Layers* shows a graphic representation of these layers. Table 1–2, *Contents of `/usr/lib/failsafe/bin`* describes the layers for Linux FailSafe, which are located in the `/usr/lib/failsafe/bin` directory.

Figure 1–3 Software Layers

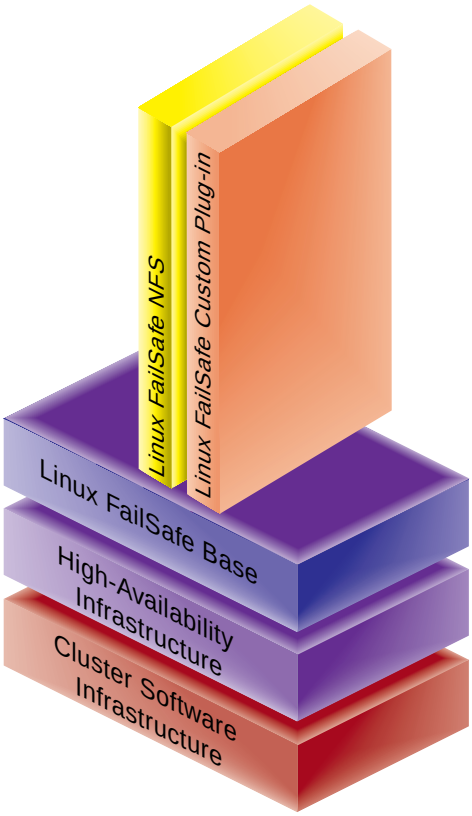


Table 1–2 Contents of `/usr/lib/failsafe/bin`

Layer	Subsystem	Process	Description
Linux FailSafe Base	`failsafe2`	`ha_fsd`	Linux FailSafe daemon. Provides basic component of the Linux FailSafe software.
High-availability cluster infrastructure	`cluster_ha`	`ha_cmsd`	Cluster membership daemon. Provides the list of nodes, called **node membership**, available to the cluster.
		`ha_gcd`	Group membership daemon. Provides group membership and reliable communication services in the presence of failures to Linux FailSafe processes.
		`ha_srmd`	System resource manager daemon. Manages resources, resource groups, and resource types. Executes action scripts for resources.

Layer	Subsystem	Process	Description
		ha_ifd	Interface agent daemon. Monitors the local node's network interfaces.
Cluster software infrastructure	cluster_admin	cad	Cluster administration daemon. Provides administration services.
	cluster_control	crsd	Node control daemon. Monitors the serial connection to other nodes. Has the ability to reset other nodes.
		cmond	Daemon that manages all other daemons. This process starts other processes in all nodes in the cluster and restarts them on failures.
		cdbd	Manages the configuration database and keeps each copy in sync on all nodes in the pool

1.12.2 Communication Paths

The following figures show communication paths in Linux FailSafe. Note that they do not represent cmond.

Figure 1–4 Read/Write Actions to the Cluster Configuration Database

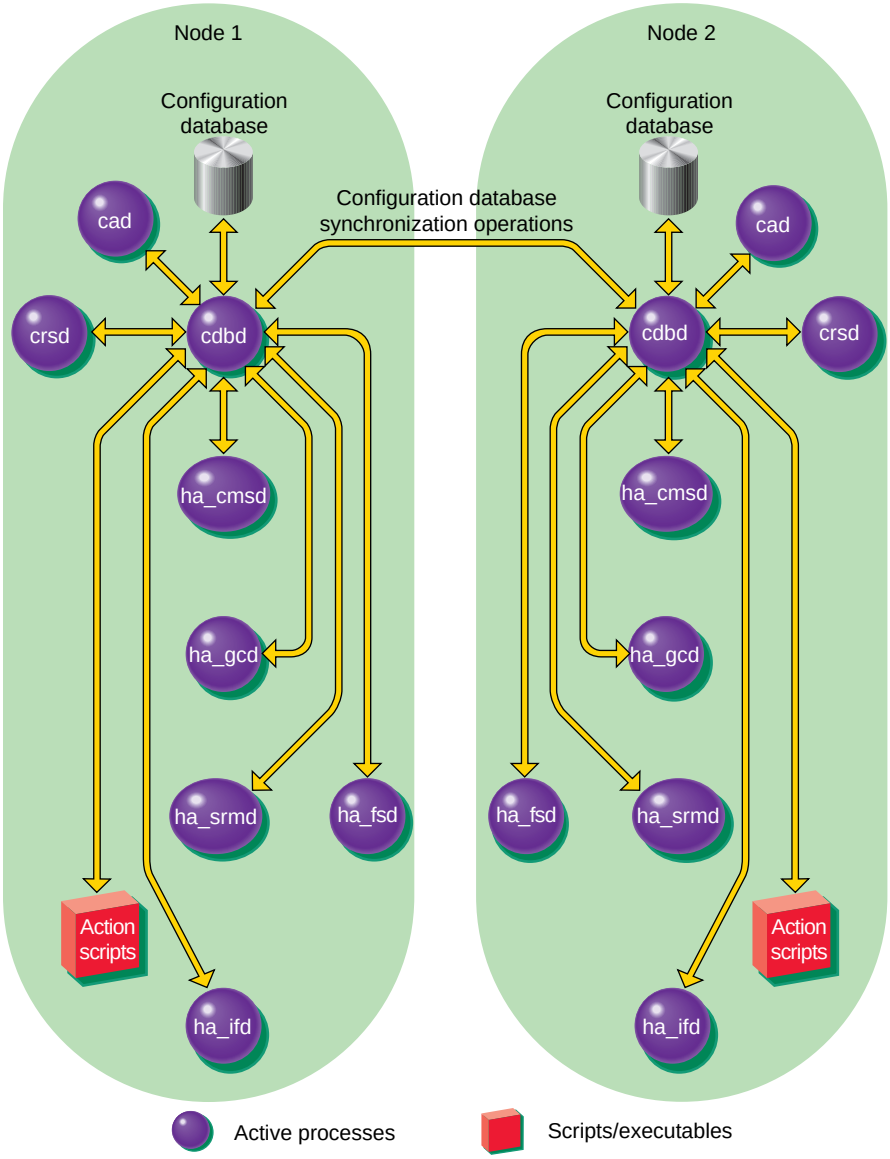
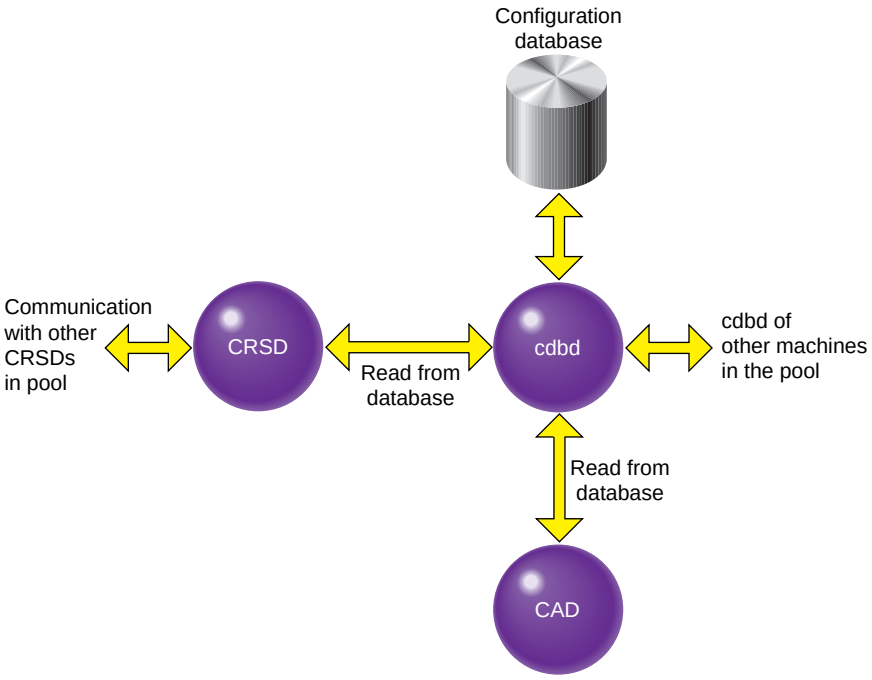


Figure 1–5, *Communication Path for a Node that is Not in a Cluster* shows the communication path for a node that is in the pool but not in a cluster.

Figure 1–5 Communication Path for a Node that is Not in a Cluster



1.12.3 Conditions Under Which Action Scripts are Executed

Action scripts are executed under the following conditions:

- `exclusive`: the resource group is made online by the user or HA processes are started
- `start`: the resource group is made online by the user, HA processes are started, or there is a resource group failover
- `stop`: the resource group is made offline, HA process are stopped, the resource group fails over, or the node is shut down
- `monitor`: the resource group is online
- `restart`: the `monitor` script fails

1.12.4 When Does FailSafe Execute Action and Failover Scripts

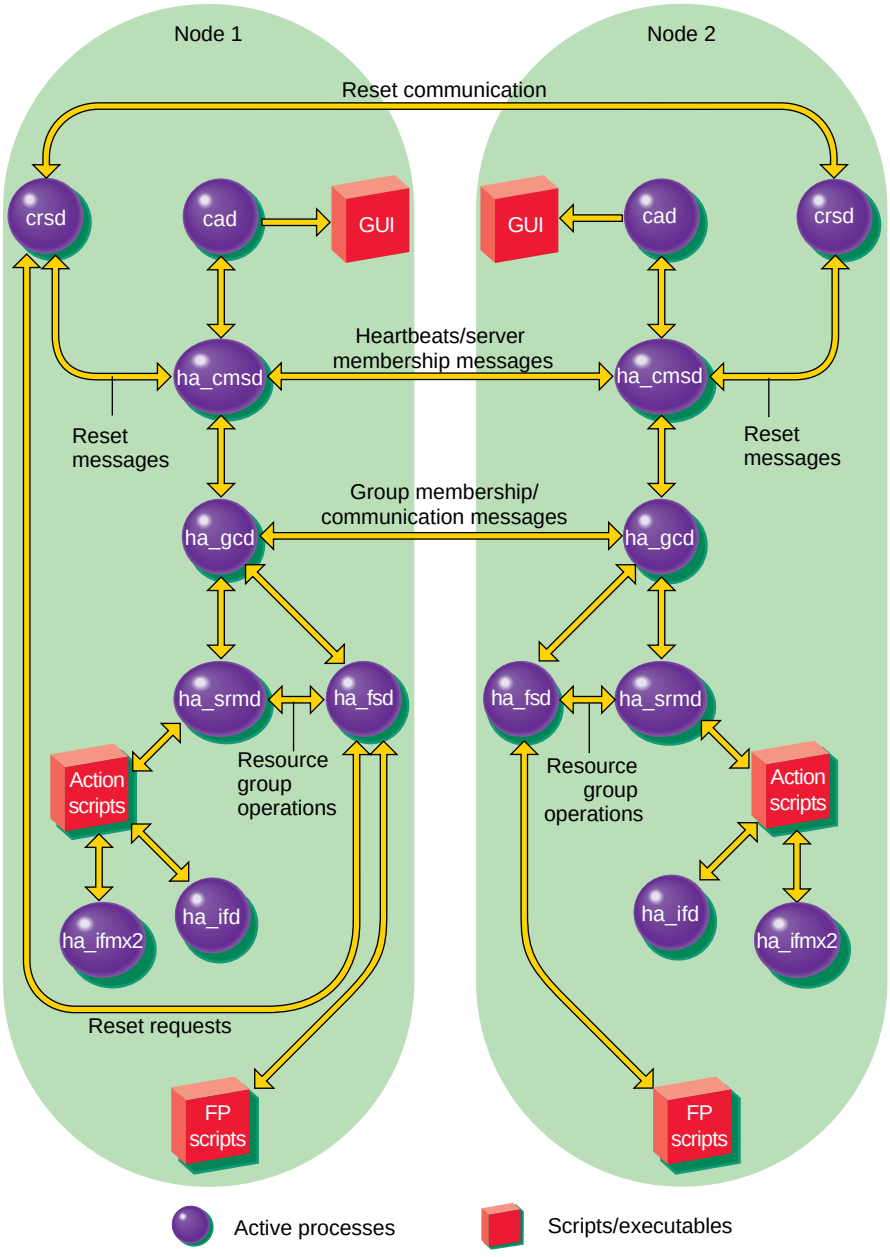
The order of execution is as follows:

1. Linux FailSafe is started, usually at node boot or manually, and reads the resource group information from the cluster configuration database.
2. Linux FailSafe asks the system resource manager (SRM) to run `exclusive` scripts for all resource groups that are in the `Online ready` state.
3. SRM returns one of the following states for each resource group:
   - `running`
   - `partially running`
   - `not running`

4. If a resource group has a state of `not running` in a node where HA services have been started, the following occurs:
   a. Linux FailSafe runs the failover policy script associated with the resource group. The failover policy scripts take the list of nodes that are capable of running the resource group (the **failover domain**) as a parameter.
   b. The failover policy script returns an ordered list of nodes in descending order of priority (the **run-time failover domain**) where the resource group can be placed.
   c. Linux FailSafe sends a request to SRM to move the resource group to the first node in the run-time failover domain.
   d. SRM executes the `start` action script for all resources in the resource group:
      - If the `start` script fails, the resource group is marked `online` on that node with an `srmd executable error` error.
      - If the `start` script is successful, SRM automatically starts monitoring those resources. After the specified start monitoring time passes, SRM executes the `monitor` action script for the resource in the resource group.
5. If the state of the resource group is `running` or `partially running` on only one node in the cluster, Linux FailSafe runs the associated failover policy script:
   - If the highest priority node is the same node where the resource group is partially running or running, the resource group is made online on the same node. In the `partially running` case, Linux FailSafe asks SRM to execute `start` scripts for resources in the resource group that are not running.
   - If the highest priority node is a another node in the cluster, Linux FailSafe asks SRM to execute `stop` action scripts for resources in the resource group. Linux FailSafe makes the resource group online in the highest priority node in the cluster.
6. If the state of the resource group is `running` or `partially running` in multiple nodes in the cluster, the resource group is marked with an `error exclusivity` error. These resource groups will require operator intervention to become online in the cluster.

Figure 1–6, *Message Paths for Action Scripts and Failover Policy Scripts* shows the message paths for action scripts and failover policy scripts.

Figure 1–6 Message Paths for Action Scripts and Failover Policy Scripts



1.12.5 Components

The cluster configuration database is a key component of Linux FailSafe software. It contains all information about the following:

- Resources
- Resource types
- Resource groups
- Failover policies

- Nodes
- Clusters

The cluster configuration database daemon (`cdbd`) maintains identical databases on each node in the cluster.

The following are the contents of the failsafe directories under the `/usr/lib` and `/var` hierarchies:

- `/var/run/failsafe/comm/`

  Directory that contains files that communicate between various daemons.

- `/usr/lib/failsafe/common_scripts/`

  Directory that contains the script library (the common functions that may be used in action scripts).

- `/var/log/failsafe/`

  Directory that contains the logs of all scripts and daemons executed by Linux FailSafe. The outputs and errors from the commands within the scripts are logged in the `script_`*`nodename`* file.

- `/usr/lib/failsafe/policies/`

  Directory that contains the failover scripts used for resource groups.

- `/usr/lib/failsafe/resource_types/template`

  Directory that contains the template action scripts.

- `/usr/lib/failsafe/resource_types/`*`rt_name`*

  Directory that contains the action scripts for the *rt_name* resource type. For example, `/usr/lib/failsafe/resource_types/filesystem` .

- `resource_types/`*`rt_name`*`/exclusive`

  Script that verifies that a resource of this resource type is not already running. For example, `resource_types/filesystem/exclusive`.

- `resource_types/`*`rt_name`*`/monitor`

  Script that monitors a resource of this type.

- `resource_types/`*`rt_name`*`/restart`

  Script that restarts a resource of this resource type on the same node after a monitoring failure.

- `resource_types/`*`rt_name`*`/start`

  Script that starts a resource of this resource type.

- `resource_types/`*`rt_name`*`/stop`

  Script that stops a resource of this resource type.

Table 1–3, *Administrative Commands for Use in Scripts* shows the administrative commands available for use in scripts.

Table 1–3 Administrative Commands for Use in Scripts

Command	Purpose
`ha_cilog`	Logs messages to the `script_` *nodename* log files.
`ha_execute_lock`	Executes a command with a file lock. This allows command execution to be serialized
`ha_exec2`	Executes a command and retries the command on failure or timeout.
`ha_filelock`	Locks a file.
`ha_fileunlock`	Unlocks a file.
`ha_ifdadmin`	Communicates with the `ha_ifd` network interface agent daemon.
`ha_http_ping2`	Checks if a web server is running.
`ha_macconfig2`	Displays or modifies MAC addresses of a network interface.

2 Planning Linux FailSafe Configuration

This chapter explains how to plan the configuration of highly available services on your Linux FailSafe cluster. The major sections of this chapter are as follows:

- Section 2.1, *Introduction to Configuration Planning*
- Section 2.2, *Disk Configuration*
- Section 2.3, *Logical Volume Configuration*
- Section 2.4, *Filesystem Configuration*
- Section 2.5, *IP Address Configuration*

2.1 Introduction to Configuration Planning

Configuration planning involves making decisions about how you plan to use the Linux FailSafe cluster, and based on that, how the disks and interfaces must be set up to meet the needs of the highly available services you want the cluster to provide. Questions you must answer during the planning process are:

- What do you plan to use the nodes for?

  Your answers might include uses such as offering home directories for users, running particular applications, supporting an Oracle database, providing Netscape World Wide Web service, and providing file service.

- Which of these uses will be provided as a highly available service?

  To offer applications as highly available services that are not currently available as Linux Failsafe software options, a set of application monitoring shell scripts needs to be developed that provides switch over and switch back functionality. Developing these scripts is described in the *Linux FailSafe Programmer's Guide*. If you need assistance in this regard, contact SGI Global Services, which offers custom Linux FailSafe agent development and HA integration services.

- Which node will be the primary node for each highly available service?

  The primary node is the node that provides the service (exports the filesystem, is a Netscape server, provides the database, and so on) when the node is in an UP state.

- For each highly available service, how will the software and data be distributed on shared and non-shared disks?

  Each application has requirements and choices for placing its software on disks that are failed over (shared) or not failed over (non-shared).

- Are the shared disks going to be part of a RAID storage system or are they going to be disks in SCSI/Fibre channel disk storage that has mirroring such as the Linux Raid Tools implemented on them?

  For reliability, shared disks must be part of a RAID storage system or in SCSI/Fibre channel disk storage with mirroring on them.

- Will the shared disks be used as raw devices/volumes or as volumes with filesystems on them?

Logical volumes, filesystems, and raw partitions are all supported by Linux Failsafe. The choice of volumes, filesystems, or raw devices depends on the application that is going to use the disk space.

- Which IP addresses will be used by clients of highly available services?

  Multiple interfaces may be required on each node because a node could be connected to more than one network or because there could be more than one interface to a single network.

- Which resources will be part of a resource group?

  All resources that are dependent on each other have to be in the resource group.

- What will be the failover domain of the resource group?

  The failover domain determines the list of nodes in the cluster where the resource group can reside. For example, a volume resource that is part of a resource group can reside only in nodes from which the disks composing the volume can be accessed.

- How many highly available IP addresses on each network interface will be available to clients of the highly available services?

  At least one highly available IP address must be available for each interface on each node that is used by clients of highly available services.

- Which IP addresses on primary nodes are going to be available to clients of the highly available services?

- For each highly available IP address that is available on a primary node to clients of highly available services, which interface on the other nodes will be assigned that IP address after a failover?

  Every highly available IP address used by a highly available service must be mapped to at least one interface in each node that can take over the resource group service. The highly available IP addresses are failed over from the interface in the primary node of the resource group to the interface in the replacement node.

As an example of the configuration planning process, say that you have a two-node Linux FailSafe cluster that is a departmental server. You want to make four XFS filesystems available for NFS mounting and have two Netscape FastTrack servers, each serving a different set of documents. These applications will be highly available services.

You decide to distribute the services across two nodes, so each node will be the primary node for two filesystems and one Netscape server. The filesystems and the document roots for the Netscape servers (on XFS filesystems) are each on their own striped LVM logical volume. The logical volumes are created from disks in a RAID storage system connected to both nodes.

There are four resource groups: NFSgroup1 and NFSgroup2 are the NFS resource groups, and Webgroup1 and Webgroup2 are the Web resource groups. NFSgroup1 and Webgroup1 will have one node as the primary node. NFSgroup2 and Webgroup2 will have the other node as the primary node.

Two networks are available on each node, eth0 and eth1. The eth0 interfaces in each node are connected to each other to form a private network.

The following sections help you answer the configuration questions above, make additional configuration decisions required by Linux FailSafe, and collect the information you need to

perform the configuration tasks described in Chapter 3, *Installing Linux FailSafe Software and Preparing the System*, and Chapter 5, *Linux FailSafe Cluster Configuration*.

2.2 Disk Configuration

The first subsection below describes the disk configuration issues that must be considered when planning a Linux FailSafe system. It explains the basic configurations of shared and non-shared disks and how they are reconfigured by Linux FailSafe after a failover. The second subsection explains how disk configurations are specified when you configure the Linux FailSafe system.

2.2.1 Planning Disk Configuration

For each disk in a Linux FailSafe cluster, you must choose whether to make it a shared disk, which enables it to be failed over, or a non-shared disk. Non-shared disks are not failed over.

The nodes in a Linux FailSafe cluster must follow these requirements:

- The system disk must be a non-shared disk.
- The Linux FailSafe software, in particular the directories `/var/run/failsafe` and `/var/lib/failsafe`, must be on a non-shared disk.

Choosing to make a disk shared or non-shared depends on the needs of the highly available services that use the disk. Each highly available service has requirements about the location of data associated with the service:

- Some data must be placed on non-shared disks
- Some data must be placed on shared disks
- Some data can be on shared or non-shared disks

The figures in the remainder of this section show the basic disk configurations on Linux FailSafe clusters before failover. Each figure also shows the configuration after failover. The basic disk configurations are these:

- A non-shared disk on each node
- Multiple shared disks contained Web server and NFS file server documents

In each of the before and after failover diagrams, just one or two disks are shown. In fact, many disks could be connected in the same way as each disk shown. Thus each disk shown can represent a set of disks.

A Linux cluster can contain a combination of the basic disk configurations listed above.

Figure 2–1, *Non-Shared Disk Configuration and Failover* shows two nodes in a Linux FailSafe cluster, each of which has a non-shared disk with two resource groups. When non-shared disks are used by highly available applications, the data required by those applications must be duplicated on non-shared disks on both nodes. When a failover occurs, IP aliases fail over. The data that was originally available on the failed node is still available from the replacement node by using the IP alias to access it.

The configuration in Figure 2–1, *Non-Shared Disk Configuration and Failover* contains two resource groups, `Group1` and `Group2`. `Group1` contains resource `192.26.50.1` of `IP_address` resource type. `Group2` contains resource `192.26.50.2` of `IP_address` resource type.

Figure 2–1 Non-Shared Disk Configuration and Failover

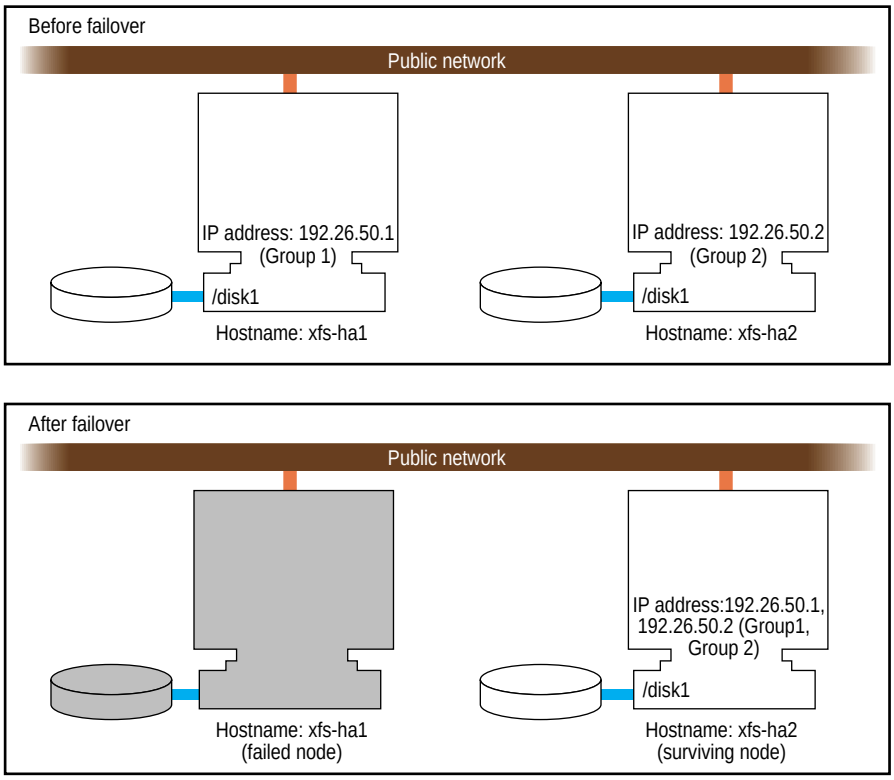


Figure 2–2, *Shared Disk Configuration for Active/Backup Use* shows a two-node configuration with one resource group, `Group1`. Resource group Group1 has a failover domain of (`xfs-ha1`, `xfs-ha2`). Resource group Group1 contains three resources: resource `192.26.50.1` of resource type `IP_address`, resource `/shared` of resource type `filesystem`, and resource `shared_vol` of resource type `volume`.

In this configuration, the resource group `Group1` has a **primary node**, which is the node that accesses the disk prior to a failover. It is shown by a solid line connection. The backup node, which accesses the disk after a failover, is shown by a dotted line. Thus, the disk is shared between the nodes. In an active/backup configuration, all resource groups have the same primary node. The backup node does not run any highly available resource groups until a failover occurs.

Figure 2–2 Shared Disk Configuration for Active/Backup Use

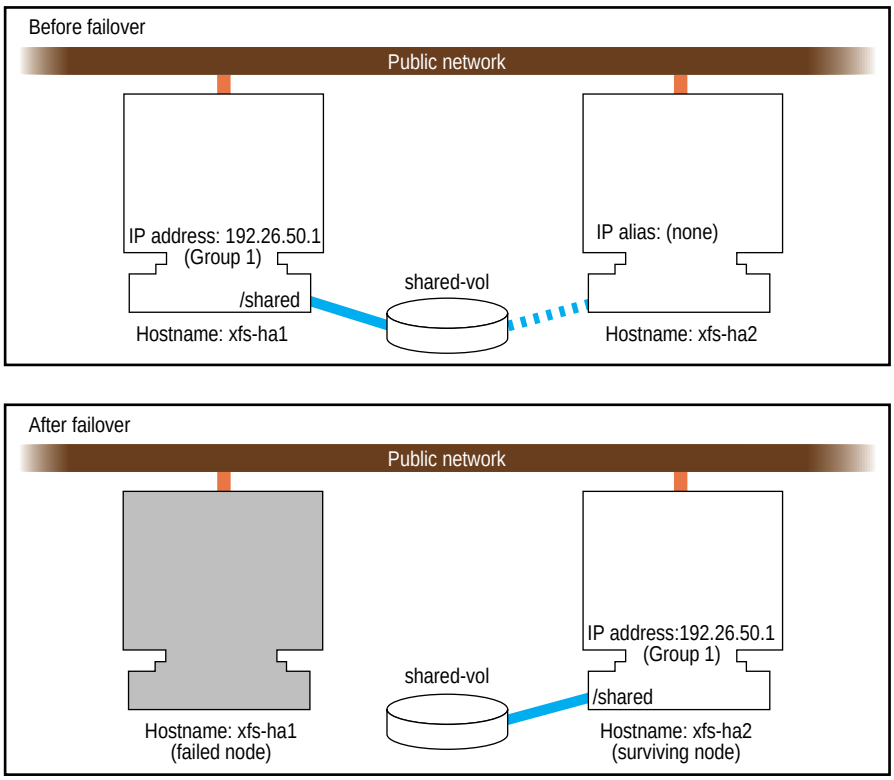


Figure 2–3, *Shared Disk Configuration For Dual-Active Use*, shows two shared disks in a two-node cluster with two resource groups, `Group1` and `Group2`. Resource group `Group1` contains the following resources:

- Resource `192.26.50.1` of type `IP_address`
- Resource `shared1_vol` of type `volume`
- Resource `/shared1` of type `filesystem`

Resource group `Group1` has a failover domain of (`xfs-ha1`, `xfs-ha2`).

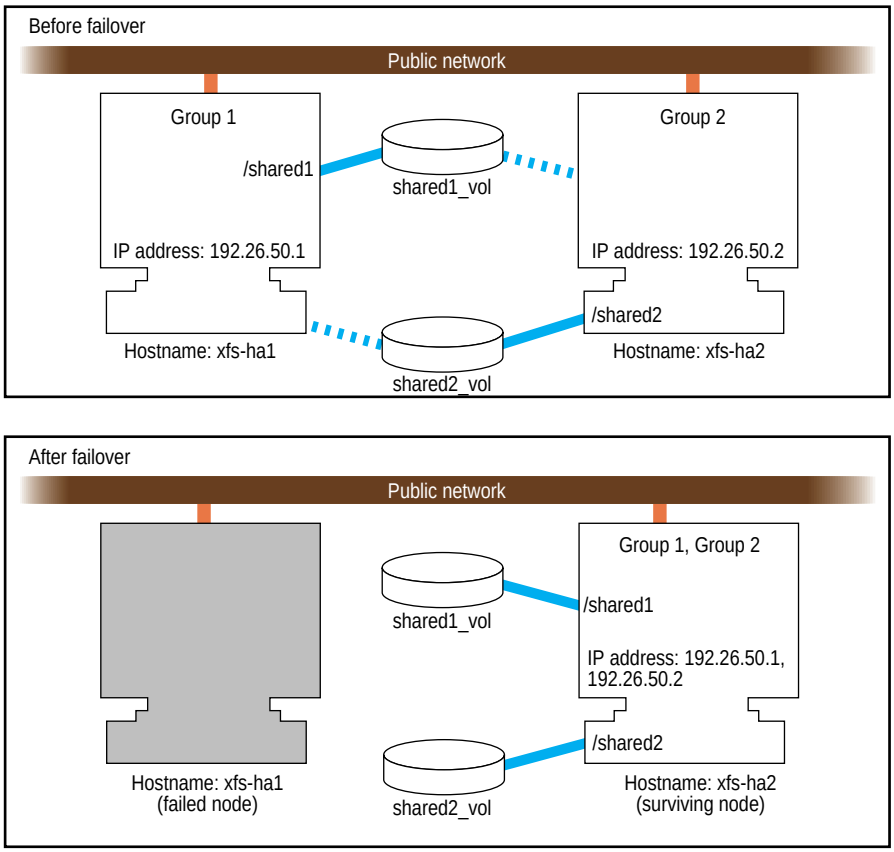
Resource group `Group2` contains the following resources:

- Resource `192.26.50.2` of type `IP_address`
- Resource `shared2_vol` of type `volume`
- Resource `/shared2` of type `filesystem`

Resource group `Group2` has a failover domain of (`xfs-ha2`, `xfs-ha1`).

In this configuration, each node serves as a primary node for one resource group. The solid line connections show the connection to the primary node prior to failover. The dotted lines show the connections to the backup nodes. After a failover, the surviving node has all the resource groups.

Figure 2–3 Shared Disk Configuration For Dual-Active Use



Other sections in this chapter provide more specific information about choosing between shared and non-shared disks for various types of data associated with each highly available service.

2.2.2 Configuration Parameters for Disks

There are no configuration parameters associated with non-shared disks. They are not specified when you configure a Linux FailSafe system. Only shared disks (actually, the logical volumes/ partitions on shared disks) are specified at configuration. See the Section 2.3.1, *Configuration Parameters for Logical Volumes* for details.

2.3 Logical Volume Configuration

The first subsection below describes logical volume issues that must be considered when planning a Linux FailSafe system. The second subsection explains the aspects of the configuration that must be specified for a Linux FailSafe system.

2.3.1 Configuration Parameters for Logical Volumes

Configuration parameters for logical volumes list

- Owner of device filename (default value: `root`)
- Group of device filename (default value: `sys`)

- Mode of device filename (default value: `600`)

Table 2–1, *Logical Volume Configuration Parameters*, lists a label and parameters for individual logical volumes.

Table 2–1 Logical Volume Configuration Parameters

Resource Attribute	volA	volB	volC	Comments
devname-owner	root	root	root	The owner of the device name.
devname-group	sys	sys	root	The group of the device name.
devname-mode	0600	0600	0600	The mode of the device name.

2.4 Filesystem Configuration

The first subsection below describes filesystem issues that must be considered when planning a Linux FailSafe system. The second subsection gives an example of an XFS filesystem configuration on a Linux FailSafe system. The third subsection explains the aspects of the configuration that must be specified for a Linux FailSafe system.

2.4.1 Planning Filesystems

The Linux FailSafe software supports the automatic failover of filesystem including XFS, ext2fs, and reiserfs on shared disks. Shared disks must be either mirrored or RAID storage systems that are shared between the nodes in the two-node Linux FailSafe cluster.

The following are special issues that you need to be aware of when you are working with filesystems on shared disks in a Linux FailSafe cluster:

- All filesystems to be failed over must be supported by Failsafe.
- For availability, filesystems to be failed over in a Linux FailSafe cluster should be created either on mirrored disks or using a system that supports hardware mirroring of the data such as a RAID storage system.
- Create the mount points for the filesystems on all nodes in the failover domain.
- When you set up the various highly available filesystems on each node, make sure that each filesystem uses a different mount point.
- Do not simultaneously mount filesystems on shared disks on more than one node. Doing so causes data corruption. Normally, Linux FailSafe performs all mounts of filesystems on shared disks. If you manually mount a filesystem on a shared disk, make sure that it is not being used by another node.
- Do not place filesystems on shared disks in the `/etc/fstab` file. Linux FailSafe mounts these filesystems only after making sure that another node does not have these filesystems mounted.

The resource name of a resource of the `filesystem` resource type is the mount point of the filesystem.

When clients are actively writing to a Linux FailSafe NFS filesystem during failover of filesystems, data corruption can occur unless filesystems are exported with the mode `sync`. This mode requires that local mounts of the filesystems use the `sync` mount mode as well. Using `sync` affects performance considerably.

2.4.2 Example Filesystem Configuration

Continuing with the example configuration from the Section 2.2.2, *Configuration Parameters for Disks*, say that volumes A and B have XFS filesystems on them:

- The filesystem on volume A is mounted at `/sharedA` with modes `rw` and `noauto`. Call it filesystem A.
- The filesystem on volume B is mounted at `/sharedB` with modes `rw`, `noauto`, and `wsync`. Call it filesystem B.

2.4.3 Configuration Parameters for Filesystems

Table 2–2, *Filesystem Configuration Parameters*, lists a label and configuration parameters for each filesystem.

Table 2–2 Filesystem Configuration Parameters

Resource Attribute	/sharedA	/sharedB	Comments
monitoring-level	2	2	There are two types of monitoring: 1 – checks `/etc/mtab` file 2 – checks if the filesystem is mounted using `stat` command
volume-name	volA	volB	The label of the logical volume on which the filesystem was created.
mode	rw,noauto	rw,noauto,wsync	The mount options used to mount the filesystem. This is specified the same as the options for the mount command or other filesystems listed in `/etc/fstab`.

See Section 3.5, *Choosing and Configuring devices and Filesystems*, for information about creating XFS filesystems.

2.5 IP Address Configuration

The first subsection below describes network interface and IP address issues that must be considered when planning a Linux FailSafe system. The second subsection gives an example of the configuration of network interfaces and IP addresses on a Linux FailSafe system. The third

subsection explains the aspects of the configuration that must be specified for a Linux FailSafe configuration.

2.5.1 Planning Network Interface and IP Address Configuration

Follow these guidelines when planning the configuration of the interfaces to the private network between nodes in a cluster that can be used as a control network between nodes (this information is used when you define the nodes):

- Each interface has one IP address.
- The IP addresses used on each node for the interfaces to the private network are on a different subnet from the IP addresses used for public networks.
- An IP name can be specified for each IP address in `/etc/hosts`.
- Choosing a naming convention for these IP addresses that identifies them with the private network can be helpful. For example, precede the hostname with `priv-` (for **private**), as in `priv-xfs-ha1` and `priv-xfs-ha2`.

Follow these guidelines when planning the configuration of the node interfaces in a cluster to one or more public networks:

- If re-MACing is required, each interface to be failed over requires a dedicated backup interface on the other node (an interface that does not have a highly available IP address). Thus, for each IP address on an interface that requires re-MACing, there should be one interface in each node in the failover domain dedicated for the interface.
- Each interface has a primary IP address. The primary IP address does not fail over.
- The hostname of a node cannot be a highly available IP address.
- All IP addresses used by clients to access highly available services must be part of the resource group to which the HA service belongs.
- If re-MACing is required, all of the highly available IP addresses must have the same backup interface.
- Making good choices for highly available IP addresses is important; these are the “hostnames” that will be used by users of the highly available services, not the true hostnames of the nodes.
- Make a plan for publicizing the highly available IP addresses to the user community, since users of highly available services must use highly available IP addresses instead of the output of the `hostname` command.
- Do not configure highly available IP addresses in static Linux configuration files.

Follow the procedure below to determine whether re-MACing is required (see Section 1.8.2, *Network Interfaces and IP Addresses*, for information about re-MACing). It requires the use of three nodes: *node1*, *node2*, and *node3*. *node1* and *node2* can be nodes of a Linux FailSafe cluster, but they need not be. They must be on the same subnet. *node3* is a third node. If you need to verify that a router accepts gratuitous ARP packets (which means that re-MACing is not required), *node3* must be on the other side of the router from *node1* and *node2*.

1. Configure an IP address on one of the interfaces of *node1*:

```
# /sbin/ifconfig interface inet ip_address netmask netmask up
```

interface is the interface to be used access the node. *ip_address* is an IP address for *node1*. This IP address is used throughout this procedure. *netmask* is the netmask of the IP address.

2. From *node3*, `ping` the IP address used in Step 1 :

```
# ping -c 2 ip_address
PING 190.0.2.1 (190.0.2.1): 56 data bytes
64 bytes from 190.0.2.1: icmp_seq=0 ttl=255 time=29 ms
64 bytes from 190.0.2.1: icmp_seq=1 ttl=255 time=1 ms

----190.0.2.1 PING Statistics----
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 1/1/1 ms
```

3. Enter this command on *node1* to shut down the interface you configured in Step1 :

```
# /sbin/ifconfig interface down
```

4. On *node2*, enter this command to move the IP address to *node2*:

```
# /sbin/ifconfig interface inet ip_address netmask netmask up
```

5. From *node3*, `ping` the IP address:

```
# ping -c 2 ip_address
```

   If the `ping` command fails, gratuitous ARP packets are not being accepted and re-MACing is needed to fail over the IP address.

2.5.2 Example IP Address Configuration

For this example, you are configuring an IP address of 192.26.50.1. This address has a network mask of 255.255.255.0, a broadcast address of 192.26.50.255, and it is configured on interface eth0.

In this example, you are also configuring an IP address of 192.26.50.2. This address also has a network mask of 255.255.255.0, a broadcast address of 192.26.50.255, and it is configured on interface eth1.

Table 2–3, *IP Address Configuration Parameters*, shows the Linux FailSafe configuration parameters you specify for these IP addresses.

Table 2–3 IP Address Configuration Parameters

Resource Attribute	Resource Name: 192.26.50.1	Resource Name: 192.26.50.2
network mask	255.255.255.0	255.255.255.0
broadcast address	192.26.50.255	192.26.50.255
interface	eth0	eth1

2.5.3 Local Failover of IP Addresses

You can configure your system so that an IP address will fail over to a second interface within the same host, for example from eth0 to eth1 on a single node. A configuration example that shows the steps you must follow for this configuration is provided in Section 6.3, *Local Failover of an IP Address*

3 Installing Linux FailSafe Software and Preparing the System

This chapter describes several system administration procedures that must be performed on the nodes in a cluster to prepare and configure them for Linux FailSafe. These procedures assume that you have done the planning described in Chapter 2, *Planning Linux FailSafe Configuration*.

The major sections in this chapter are as follows:

- Section 3.1, *Overview of Configuring Nodes for Linux FailSafe*
- Section 3.2, *Installing Required Software*
- Section 3.3, *Configuring System Files*
- Section 3.4, *Additional Configuration Issues*
- Section 3.5, *Choosing and Configuring devices and Filesystems*
- Section 3.6, *Configuring Network Interfaces*
- Section 3.7, *Configuration for Reset*

3.1 Overview of Configuring Nodes for Linux FailSafe

Performing the system administration procedures required to prepare nodes for Linux FailSafe involves these steps:

1. Install required software as described in Section 3.2, *Installing Required Software*.
2. Configure the system files on each node, as described inSection 3.3, *Configuring System Files*.
3. Check the setting of two important configuration issues on each node as described in Section 3.4, *Additional Configuration Issues*.
4. Create the devices and filesystems required by the highly available applications you plan to run on the cluster. See Section 3.5, *Choosing and Configuring devices and Filesystems*.
5. Configure the network interfaces on the nodes using the procedure in Section 3.6, *Configuring Network Interfaces*.
6. Configure the serial ports used on each node for the serial connection to the other nodes by following the procedure in Section 3.7, *Configuration for Reset*.
7. When you are ready configure the nodes so that Linux FailSafe software starts up when they are rebooted.

To complete the configuration of nodes for Linux FailSafe, you must configure the components of the Linux FailSafe system, as described in Chapter 5, *Linux FailSafe Cluster Configuration*.

3.2 Installing Required Software

The Linux FailSafe base CD requires about 25 MB.

To install the software, follow these steps:

1. Make sure all servers in the cluster are running a supported release of Linux.
2. Depending on the servers and storage in the configuration and the Linux revision level, install the latest install patches that are required for the platform and applications.
3. On each system in the pool, install the version of the multiplexer driver that is appropriate to the operating system. Use the CD that accompanies the multiplexer. Reboot the system after installation.
4. On each node that is part of the pool, install the following software, in order:
   a. `sysadm_base-tcpmux`
   b. `sysadm_base-lib`
   c. `sysadm_base-server`
   d. `cluster_admin`
   e. `cluster_services`
   f. `failsafe`
   g. `sysadm_failsafe-server`

   Note

   You must install `sysadm_base-tcpmux`, `sysadm_base-server`, and `sysadm_failsafe` packages on those nodes from which you want to run the FailSafe GUI. If you do not want to run the GUI on a specific node, you do not need to install these software packages on that node.

5. If the pool nodes are to be administered by a Web-based version of the Linux FailSafe Cluster Manager GUI, install the following subsystems, in order:
   a. `IBMJava118-JRE`
   b. `sysadm_base-client`
   c. `sysadm_failsafe-web`

   If the workstation launches the GUI client from a Web browser that supports Java™, install: `java_plugin` from the Linux FailSafe CD

   If the Java plug-in is not installed when the Linux FailSafe Manager GUI is run from a browser, the browser is redirected to `http://java.sun.com/products/plugin/1.1/plugin-install.html`

   After installing the Java plug-in, you must close all browser windows and restart the browser.

   For a non-Linux workstation, download the Java Plug-in from `http://java.sun.com/products/plugin/1.1/plugin-install.html`

If the Java plug-in is not installed when the Linux FailSafe Manager GUI is run from a browser, the browser is redirected to this site.

d. `sysadm_failsafe-client`

6. Install software on the administrative workstation (GUI client).

   If the workstation runs the GUI client from a Linux desktop, install these subsystems:

   a. `IBMJava118-JRE`

   b. `sysadm_base-client`

7. On the appropriate servers, install other optional software, such as storage management or network board software.

8. Install patches that are required for the platform and applications.

3.3 Configuring System Files

When you install the Linux FailSafe Software, there are some system file considerations you must take into account. This section describes the required and optional changes you make to the following files for every node in the pool:

- `/etc/services`
- `/etc/failsafe/config/cad.options`
- `/etc/failsafe/config/cdbd.options`
- `/etc/failsafe/config/cmond.options`

3.3.1 Configuring /etc/services for Linux FailSafe

The `/etc/services` file must contain entries for `sgi-cmsd`, `sgi-crsd`, `sgi-gcd`, and `sgi-cad` on each node before starting HA services in the node. The port numbers assigned for these processes must be the same in all nodes in the cluster. Note that `sgi-cad` requires a TCP port.

The following shows an example of `/etc/services` entries for `sgi-cmsd`, `sgi-crsd`, `sgi-gcd` and `sgi-cad`:

```
sgi-cmsd   7000/udp           # SGI Cluster Membership Daemon
sgi-crsd   17001/udp           # Cluster reset services daemon
sgi-gcd    17002/udp           # SGI Group Communication Daemon
sgi-cad    17003/tcp           # Cluster Admin daemon
```

3.3.2 Configuring /etc/failsafe/config/cad.options for Linux FailSafe

The `/etc/failsafe/config/cad.options` file contains the list of parameters that the cluster administration daemon (CAD) reads when the process is started. The CAD provides cluster information to the Linux FailSafe Cluster Manager GUI.

The following options can be set in the `cad.options` file:

`--append_log`

Append CAD logging information to the CAD log file instead of overwriting it.

`--log_file` ***filename***

CAD log file name. Alternately, this can be specified as `-lf` *filename.*

`-vvvv`

Verbosity level. The number of "v"s indicates the level of logging. Setting `-v` logs the fewest messages. Setting `-vvvv` logs the highest number of messages.

The following example shows an `/etc/failsafe/config/cad.options` file:

```
-vv -lf /var/log/failsafe/cad_nodename --append_log
```

When you change the `cad.options` file, you must restart the CAD processes with the `/etc/rc.d/init.d/fs_cluster restart` command for those changes to take affect.

3.3.3 Configuring /etc/failsafe/config/cdbd.options for Linux FailSafe

The `/etc/failsafe/config/cdbd.options` file contains the list of parameters that the cdbd daemon reads when the process is started. The cdbd daemon is the configuration database daemon that manages the distribution of cluster configuration database (CDB) across the nodes in the pool.

The following options can be set in the `cdbd.options` file:

`-logevents` ***eventname***

Log selected events. These event names may be used: `all`, `internal`, `args`, `attach`, `chandle`, `node`, `tree`, `lock`, `datacon`, `trap`, `notify`, `access`, `storage`.

The default value for this option is `all`.

`-logdest` ***log_destination***

Set log destination. These log destinations may be used: `all`, `stdout`, `stderr`, `syslog`, `logfile`. If multiple destinations are specified, the log messages are written to all of them. If `logfile` is specified, it has no effect unless the -logfile option is also specified. If the log destination is `stderr` or `stdout`, logging is then disabled if `cdbd` runs as a daemon, because `stdout` and `stderr` are closed when `cdbd` is running as a daemon.

The default value for this option is `logfile`.

`-logfile` ***filename***

Set log file name.

The default value is `/var/log/failsafe/cdbd_log`

`-logfilemax` ***maximum_size***

Set log file maximum size (in bytes). If the file exceeds the maximum size, any preexisting `filename.old` will be deleted, the current file will be renamed to `filename.old`, and a new file will be created. A single message will not be split across files.

If `-logfile` is set, the default value for this option is 10000000.

`-loglevel` ***log level***

Set log level. These log levels may be used: `always`, `critical`, `error`, `warning`, `info`, `moreinfo`, `freq`, `morefreq`, `trace`, `busy`.

The default value for this option is `info`.

`-trace` ***trace class***

Trace selected events. These trace classes may be used: `all`, `rpcs`, `updates`, `transactions`, `monitor`. No tracing is done, even if it is requested for one or more classes of events, unless either or both of `-tracefile` or `-tracelog` is specified.

The default value for this option is `transactions`.

`-tracefile` ***filename***

Set trace file name.

`-tracefilemax` ***maximum size***

Set trace file maximum size (in bytes). If the file exceeds the maximum size, any preexisting `filename.old` will be deleted, the current file will be renamed to `filename.old`.

`-[no]tracelog`

[Do not] trace to log destination. When this option is set, tracing messages are directed to the log destination or destinations. If there is also a trace file, the tracing messages are written there as well.

`-[no]parent_timer`

[Do not] exit when parent exits.

The default value for this option is `-noparent_timer`.

`-[no]daemonize`

[Do not] run as a daemon.

The default value for this option is `-daemonize`.

`-l`

Do not run as a daemon.

`-h`

Print usage message.

`-o help`

Print usage message.

Note that if you use the default values for these options, the system will be configured so that all log messages of level `info` or less, and all trace messages for transaction events to file `/var/log/failsafe/cdbd_log`. When the file size reaches 10MB, this file will be moved to its namesake with the `.old` extension, and logging will roll over to a new file of the same name. A single message will not be split across files.

The following example shows an `/etc/failsafe/config/cdbd.options` file that directs all `cdbd` logging information to `/var/log/messages`, and all `cdbd`

tracing information to /var/log/failsafe/cdbd_ops1. All log events are being logged, and the following trace events are being logged: RPCs, updates and transactions. When the size of the tracefile /var/log/failsafe/cdbd_ops1 exceeds 100000000, this file is renamed to /var/log/failsafe/cdbd_ops1.old and a new file /var/log/failsafe/cdbd_ops1 is created. A single message is not split across files.

```
-logevents all -loglevel trace -logdest syslog -trace rpcs -trace
updates -trace transactions -tracefile /var/log/failsafe/cdbd_ops1
-tracefilemax 100000000
```

The following example shows an /etc/failsafe/config/cdbd.options file that directs all log and trace messages into one file, /var/log/failsafe/cdbd_chaos6, for which a maximum size of 100000000 is specified. -tracelog directs the tracing to the log file.

```
-logevents all -loglevel trace -trace rpcs -trace updates -trace
transactions -tracelog -logfile /var/log/failsafe/cdbd_chaos6
-logfilemax 100000000 -logdest logfile.
```

When you change the cdbd.options file, you must restart the cdbd processes with the /etc/rc.d/init.d/fs_cluster restart command for those changes to take affect.

3.3.4 Configuring /etc/failsafe/config/cmond.options for Linux FailSafe

The/etc/failsafe/config/cmond.options file contains the list of parameters that the cluster monitor daemon (cmond) reads when the process is started. It also specifies the name of the file that logs cmond events. The cluster monitor daemon provides a framework for starting, stopping, and monitoring process groups. See the cmond man page for information on the cluster monitor daemon.

The following options can be set in the cmond.options file:

-L ***loglevel***

Set log level to *loglevel*

-d

Run in debug mode

-l

Lazy mode, where cmond does not validate its connection to the cluster database

-t ***napinterval***

The time interval in milliseconds after which cmond checks for liveliness of process groups it is monitoring

-s [*eventname*]

Log messages to stderr

A default cmond.options file is shipped with the following options. This default options file logs cmond events to the /var/log/failsafe/cmond_log file.

```
-L info -f /var/log/failsafe/cmond_log
```

3.4 Additional Configuration Issues

During the hardware installation of Linux FailSafe nodes, two additional issues must be considered:

- The Linux FailSafe software requires the nodes to be automatically booted when they are reset or when the node is powered on. Linux on x86 will be dependent upon BIOS configuration to ensure this. Some PC BIOSes will hang indefinitely upon error. Clearly this is not useful for high availability situations. On other platforms, such as PowerPC, Alpha, etc, the necessary steps will vary.

  A related, but not identical issue is that of reboots on kernel panics. To ensure the system will reboot even in the case of a kernel failure, set the panic value in a system boot file, such as `init.d/boot.local`:

  ```
  echo "number" > /proc/sys/kernel/panic
  ```

  number is the number of seconds after a panic before the system will reset.

  If you would prefer administrator intervention to be required during a hardware or kernel failure, you may leave this disabled

- The SCSI controllers' host IDs of the nodes in a Linux FailSafe cluster using physically shared storage must be different. If a cluster has no shared storage or is using shared Fibre Channel storage, the value of SCSI host ID is not important.

You can check the ID of most Linux controllers in the logged kernel messages from boot time:

```
# grep ID= /var/log/messages
<6>(scsi0) Wide Channel, SCSI ID=7, 16/255 SCBs
```

Changing the SCSI host ID is specific to the SCSI controller in use. Refer to the controller documentation.

A controller uses its SCSI ID on all buses attached to it. Therefore, you must make sure that no device attached to a node has the same number as its SCSI unit number.

3.5 Choosing and Configuring devices and Filesystems

Creating devices, logical volumes, and filesystems will have a variety of steps specific to the filesystems and other tools selected. Documenting these is outside the scope of this guide. Please refer to the system and distribution-specific documentation for more assistance in this area.

When you create the volumes and filesystems for use with Linux FailSafe, remember these important points:

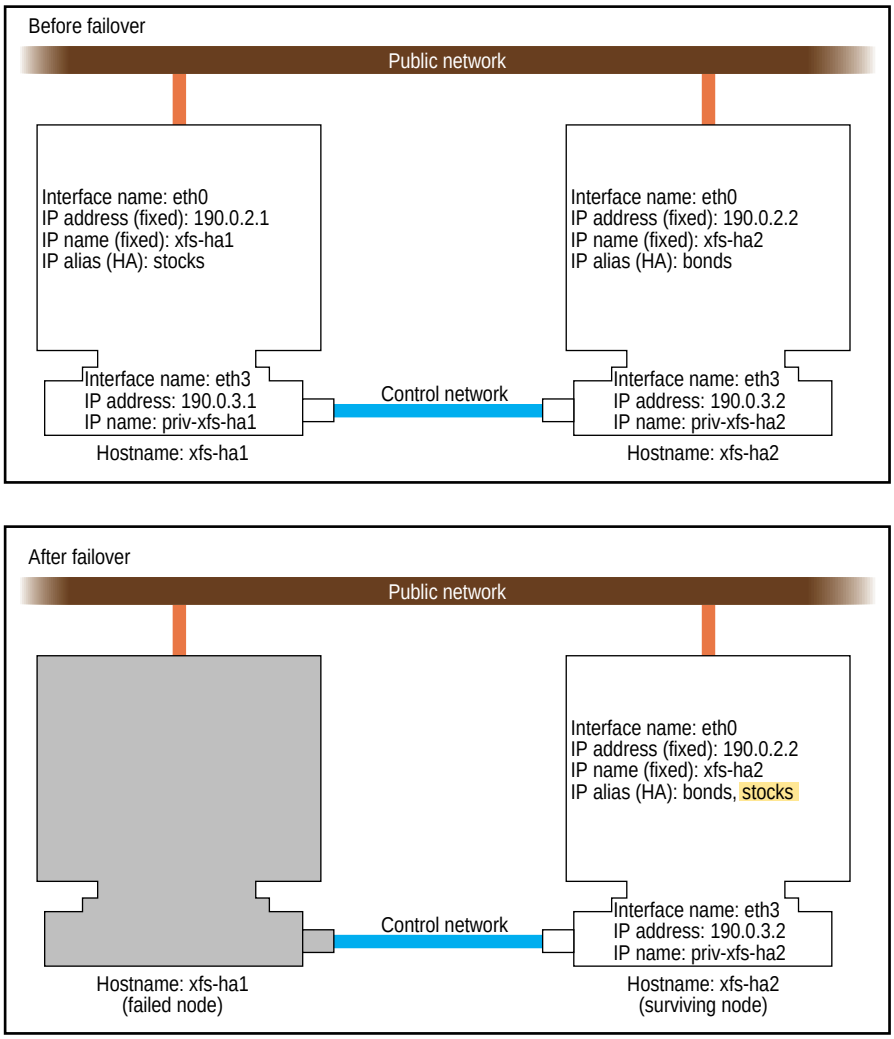
- If the shared disks are not in a RAID storage system, mirrored volumes should be used.
- Each device used must be owned by the same node that is the primary node for the highly available applications that use the logical volume. To simplify the management of the *nodenames* (owners) of volumes on shared disks, follow these recommendations:
  - Work with the volumes on a shared disk from only one node in the cluster.
  - After you create all the volumes on one node, you can selectively shift the ownership to the other node.

- If the volumes you create are used as raw volumes (no filesystem) for storing database data, the database system may require that the device names have specific owners, groups, and modes. If this is the case (see the documentation provided by the database vendor), use the `chown` and `chmod` commands (see the `chown` and `chmod` reference pages) to set the owner, group, and mode as required.
- No filesystem entries are made in `/etc/fstab` for filesystems on shared disks; Linux FailSafe software mounts the filesystems on shared disks. However, to simplify system administration, consider adding comments to `/etc/fstab` that list the filesystems configured for Linux FailSafe. Thus, a system administrator who sees mounted Linux FailSafe filesystems in the output of the `df` command and looks for the filesystems in the `/etc/fstab` file will learn that they are filesystems managed by Linux FailSafe.
- Be sure to create the mount point directory for each filesystem on all nodes in the failover domain.

3.6 Configuring Network Interfaces

The procedure in this section describes how to configure the network interfaces on the nodes in a Linux FailSafe cluster. The example shown in Figure 3–1, *Example Interface Configuration* is used in the procedure.

Figure 3–1 Example Interface Configuration



1. If possible, add every IP address, IP name, and IP alias for the nodes to `/etc/hosts` on one node.

```
190.0.2.1 xfs-ha1.company.com xfs-ha1
190.0.2.3 stocks
190.0.3.1 priv-xfs-ha1
190.0.2.2 xfs-ha2.company.com xfs-ha2
190.0.2.4 bonds
190.0.3.2 priv-xfs-ha2
```

IP aliases that are used exclusively by highly available services should not be added to system configuration files. These aliases will be added and removed by Linux FailSafe.

---

2. Add all of the IP addresses from Step 1 to */etc/hosts* on the other nodes in the cluster.

3. If there are IP addresses, IP names, or IP aliases that you did not add to `/etc/hosts` in Steps 1 and 2, verify that NIS is configured on all nodes in the cluster.

   If the `ypbind` is `off`, you must start NIS. See your distribution's documentation for details.

4. For IP addresses, IP names, and IP aliases that you did not add to `/etc/hosts` on the nodes in Steps 1 and 2, verify that they are in the NIS database by entering this command for each address:

   ```
   # ypmatch address mapname
   190.0.2.1 xfs-ha1.company.com xfs-ha1
   ```

   address is an IP address, IP name, or IP alias. *mapname* is `hosts.byaddr` if address is an IP address; otherwise, it is `hosts`. If `ypmatch` reports that *address* doesn't match, it must be added to the NIS database. See your distribution's documentation for details.

5. On one node, statically configure that node's interface and IP address with the provided distribution tools.

   For the example in Figure 3–1, *Example Interface Configuration*, on a SuSE system, the public interface name and IP address lines are configured into `/etc/rc.config` in the following variables. Please note that YaST is the preferred method for modifying these variables. In any event, you should refer to the documentation of your distribution for help here:

   ```
   NETDEV_0=eth0
   IPADDR_0=$HOSTNAME
   ```

   `$HOSTNAME` is an alias for an IP address that appears in `/etc/hosts`.

   If there are additional public interfaces, their interface names and IP addresses appear on lines like these:

   ```
   NETDEV_1=
   IPADDR_1=
   ```

   In the example, the control network name and IP address are

   ```
   NETDEV_2=eth3
   IPADDR_3=priv-$HOSTNAME
   ```

   The control network IP address in this example, `priv-$HOSTNAME`, is an alias for an IP address that appears in `/etc/hosts`.

6. Repeat Steps 5 and 6 on the other nodes.

7. Verify that Linux FailSafe is `off` on each node:

   ```
   # /usr/lib/failsafe/bin/fsconfig failsafe
   # if [ $? -eq 1 ]; then echo off; else echo on; fi
   ```

   If `failsafe` is `on` on any node, enter this command on that node:

   ```
   # /usr/lib/failsafe/bin/fsconfig failsafe off
   ```

8. Configure an e-mail alias on each node that sends the Linux FailSafe e-mail notifications of cluster transitions to a user outside the Linux FailSafe cluster and to a user on the other

nodes in the cluster. For example, if there are two nodes called `xfs-ha1` and `xfs-ha2`, in `/etc/aliases` on `xfs-ha1`, add

```
fsafe_admin:operations@console.xyz.com,admin_user@xfs-ha2.xyz.com
```

On xfs-ha2, add this line to `/etc/aliases`:

```
fsafe_admin:operations@console.xyz.com,admin_user@xfs-ha1.xyz.com
```

The alias you choose, `fsafe_admin` in this case, is the value you will use for the mail destination address when you configure your system. In this example, `operations` is the user outside the cluster and `admin_user` is a user on each node.

9. If the nodes use NIS (`ypbind` is enabled to start at boot time, or the BIND domain name server (DNS), switching to local name resolution is recommended. Additionally, you should modify the `/etc/nsswitch.conf` file so that it reads as follows:

```
hosts:                  files nis dns
```

Exclusive use of NIS or DNS for IP address lookup for the cluster nodes has been shown to reduce availability in situations where the NIS service becomes unreliable.

---

10. Reboot both nodes to put the new network configuration into effect.

3.7 Configuration for Reset

You can use one of the following methods for reset:

- EMP, which requires the following:
  - Verify that the `getty` processes for serial ports `/dev/ttyS0` and `/dev/ttyS1` are turned off (this is normally the default)
  - Configure the BIOS
- A serial port PCI board to supply additional serial ports
- A USB serial port adapter to supply additional serial ports
- STONITH network-attached power switch,which requires that you enable a `getty` on `/dev/ttyS0`.

3.7.1 Changing the getty Process

The `getty` process for serial ports `/dev/ttyS0` and `/dev/ttyS1` should be off if you are using the EMP port for reset. The `getty` process for serial port `/dev/ttyS0` should be on if you are using STONITH.

To change the setting, perform these steps on each node:

1. Open the file `/etc/inittab` for editing.
2. Find the line for the port by looking at the comments on the right for the port number.
3. Change the third field of this line to `off` or `on`, as required. For example:

```
t2:23:off:/sbin/getty -N ttyd2 co_9600          # port 2
```

4. Save the file.
5. Enter these commands to make the change take effect:

```
# killall getty
# init q
```

3.7.2 Configuring the BIOS

To use the EMP for reset, you must enable the EMP port in the BIOS (server systems shipped by SGI have it enabled by default). If you are comfortable not having a serial console available, then the remaining serial port can be used for reset purposes. This involves going into the BIOS and disabling the console redirection option.

4 Linux FailSafe Administration Tools

This chapter describes Linux FailSafe administration tools and their operation. The major sections in this chapter are as follows:

- Section 4.1, *The Linux FailSafe Cluster Manager Tools*
- Section 4.2, *Using the Linux FailSafe Cluster Manager GUI*
- Section 4.3, *Using the FailSafe Cluster Manager CLI*

4.1 The Linux FailSafe Cluster Manager Tools

You can perform the Linux FailSafe administrative tasks using either of the following tools:

- The Linux FailSafe Cluster Manager Graphical User Interface (GUI)
- The Linux FailSafe Cluster Manager Command Line Interface (CLI)

Although these tools use the same underlying software to configure and monitor a Linux FailSafe system, the GUI provides the following additional features, which are particularly important in a production system:

- Online help is provided with the **Help** button. You can also click any blue text to get more information about that concept or input field.
- The cluster state is shown visually for instant recognition of status, problems, and failovers.
- The state is updated dynamically for continuous system monitoring.
- All inputs are checked for correct syntax before attempting to change the cluster database information.In every task, the cluster configuration will not update until you click **OK**.
- Tasks and tasksets take you step-by-step through configuration and management operations, making actual changes to the cluster database as the you perform a task.
- The graphical tools can be run securely and remotely on any computer that has a Java virtual machine, including Windows® computers and laptops.

The Linux FailSafe Cluster Manager CLI, on the other hand, is more limited in its functions. It enables you to configure and administer a Linux FailSafe system using a command-line interface only on a Linux system. It provides a minimum of help or formatted output and does not provide dynamic status except when queried. An experienced Linux FailSafe administrator may find the Cluster Manager CLI to be convenient when performing basic Linux FailSafe configuration tasks, isolated single tasks in a production environment, or when running scripts to automate some cluster administration tasks.

4.2 Using the Linux FailSafe Cluster Manager GUI

The Linux FailSafe Cluster Manager GUI lets you configure, administer, and monitor a cluster using a graphical user interface.To ensure that the required privileges are available for performing all of the tasks, you should log in to the GUI as `root`. However, some or all privileges can be granted to any user by the system administrator using the Privilege Manager, part of the Linux Interactive Desktop System Administration (`sysadmdesktop`) product. For more information, see the *Personal System Administration Guide*.

The Cluster Manager GUI consists of the FailSafe Cluster View and the FailSafe Manager and its tasks and tasksets.These interfaces are described in the following sections.

4.2.1 The FailSafe Cluster View

The FailSafe Cluster View window provides the following capabilities:

- Shows the relationships among the cluster items (nodes, resources groups, etc.)
- Gives access to every item's configuration and status details
- Shows health of the cluster
- Gives access to the FailSafe Manager and to the SYSLOG
- Gives access to Help information

From the FailSafe Cluster View, the user can click on any item to display key information about it. The items that can be viewed in this way are the following:

- Clusters
- Cluster Nodes
- Resource Types
- Resources
- Resource Groups
- Failover Policies

4.2.2 The FailSafe Manager

The FailSafe Manager provides access to the tasks that help you set up and administer your highly available cluster. The FailSafe Manager also provides access to the FailSafe Guided Configuration tasksets.

- Tasksets consist of a group of tasks collected together to accomplish a larger goal. For example, "Set Up a New Cluster" steps you through the process for creating a new cluster and allows you to launch the necessary tasks by simply clicking their titles.
- FailSafe tasksets let you set up and monitor all the components of a Linux FailSafe cluster using an easy-to-use graphical user interface.

4.2.3 Starting the FailSafe Manager GUI

You can start the FailSafe Manager GUI by launching either the FailSafe Manager or the FailSafe Cluster View.

To launch the FailSafe Manager, use one of these methods:

- Choose "FailSafe Manager" from the desktop (KDE or GNOME) menu.

  You will need to restart the desktop panel after installing Linux FailSafe to see the FailSafe entry in the appropriate menu. To restart the panel, right-click (ring-click) on the panel, and select **`restart`**. In order for this to take effect, the `sysadm_failsafe_client` package must be installed on the client system.

- Enter the following command line:

```
% /usr/bin/fstask
```

- In your Web browser, enter `http://`*`server`*`/FailSafeManager/` (where *server* is the name of node in the pool or cluster that you want to administer) and press Enter. At the resulting Web page, click on the shield icon.

  You can use this method of launching FailSafe Manager if you want to administer the Cluster Manager GUI from a non-Linux system. If you are running the Cluster Manager GUI on a Linux system, the preferred method is to use the desktop panel menu or `/usr/bin/fstask`.

  This method of launching FailSafe Manager works only if you have installed the Java Plug-in, exited all Java processes, restarted your browser, and enabled Java. If there is a long delay before the shield appears, you can click on the “non plug-in” link, but operational glitches may be the result of running in the browser-specific Java.

To launch the FailSafe Cluster View, use one of these methods

- Choose "FailSafe Manager" from the desktop (KDE or GNOME) menu.

  You must restart the desktop panel after installingLinux FailSafe to see the FailSafe entry in the appropriate menu. To restart the panel, right-click (ring-click) on the panel, and select restart. In order for this to take effect, the `sysadm_failsafe-client` package must be installed on the client system.

- Enter the following command line:

```
% /usr/bin/fsdetail
```

The Cluster Manager GUI allows you to administer the entire cluster from a single point of administration. When Linux FailSafe daemons have been activated in a cluster, you must be sure to connect to a node that is running all the Linux FailSafe daemons to obtain the correct cluster status. When Linux FailSafe daemons have not yet been activated in a cluster, you can connect to any node in the pool.

4.2.4 Opening the FailSafe Cluster View window

You can open the FailSafe Cluster View window using either of the following methods:

- Click the “FailSafe Cluster View“ button at the bottom of the FailSafe Manager window.

  This is the preferred method of opening the FailSafe Cluster View window if you will have both the FailSafe Manager and the FailSafe Cluster View windows open at the same time, since it reuses the existing Java process to open the second window instead of starting a new one, which saves memory usage on the client.

- Open the FailSafe Cluster View window directly when you start the FailSafe Manager GUI, as described above in Section 4.2.3, *Starting the FailSafe Manager GUI*.

4.2.5 Viewing Cluster Item Details

To view the details on any cluster item, use the following procedure:

1. Open the FailSafe Cluster View Window.
2. Click the name or icon of any item.

The configuration and status details will appear in a separate window. To see the details in the same window, select Options. When you then click on the Show Details option, the status details will appear in the right side of the window.

4.2.6 Performing Tasks

To perform an individual task with the FailSafe GUI, do the following:

1. Click the name of a category in the left-hand column of the FailSafe Manager window.

   A list of individual tasksets and taskset topics appears in the right-hand column.

2. Click the title of a task in the right-hand column.

   The task window appears.

   You can click any blue text to get more information about that concept or input field.

   ---

3. Enter information in the appropriate fields and click **OK**. to complete the task. (Some tasks consist of more than one window; in these cases, click **Next** to go to the next window, complete the information there, and then click **OK**.

   A dialog box appears confirming the successful completion of the task and displaying additional tasks that you can launch.

4. Continue launching tasks as needed.

4.2.7 Using the Linux FailSafe Tasksets

The FailSafe Manager GUI also provides tasksets to guide you through the steps necessary to complete a goal that encompasses several different tasks. Follow these steps to access the Linux FailSafe tasksets:

1. Click the Guided Configuration category in the lefthand column of the FailSafe Manager window.

   A list of tasksets appears in the right-hand column.

2. Click a taskset in the right-hand column.

   A window appears and lists the series of tasks necessary to accomplish the desired goal.

3. Follow the steps shown, launching tasks by clicking them.

   As you click a task, its task window appears. After you complete all of the tasks listed, you can close the taskset window by double-clicking the upper left corner of its window or clicking Close if there is a Close button on the window.

4.3 Using the FailSafe Cluster Manager CLI

This section documents how to perform cluster administrative tasks by means of the FailSafe Cluster Manager CLI. In order to execute commands with the FailSafe Cluster Manager CLI, you should be logged in as root.

To use the cluster manager, enter either of the following:

```
# /usr/lib/failsafe/bin/cluster_mgr
```

or

```
# /usr/lib/failsafe/bin/cmgr
```

After you have entered this command, you should see the following message and the cluster manager CLI command prompt:

```
Welcome to SGI Cluster Manager Command-Line Interface
cmgr>
```

Once the command prompt displays, you can enter the cluster manager commands.

At any time, you can enter `?` or `help` to bring up the CLI help display.

When you are creating or modifying a component of a Linux FailSafe system, you can enter either of the following commands:

`cancel`

Abort the current mode and discard any changes you have made.

`done`

Commit the current definitions or modifications and return to the `cmgr` prompt.

4.3.1 Entering CLI Commands Directly

There are some Cluster Manager CLI command that you can execute directly from the command line, without entering `cmgr` mode, by using the `-c` option of the `cluster_mgr` command. These commands are `show`, `delete`, `admin`, `install`, `start`, `stop`, `test`, `help`, and `quit`. You can execute these commands directly using the following format:

```
cluster_mgr -c "command"
```

For example, you can execute a `show clusters` CLI command as follows:

```
% /usr/lib/failsafe/bin/cluster_mgr -c "show clusters"

1 Cluster(s) defined
        eagan
```

4.3.2 Invoking the Cluster Manager CLI in Prompt Mode

The Cluster Manager CLI provides an option which displays prompts for the required inputs of administration commands that define and modify Linux FailSafe components. You can run the CLI in prompt mode in either of the following ways:

- Specify a `-p` option when you enter the `cluster_mgr` (or `cmgr`) command, as in the following example:

  ```
  # /usr/lib/failsafe/bin/cluster_mgr -p
  ```

- Execute a `set prompting on` command after you have brought up the CLI, as in the following example:

  ```
  cmgr> set prompting on
  ```

This method of entering prompt mode allows you to toggle in and out of prompt mode as you execute individual CLI commands. To get out of prompt mode while you are running the CLI, enter the following CLI command:

```
cmgr> set prompting
```

For example, if you are not in the prompt mode of the CLI and you enter the following command to define a node, you will see a single prompt, as indicated:

```
cmgr> define node A
Enter commands, when finished enter either "done" or "cancel"
```

A?

At this prompt, you enter the individual node definition commands in the following format (for full information on defining nodes, see Section 5.4.1.2, *Defining a Node with the Cluster Manager CLI*):

```
set hostname to B

set nodeid to C
set sysctrl_type to D
set sysctrl_password to E
set sysctrl_status to F
set sysctrl_owner to G
set sysctrl_device to H
set sysctrl_owner_type to I
add nic J
```

Then, after you add a network interface, a prompt appears requesting the parameters for the network interface, which you enter similarly.

If you are running CLI in prompt mode, however, the display appears as follows (when you provide the appropriate inputs):

```
cmgr> define node A

Enter commands, when finished enter either "done" or "cancel"

Node Name [A]?
Hostname?
Node ID [0]?
Sysctrl Type <chalL|msc|mmsc>?
Sysctrl Password [ ]?
Sysctrl Status <enabled|disabled>?
Sysctrl Owner?
Sysctrl Device?
Sysctrl Owner Type <tty> ?
Number of Controllers [2]?
Controller IP Address?
Controller Heartbeat HB (use network for heartbeats) <true|false>?
Controller (use network for control messages) <true|false>?
Controller Priority <1,2,...>?
```

4.3.3 Using Input Files of CLI Commands

You can execute a series of Cluster Manager CLI commands by using the `-f` option of the `cluster_mgr` command and specifying an input file:

```
/usr/lib/failsafe/bin/cluster_mgr -f "input_file"
```

The input file must contain Cluster Manager CLI commands and end with a `quit` command.

For example, the file `input.file` contains the following:

```
show clusters
show nodes in cluster beta3
quit
```

You can execute the following command, which will yield the indicated output:

```
% /usr/lib/failsafe/bin/cluster_mgr -f input.file

1 Cluster(s) defined
        eagan
Cluster eagan has following 2 machine(s)
        cm1
        cm2
```

The `cluster_mgr` command provides a `-i` option to be used with the `-f` option. This is the "ignore" option which indicates that the Cluster Manager should not exit if a command fails while executing a script.

4.3.4 CLI Command Scripts

You can use the `-f` option of the `cluster_mgr` command to write a script of Cluster Manager CLI commands that you can execute directly. The script must contain the following line as the first line of the script.

```
#!/usr/lib/failsafe/bin/cluster_mgr -f
```

When you use the `-i` option of the `cluster_mgr` command to indicate that the Cluster Manager should not exit if a command fails while executing a script, you must use the following syntax in the first line of the script file: `#!/usr/lib/failsafe/bin/cluster_mgr -if`. It is not necessary to use the `-if` syntax when using the `-i` option from the command line directly.

Each line of the script must be a valid `cluster_mgr command` line, similar to a `here` document. Because the Cluster Manager CLI will run through commands as if entered interactively, you must include `done` and `quit` lines to finish a multi-level command and exit out of the Cluster Manager CLI.

There are CLI template files of scripts that you can modify to configure the different components of your system. These files are located in the `/usr/lib/failsafe/cmgr-templates` directory. For information on CLI templates, see Section 4.3.5, *CLI Template Scripts*.

The following shows an example of a CLI command script `cli.script`.

```
% more cli.script
#!/usr/lib/failsafe/bin/cluster_mgr -f

show clusters
show nodes in cluster beta3
quit

% cli.script
1 Cluster(s) defined
        eagan
Cluster eagan has following 2 machine(s)
        cm1
        cm2

%
```

For a complete example of a CLI command script that configures a cluster, see Section 5.8, *Linux FailSafe Configuration Example CLI Script* in Chapter 5, *Linux FailSafe Cluster Configuration*.

4.3.5 CLI Template Scripts

Template files of CLI scripts that you can modify to configure the different components of your system are located in the `/usr/lib/failsafe/cmgr-templates` directory.

Each template file contains list of `cluster_mgr` commands to create a particular object, as well as comments describing each field. The template also provides default values for optional fields.

The `/usr/lib/failsafe/cmgr-templates` directory contains following templates:

Table 4–1 Available Templates

File name	*Description*
`cmgr-create-cluster`	Creation of a cluster
`cmgr-create-failover_policy`	Creation of failover policy
`cmgr-create-node`	Creation of node
`cmgr-create-resource_group`	Creation of Resource Group
`cmgr-create-resource_type`	Creation of resource type
`cmgr-create-resource-`*`resource type`*	CLI script template for creation of resource of type *resource type*

To create a Linux FailSafe configuration, you can concatenate multiple templates into one file and execute the resulting CLI command script.

If you concatenate information from multiple template scripts to prepare your cluster configuration, you must remove the `quit` at the end of each template script, except for the final `quit`. A `cluster_mgr` script must have only one `quit` line.

For example: For a 3 node configuration with an NFS resource group containing 1 volume, 1 filesystem, 1 IP address and 1 NFS resource, you would concatenate the following files, removing the `quit` at the end of each template script except the last one:

- 3 copies of the `cmgr-create-node` file
- 1 copy of the `cmgr-create-cluster` file
- 1 copy of the `cmgr-create-failover_policy` file
- 1 copy of the `cmgr-create-resource_group` file
- 1 copy of the `cmgr-create-resource-volume` file
- 1 copy of the `cmgr-create-resource-filesystem` file
- 1 copy of the `cmgr-create-resource-IP_address` file
- 1 copy of the `cmgr-create-resource-NFS` file

4.3.6 Invoking a Shell from within CLI

You can invoke a shell from within the Cluster Manager CLI. Enter the following command to invoke a shell:

```
cmgr> sh
```

To exit the shell and to return to the CLI, enter `exit` at the shell prompt.

5 Linux FailSafe Cluster Configuration

This chapter describes administrative tasks you perform to configure the components of a Linux FailSafe system. It describes how to perform tasks using the FailSafe Cluster Manager Graphical User Interface (GUI) and the FailSafe Cluster Manager Command Line Interface (CLI). The major sections in this chapter are as follows:

- Section 5.1, *Setting Configuration Defaults*
- Section 5.2, *Name Restrictions*
- Section 5.3, *Configuring Timeout Values and Monitoring Intervals*
- Section 5.4, *Cluster Configuration*
- Section 5.5, *Resource Configuration*
- Section 5.6, *Linux FailSafe System Log Configuration*
- Section 5.7, *Resource Group Creation Example*
- Section 5.8, *Linux FailSafe Configuration Example CLI Script*

5.1 Setting Configuration Defaults

Before you configure the components of a FailSafe system, you can set default values for some of the components that Linux FailSafe will use when defining the components.

Default cluster

Certain cluster manager commands require you to specify a cluster. You can specify a default cluster to use as the default if you do not specify a cluster explicitly.

Default node

Certain cluster manager commands require you to specify a node. With the Cluster Manager CLI, you can specify a default node to use as the default if you do not specify a node explicitly.

Default resource type

Certain cluster manager commands require you to specify a resource type. With the Cluster Manager CLI, you can specify a default resource type to use as the default if you do not specify a resource type explicitly.

5.1.1 Setting Default Cluster with the Cluster Manager GUI

The GUI prompts you to enter the name of the default cluster when you have not specified one. Alternately, you can set the default cluster by clicking the “Select Cluster...” button at the bottom of the FailSafe Manager window.

When using the GUI, there is no need to set a default node or resource type.

5.1.2 Setting and Viewing Configuration Defaults with the Cluster Manager CLI

When you are using the Cluster Manager CLI, you can use the following commands to specify default values. The default values are in effect only for the current session of the Cluster Manager CLI.

Use the following command to specify a default cluster:

```
cmgr> set cluster A
```

Use the following command to specify a default node:

```
cmgr> set node A
```

Use the following command to specify a default resource type:

```
cmgr> set resource_type A
```

You can view the current default configuration values of the Cluster Manager CLI with the following command:

```
cmgr> show set defaults
```

5.2 Name Restrictions

When you specify the names of the various components of a Linux FailSafe system, the name cannot begin with an underscore (_) or include any whitespace characters. In addition, the name of any Linux FailSafe component cannot contain a space, an unprintable character, or a *, ?, \, or #.

The following is the list of permitted characters for the name of a Linux FailSafe component:

- alphanumeric characters
- /
- .
- - (hyphen)
- _ (underscore)
- :
- “
- =
- @
- ’

These character restrictions hold true whether you are configuring your system with the Cluster Manager GUI or the Cluster Manager CLI.

5.3 Configuring Timeout Values and Monitoring Intervals

When you configure the components of a Linux FailSafe system, you configure various timeout values and monitoring intervals that determine the application downtown of a highly-available system when there is a failure. To determine reasonable values to set for your system, consider the following equation:

application downtime = failure detection + time to handle failure + failure recovery

Failure detection depends on the type of failure that is detected:

- When a node goes down, there will be a node failure detection after the node timeout; this is an HA parameter that you can modify. All failures that translate into a node failure (such as heartbeat failure and OS failure) fall into this failure category. Node timeout has a default value of 15 seconds. For information on modifying the node timeout value, see Section 5.4.4, *Linux FailSafe HA Parameters*.
- When there is a resource failure, there is a monitor failure of a resource. The amount of time this will take is determined by the following:
  - The monitoring interval for the resource type
  - The monitor timeout for the resource type
  - The number of restarts defined for the resource type, if the restart mode is configured on

  For information on setting values for a resource type, see Section 5.5.6, *Defining a Resource Type*.

Reducing these values will result in a shorter failover time, but reducing these values could lead to significant increase in the Linux FailSafe overhead on the system performance and could also lead to false failovers.

The time to handle a failure is something that the user cannot control. In general, this should take a few seconds.

The failure recovery time is determined by the total time it takes for Linux FailSafe to perform the following:

- Execute the failover policy script (approximately five seconds).
- Run the stop action script for all resources in the resource group. This is not required for node failure; the failing node will be reset.
- Run the start action script for all resources in the resource group

5.4 Cluster Configuration

To set up a Linux FailSafe system, you configure the cluster that will support the highly available services. This requires the following steps:

- Defining the local host
- Defining any additional nodes that are eligible to be included in the cluster
- Defining the cluster

The following subsections describe these tasks.

5.4.1 Defining Cluster Nodes

A **cluster node** is a single Linux image. Usually, a cluster node is an individual computer. The term **node** is also used in this guide for brevity.

The **pool** is the entire set of nodes available for clustering.

The first node you define must be the local host, which is the host you have logged into to perform cluster administration.

When you are defining multiple nodes, it is advisable to wait for a minute or so between each node definition. When nodes are added to the configuration database, the contents of the configuration database are also copied to the node being added. The node definition operation is completed when the new node configuration is added to the database, at which point the database configuration is synchronized. If you define two nodes one after another, the second operation might fail because the first database synchronization is not complete.

To add a logical node definition to the pool of nodes that are eligible to be included in a cluster, you must provide the following information about the node:

- Logical name: This name can contain letters and numbers but not spaces or pound signs. The name must be composed of no more than 255 characters. Any legal hostname is also a legal node name. For example, for a node whose hostname is "venus.eng.company.com" you can use a node name of "venus", "node1", or whatever is most convenient.
- Hostname: The fully qualified name of the host, such as "server1.company.com". Hostnames cannot begin with an underscore, include any whitespace, or be longer than 255 characters. This hostname should be the same as the output of the hostname command on the node you are defining. The IP address associated with this hostname should not be the same as any IP address you define as highly available when you define a Linux FailSafe IP address resource. Linux FailSafe will not accept an IP address (such as "192.0.2.22") for this input.
- Node ID: This number must be unique for each node in the pool and be in the range 1 through 32767.
- System controller information. If the node has a system controller and you want Linux FailSafe to use the controller to reset the node, you must provide the following information about the system controller:
  - Type of system controller: `chalL`, `msc`, `mmsc`
  - System controller port password (optional)
  - Administrative status, which you can set to determine whether Linux FailSafe can use the port: `enabled`, `disabled`
  - Logical node name of system controller owner (i.e. the system that is physically attached to the system controller)
  - Device name of port on owner node that is attached to the system controller
  - Type of owner device: `tty`
- A list of control networks, which are the networks used for heartbeats, reset messages, and other Linux FailSafe messages. For each network, provide the following:
  - Hostname or IP address. This address must not be the same as any IP address you define as highly available when you define a Linux FailSafe IP address resource, and it must be resolved in the `/etc/hosts` file.
  - Flags (`hb` for heartbeats, `ctrl` for control messages, `priority`). At least two control networks must use heartbeats, and at least one must use control messages.

    Linux FailSafe requires multiple heartbeat networks. Usually a node sends heartbeat messages to another node on only one network at a time. However, there are times when a

node might send heartbeat messages to another node on multiple networks simultaneously. This happens when the sender node does not know which networks are up and which others are down. This is a transient state and eventually the heartbeat network converges towards the highest priority network that is up.

Note that at any time different pairs of nodes might be using different networks for heartbeats.

Although all nodes in the Linux FailSafe cluster should have two control networks, it is possible to define a node to add to the pool with one control network.

5.4.1.1 Defining a Node with the Cluster Manager GUI

To define a node with the Cluster Manager GUI, perform the following steps:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the "Nodes & Cluster" category.
3. On the right side of the display click on the "Define a Node" task link to launch the task.
4. Enter the selected inputs on this screen. Click on "Next" at the bottom of the screen and continue inputing information on the second screen.
5. Click on "OK" at the bottom of the screen to complete the task, or click on "Cancel" to cancel.

5.4.1.2 Defining a Node with the Cluster Manager CLI

Use the following command to add a logical node definition:

```
cmgr> define node A
```

Entering this command specifies the name of the node you are defining and puts you in a mode that enables you to define the parameters of the node. These parameters correspond to the items defined in Section 5.4.1, *Defining Cluster Nodes*. The following prompts appear:

```
Enter commands, when finished enter either "done" or "cancel"
```

A?

When this prompt of the node name appears, you enter the node parameters in the following format:

```
set hostname to B
set nodeid to C
set sysctrl_type to D
set sysctrl_password to E
set sysctrl_status to F
set sysctrl_owner to G
set sysctrl_device to H
set sysctrl_owner_type to I
add nic J
```

You use the `add nic` *J* command to define the network interfaces. You use this command for each network interface to define. When you enter this command, the following prompt appears:

```
Enter network interface commands, when finished enter "done" or "cancel"
NIC - J?
```

When this prompt appears, you use the following commands to specify the flags for the control network:

```
set heartbeat to K
set ctrl_msgs to L
set priority to M
```

After you have defined a network controller, you can use the following command from the node name prompt to remove it:

```
cmgr> remove nic N
```

When you have finished defining a node, enter `done`.

The following example defines a node called cm1a, with one controller:

```
cmgr> define node cm1a
Enter commands, when finished enter either "done" or "cancel"

cm1a? set hostname to cm1a
cm1a? set nodeid to 1
cm1a? set sysctrl_type to msc
cm1a? set sysctrl_password to [ ]
cm1a? set sysctrl_status to enabled
cm1a? set sysctrl_owner to cm2
cm1a? set sysctrl_device to /dev/ttyd2
cm1a? set sysctrl_owner_type to tty
cm1a? add nic cm1
Enter network interface commands, when finished enter ‘‘done’’
or ‘‘cancel’’

NIC - cm1 > set heartbeat to true
NIC - cm1 > set ctrl_msgs to true
NIC - cm1 > set priority to 0
NIC - cm1 > done
cm1a? done
cmgr>
```

If you have invoked the Cluster Manager CLI with the `-p` option,or you entered the `set prompting on` command, the display appears as in the following example:

```
cmgr> define node cm1a
Enter commands, when finished enter either "done" or "cancel"

Nodename [optional]? cm1a

Node ID? 1
Do you wish to define system controller info[y/n]:y
Sysctrl Type <null>? (null)
Sysctrl Password[optional]? ( )
Sysctrl Status <enabled|disabled>? enabled
Sysctrl Owner? cm2
Sysctrl Device? /dev/ttyd2
Sysctrl Owner Type <tty>? (tty)
Number of Network Interfaces ? (1)
NIC 1 - IP Address? cm1
```

```
NIC 1 - Heartbeat HB (use network for heartbeats) <true|false>? true
NIC 1 - Priority <1,2,...>? 0
NIC 2 - IP Address? cm2
NIC 2 - Heartbeat HB (use network for heartbeats) <true|false>? true
NIC 2 - (use network for control messages) <true|false>? false
NIC 2 - Priority <1,2,...>? 1
```

5.4.2 Modifying and Deleting Cluster Nodes

After you have defined a cluster node, you can modify or delete the cluster with the Cluster Manager GUI or the Cluster Manager CLI. You must remove a node from a cluster before you can delete the node.

5.4.2.1 Modifying a Node with the Cluster Manager GUI

To modify a node with the Cluster Manager GUI, perform the following steps:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the “Nodes & Cluster” category.
3. On the right side of the display click on the “Modify a Node Definition” task link to launch the task.
4. Modify the node parameters.
5. Click on “OK” at the bottom of the screen to complete the task, or click on “Cancel” to cancel.

5.4.2.2 Modifying a Node with the Cluster Manager CLI

You can use the following command to modify an existing node. After entering this command, you can execute any of the commands you use to define a node.

```
cmgr> modify node A
```

5.4.2.3 Deleting a Node with the Cluster Manager GUI

To delete a node with the Cluster Manager GUI, perform the following steps:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the “Nodes & Cluster” category.
3. On the right side of the display click on the “Delete a Node” task link to launch the task.
4. Enter the name of the node to delete.
5. Click on “OK” at the bottom of the screen to complete the task, or click on “Cancel” to cancel.

5.4.2.4 Deleting a Node with the Cluster Manager CLI

After defining a node, you can delete it with the following command:

```
cmgr> delete node A
```

You can delete a node only if the node is not currently part of a cluster. This means that first you must modify a cluster that contains the node so that it no longer contains that node before you can delete it.

5.4.3 Displaying Cluster Nodes

After you define cluster nodes, you can perform the following display tasks:

- display the attributes of a node
- display the nodes that are members of a specific cluster
- display all the nodes that have been defined

You can perform any of these tasks with the FailSafe Cluster Manager GUI or the Linux FailSafe Cluster Manager CLI.

5.4.3.1 Displaying Nodes with the Cluster Manager GUI

The Cluster Manager GUI provides a convenient graphic display of the defined nodes of a cluster and the attributes of those nodes through the FailSafe Cluster View. You can launch the FailSafe Cluster View directly, or you can bring it up at any time by clicking on "FailSafe Cluster View" at the bottom of the "FailSafe Manager" display.

From the View menu of the FailSafe Cluster View, you can select "Nodes in Pool" to view all nodes defined in the Linux FailSafe pool. You can also select "Nodes In Cluster" to view all nodes that belong to the default cluster. Click any node's name or icon to view detailed status and configuration information about the node.

5.4.3.2 Displaying Nodes with the Cluster Manager CLI

After you have defined a node, you can display the node's parameters with the following command:

```
cmgr> show node A
```

A `show node` command on node cm1a would yield the following display:

```
cmgr> show node cm1
Logical Node Name: cm1
Hostname: cm1
Nodeid: 1
Reset type: reset
System Controller: msc
System Controller status: enabled
System Controller owner: cm2
System Controller owner device: /dev/ttyd2
System Controller owner type: tty
ControlNet Ipaddr: cm1
ControlNet HB: true
ControlNet Control: true
ControlNet Priority: 0
```

You can see a list of all of the nodes that have been defined with the following command:

```
cmgr> show nodes in pool
```

You can see a list of all of the nodes that have defined for a specified cluster with the following command:

```
cmgr> show nodes [in cluster A]
```

If you have specified a default cluster, you do not need to specify a cluster when you use this command and it will display the nodes defined in the default cluster.

5.4.4 Linux FailSafe HA Parameters

There are several parameters that determine the behavior of the nodes in a cluster of a Linux FailSafe system.

The Linux FailSafe parameters are as follows:

- The tie-breaker node, which is the logical name of a machine used to compute node membership in situations where 50% of the nodes in a cluster can talk to each other. If you do not specify a tie-breaker node, the node with the lowest node ID number is used.

  The tie-breaker node is a cluster-wide parameter.

  It is recommended that you configure a tie-breaker node even if there is an odd number of nodes in the cluster, since one node may be deactivated, leaving an even number of nodes to determine membership.

  In a heterogeneous cluster, where the nodes are of different sizes and capabilities, the largest node in the cluster with the most important application or the maximum number of resource groups should be configured as the tie-breaker node.

- Node timeout, which is the timeout period, in milliseconds. If no heartbeat is received from a node in this period of time, the node is considered to be dead and is not considered part of the cluster membership.

  The node timeout must be at least 5 seconds. In addition, the node timeout must be at least 10 times the heartbeat interval for proper Linux FailSafe operation; otherwise, false failovers may be triggered.

  Node timeout is a cluster-wide parameter.

- The interval, in milliseconds, between heartbeat messages. This interval must be greater than 500 milliseconds and it must not be greater than one-tenth the value of the node timeout period. This interval is set to one second, by default. Heartbeat interval is a cluster-wide parameter.

  The higher the number of heartbeats (smaller heartbeat interval), the greater the potential for slowing down the network. Conversely, the fewer the number of heartbeats (larger heartbeat interval), the greater the potential for reducing availability of resources.

- The node wait time, in milliseconds, which is the time a node waits for other nodes to join the cluster before declaring a new cluster membership. If the value is not set for the cluster, Linux FailSafe assumes the value to be the node timeout times the number of nodes.

- The powerfail mode, which indicates whether a special power failure algorithm should be run when no response is received from a system controller after a reset request. This can be set to `ON` or `OFF`. Powerfail is a node-specific parameter, and should be defined for the machine that performs the reset operation.

5.4.4.1 Resetting Linux FailSafe Parameters with the Cluster Manager GUI

To set Linux FailSafe parameters with the Cluster Manager GUI, perform the following steps:

1. Launch the FailSafe Manager from a menu or the command line.

2. On the left side of the display, click on the “Nodes & Cluster” category.
3. On the right side of the display click on the “Set Linux FailSafe HA Parameters” task link to launch the task.
4. Enter the selected inputs.
5. Click on “OK” at the bottom of the screen to complete the task, or click on “Cancel” to cancel.

5.4.4.2 Resetting Linux FailSafe Parameters with the Cluster Manager CLI

You can modify the Linux FailSafe parameters with the following command:

```
cmgr> modify ha_parameters [on node A] [in cluster B]
```

If you have specified a default node or a default cluster, you do not have to specify a node or a cluster in this command. Linux FailSafe will use the default.

```
Enter commands, when finished enter either "done" or "cancel"
```

A?

When this prompt of the node name appears, you enter the Linux FailSafe parameters you wish to modify in the following format:

```
set node_timeout to A
set heartbeat to B
set run_pwrfail to C
set tie_breaker to D
```

5.4.5 Defining a Cluster

A **cluster** is a collection of one or more nodes coupled with each other by networks or other similar interconnects. In Linux FailSafe, a cluster is identified by a simple name. A given node may be a member of only one cluster.

To define a cluster, you must provide the following information:

- The logical name of the cluster, with a maximum length of 255 characters.
- The mode of operation: `normal` (the default) or `experimental`. Experimental mode allows you to configure a Linux FailSafe cluster in which resource groups do not fail over when a node failure is detected. This mode can be useful when you are tuning node timeouts or heartbeat values. When a cluster is configured in normal mode, Linux FailSafe fails over resource groups when it detects failure in a node or resource group.
- (Optional) The email address to use to notify the system administrator when problems occur in the cluster (for example, `root@system`)
- (Optional) The email program to use to notify the system administrator when problems occur in the cluster (for example, `/usr/bin/mail`).

  Specifying the email program is optional and you can specify only the notification address in order to receive notifications by mail. If an address is not specified, notification will not be sent.

5.4.5.1 Adding Nodes to a Cluster

After you have added nodes to the pool and defined a cluster, you must provide the names of the nodes to include in the cluster.

5.4.5.2 Defining a Cluster with the Cluster Manager GUI

To define a cluster with the Cluster Manager GUI, perform the following steps:

1. Launch the Linux FailSafe Manager.
2. On the left side of the display, click on “Guided Configuration”.
3. On the right side of the display click on “Set Up a New Cluster” to launch the task link.
4. In the resulting window, click each task link in turn, as it becomes available. Enter the selected inputs for each task.
5. When finished, click “OK” to close the taskset window.

5.4.5.3 Defining a Cluster with the Cluster Manager CLI

When you define a cluster with the CLI, you define and cluster and add nodes to the cluster with the same command.

Use the following cluster manager CLI command to define a cluster:

```
cmgr> define cluster A
```

Entering this command specifies the name of the node you are defining and puts you in a mode that allows you to add nodes to the cluster. The following prompt appears:

```
cluster A?
```

When this prompt appears during cluster creation, you can specify nodes to include in the cluster and you can specify an email address to direct messages that originate in this cluster.

You specify nodes to include in the cluster with the following command:

```
cluster A? add node C
cluster A?
```

You can add as many nodes as you want to include in the cluster.

You specify an email program to use to direct messages with the following command:

```
cluster A? set notify_cmd to B
cluster A?
```

You specify an email address to direct messages with the following command:

```
cluster A? set notify_addr to B
cluster A?
```

You specify a mode for the cluster (normal or experimental) with the following command:

```
cluster A? set ha_mode to D
cluster A?
```

When you are finished defining the cluster, enter `done` to return to the `cmgr` prompt.

5.4.6 Modifying and Deleting Clusters

After you have defined a cluster, you can modify the attributes of the cluster or you can delete the cluster. You cannot delete a cluster that contains nodes; you must move those nodes out of the cluster first.

5.4.6.1 Modifying and Deleting a Cluster with the Cluster Manager GUI

To modify a cluster with the Cluster Manager GUI, perform the following procedure:

1. Launch the Linux FailSafe Manager.
2. On the left side of the display, click on the “Nodes & Cluster” category.
3. On the right side of the display click on the “Modify a Cluster Definition” task link to launch the task.
4. Enter the selected inputs.
5. Click on “OK” at the bottom of the screen to complete the task, or click on “Cancel” to cancel.

To delete a cluster with the Cluster Manager GUI, perform the following procedure:

1. Launch the Linux FailSafe Manager.
2. On the left side of the display, click on the “Nodes & Cluster” category.
3. On the right side of the display click on the “Delete a Cluster” task link to launch the task.
4. Enter the selected inputs.
5. Click on “OK” at the bottom of the screen to complete the task, or click on “Cancel” to cancel.

5.4.6.2 Modifying and Deleting a Cluster with the Cluster Manager CLI

To modify an existing cluster, enter the following command:

```
cmgr> modify cluster A
```

Entering this command specifies the name of the cluster you are modifying and puts you in a mode that allows you to modify the cluster. The following prompt appears:

```
cluster A?
```

When this prompt appears, you can modify the cluster definition with the following commands:

```
cluster A? set notify_addr to B
cluster A? set notify_cmd to B
cluster A? add node C
cluster A? remove node D
cluster A?
```

When you are finished modifying the cluster, enter `done` to return to the `cmgr` prompt.

You can delete a defined cluster with the following command:

```
cmgr> delete cluster A
```

5.4.7 Displaying Clusters

You can display defined clusters with the Cluster Manager GUI or the Cluster Manager CLI.

5.4.7.1 Displaying a Cluster with the Cluster Manager GUI

The Cluster Manager GUI provides a convenient display of a cluster and its components through the FailSafe Cluster View. You can launch the FailSafe Cluster View directly, or you can bring it up at any time by clicking on the "FailSafe Cluster View" prompt at the bottom of the "FailSafe Manager" display.

From the View menu of the FailSafe Cluster View, you can choose elements within the cluster to examine. To view details of the cluster, click on the cluster name or icon. Status and configuration information will appear in a new window. To view this information within the FailSafe Cluster View window, select Options. When you then click on the Show Details option, the status details will appear in the right side of the window.

5.4.8 Displaying a Cluster with the Cluster Manager CLI

After you have defined a cluster, you can display the nodes in that cluster with the following command:

```
cmgr> show cluster A
```

You can see a list of the clusters that have been defined with the following command:

```
cmgr> show clusters
```

5.5 Resource Configuration

A **resource** is a single physical or logical entity that provides a service to clients or other resources. A resource is generally available for use on two or more nodes in a cluster, although only one node controls the resource at any given time. For example, a resource can be a single disk volume, a particular network address, or an application such as a web node.

5.5.1 Defining Resources

Resources are identified by a resource name and a resource type. A **resource name** identifies a specific instance of a resource type. A **resource type** is a particular class of resource. All of the resources in a given resource type can be handled in the same way for the purposes of failover. Every resource is an instance of exactly one resource type.

A resource type is identified with a simple name. A resource type can be defined for a specific logical node, or it can be defined for an entire cluster. A resource type that is defined for a node will override a clusterwide resource type definition of the same name; this allows an individual node to override global settings from a clusterwide resource type definition.

The Linux FailSafe software includes many predefined resource types. If these types fit the application you want to make into a highly available service, you can reuse them. If none fit, you can define additional resource types.

To define a resource, you provide the following information:

- The name of the resource to define, with a maximum length of 255 characters.

- The type of resource to define. The Linux FailSafe system contains some pre-defined resource types (template and `IP_Address`). You can define your own resource type as well.
- The name of the cluster that contains the resource.
- The logical name of the node that contains the resource (optional). If you specify a node, a local version of the resource will be defined on that node.
- Resource type-specific attributes for the resource. Each resource type may require specific parameters to define for the resource, as described in the following subsections.

You can define up to 100 resources in a Linux FailSafe configuration.

5.5.1.1 IP Address Resource Attributes

The IP Address resources are the IP addresses used by clients to access the highly available services within the resource group. These IP addresses are moved from one node to another along with the other resources in the resource group when a failure is detected.

You specify the resource name of an IP address in dotted decimal notation. IP names that require name resolution should not be used. For example, 192.26.50.1 is a valid resource name of the IP Address resource type.

The IP address you define as a Linux FailSafe resource must not be the same as the IP address of a node hostname or the IP address of a node's control network.

When you define an IP address, you can optionally specifying the following parameters. If you specify any of these parameters, you must specify all of them.

- The broadcast address for the IP address.
- The network mask of the IP address.
- A comma-separated list of interfaces on which the IP address can be configured. This ordered list is a superset of all the interfaces on all nodes where this IP address might be allocated. Hence, in a mixed cluster with different ethernet drivers, an IP address might be placed on eth0 on one system and ln0 on a another. In this case the `interfaces` field would be `eth0,ln0` or `ln0,eth0`.

  The order of the list of interfaces determines the priority order for determining which IP address will be used for local restarts of the node.

5.5.2 Adding Dependency to a Resource

One resource can be dependent on one or more other resources; if so, it will not be able to start (that is, be made available for use) unless the dependent resources are started as well. Dependent resources must be part of the same resource group.

Like resources, a resource type can be dependent on one or more other resource types. If such a dependency exists, at least one instance of each of the dependent resource types must be defined. For example, a resource type named `Netscape_web` might have resource type dependencies on a resource types named `IP_address` and `volume`. If a resource named `ws1` is defined with the `Netscape_web` resource type, then the resource group containing `ws1` must also contain at least one resource of the type `IP_address` and one resource of the type `volume`.

You cannot make resources mutually dependent. For example, if resource A is dependent on resource B, then you cannot make resource B dependent on resource A. In addition, you cannot

define cyclic dependencies. For example, if resource A is dependent on resource B, and resource B is dependent on resource C, then resource C cannot be dependent on resource A.

When you add a dependency to a resource definition, you provide the following information:

- The name of the existing resource to which you are adding a dependency.
- The resource type of the existing resource to which you are adding a dependency.
- The name of the cluster that contains the resource.
- Optionally, the logical node name of the node in the cluster that contains the resource. If specified, resource dependencies are added to the node's definition of the resource. If this is not specified, resource dependencies are added to the cluster-wide resource definition.
- The resource name of the resource dependency.
- The resource type of the resource dependency.

5.5.2.1 Defining a Resource with the Cluster Manager GUI

To define a resource with the Cluster Manager GUI, perform the following steps:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the "Resources & Resource Types" category.
3. On the right side of the display click on the "Define a New Resource" task link to launch the task.
4. Enter the selected inputs.
5. Click on "OK" at the bottom of the screen to complete the task.
6. On the right side of the display, click on the "Add/Remove Dependencies for a Resource Definition" to launch the task.
7. Enter the selected inputs.
8. Click on "OK" at the bottom of the screen to complete the task.

When you use this command to define a resource, you define a cluster-wide resource that is not specific to a node. For information on defining a node-specific resource, see Section 5.5.3, *Defining a Node-Specific Resource*.

5.5.2.2 Defining a Resource with the Cluster Manager CLI

Use the following CLI command to define a clusterwide resource:

```
cmgr> define resource A [of resource_type B] [in cluster C]
```

Entering this command specifies the name and resource type of the resource you are defining within a specified cluster. If you have specified a default cluster or a default resource type, you do not need to specify a resource type or a cluster in this command and the CLI will use the default.

When you use this command to define a resource, you define a clusterwide resource that is not specific to a node. For information on defining a node-specific resource, see Section 5.5.3, *Defining a Node-Specific Resource*.

The following prompt appears:

```
resource A?
```

When this prompt appears during resource creation, you can enter the following commands to specify the attributes of the resource you are defining and to add and remove dependencies from the resource:

```
resource A? set key to value
resource A? add dependency E of type F
resource A? remove dependency E of type F
```

The attributes you define with the `set` *key* `to` *value* command will depend on the type of resource you are defining, as described in Section 5.5.1, *Defining Resources*.

For detailed information on how to determine the format for defining resource attributes, see Section 5.5.2.3, *Specifying Resource Attributes with Cluster Manager CLI*.

When you are finished defining the resource and its dependencies, enter `done` to return to the `cmgr` prompt.

5.5.2.3 Specifying Resource Attributes with Cluster Manager CLI

To see the format in which you can specify the user-specific attributes that you need to set for a particular resource type, you can enter the following command to see the full definition of that resource type:

```
cmgr> show resource_type A in cluster B
```

For example, to see the *key* attributes you define for a resource of a defined resource type `IP_address`, you would enter the following command:

```
cmgr>  show resource_type IP_address in cluster nfs-cluster

Name: IP_address
Predefined: true
Order: 401
Restart mode: 1
Restart count: 2

Action name: stop
        Executable: /usr/lib/failsafe/resource_types/IP_address/stop
        Maximum execution time: 80000ms
        Monitoring interval: 0ms
        Start monitoring time: 0ms
Action name: exclusive
        Executable: /usr/lib/failsafe/resource_types/IP_address/exclusive
        Maximum execution time: 100000ms
        Monitoring interval: 0ms
        Start monitoring time: 0ms
Action name: start
        Executable: /usr/lib/failsafe/resource_types/IP_address/start
        Maximum execution time: 80000ms
        Monitoring interval: 0ms
        Start monitoring time: 0ms
Action name: restart
        Executable: /usr/lib/failsafe/resource_types/IP_address/restart
```

```
        Maximum execution time: 80000ms
        Monitoring interval: 0ms
        Start monitoring time: 0ms
Action name: monitor
        Executable: /usr/lib/failsafe/resource_types/IP_address/monitor
        Maximum execution time: 40000ms
        Monitoring interval: 20000ms
        Start monitoring time: 50000ms

Type specific attribute: NetworkMask
        Data type: string
Type specific attribute: interfaces
        Data type: string
Type specific attribute: BroadcastAddress
        Data type: string

No resource type dependencies
```

The display reflects the format in which you can specify the group id, the device owner, and the device file permissions for the volume. In this case, the `devname-group` key specifies the group id of the device file, the `devname_owner` key specifies the owner of the device file, and the `devname_mode` key specifies the device file permissions.

For example, to set the group id to `sys`, enter the following command:

```
resource A? set devname-group to sys
```

This remainder of this section summarizes the attributes you specify for the predefined Linux FailSafe resource types with the *set key to value* command of the Cluster Manger CLI.

When you define an IP address, you specify the following attributes:

`NetworkMask`

The subnet mask of the IP address

`interfaces`

A comma-separated list of interfaces on which the IP address can be configured

`BroadcastAddress`

The broadcast address for the IP address

5.5.3 Defining a Node-Specific Resource

You can redefine an existing resource with a resource definition that applies only to a particular node. Only existing clusterwide resources can be redefined; resources already defined for a specific cluster node cannot be redefined.

You use this feature when you configure heterogeneous clusters for an `IP_address` resource. For example, `IP_address` 192.26.50.2 can be configured on et0 on an SGI Challenge node and on eth0 on all other Linux servers. The clusterwide resource definition for 192.26.50.2 will have the `interfaces` field set to eth0 and the node-specific definition for the Challenge node will have et0 as the `interfaces` field.

5.5.3.1 Defining a Node-Specific Resource with the Cluster Manager GUI

Using the Cluster Manager GUI, you can take an existing clusterwide resource definition and redefine it for use on a specific node in the cluster:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the "Resources & Resource Types" category.
3. On the right side of the display click on the "Redefine a Resource For a Specific Node" task link to launch the task.
4. Enter the selected inputs.
5. Click on "OK" at the bottom of the screen to complete the task.

5.5.3.2 Defining a Node-Specific Resource with the Cluster Manager CLI

You can use the Cluster Manager CLI to redefine a clusterwide resource to be specific to a node just as you define a clusterwide resource, except that you specify a node on the `define resource` command.

Use the following CLI command to define a node-specific resource:

```
cmgr> define resource A of resource_type B on node C [in cluster D]
```

If you have specified a default cluster, you do not need to specify a cluster in this command and the CLI will use the default.

5.5.4 Modifying and Deleting Resources

After you have defined resources, you can modify and delete them.

You can modify only the type-specific attributes for a resource. You cannot rename a resource once it has been defined.

There are some resource attributes whose modification does not take effect until the resource group containing that resource is brought online again. For example, if you modify the export options of a resource of type NFS, the modifications do not take effect immediately; they take effect when the resource is brought online.

5.5.4.1 Modifying and Deleting Resources with the Cluster Manager GUI

To modify a resource with the Cluster Manager GUI, perform the following procedure:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the "Resources & Resource Types" category.
3. On the right side of the display click on the "Modify a Resource Definition" task link to launch the task.
4. Enter the selected inputs.

5. Click on “OK” at the bottom of the screen to complete the task, or click on “Cancel” to cancel.

To delete a resource with the Cluster Manager GUI, perform the following procedure:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the “Resources & Resource Types” category.
3. On the right side of the display click on the “Delete a Resource” task link to launch the task.
4. Enter the selected inputs.
5. Click on “OK” at the bottom of the screen to complete the task, or click on “Cancel” to cancel.

5.5.4.2 Modifying and Deleting Resources with the Cluster Manager CLI

Use the following CLI command to modify a resource:

```
cmgr> modify resource A of resource_type B [in cluster C]
```

Entering this command specifies the name and resource type of the resource you are modifying within a specified cluster. If you have specified a default cluster, you do not need to specify a cluster in this command and the CLI will use the default.

You modify a resource using the same commands you use to define a resource.

You can use the following command to delete a resource definition:

```
cmgr> delete resource A of resource_type B [in cluster D]
```

5.5.5 Displaying Resources

You can display resources in various ways. You can display the attributes of a particular defined resource, you can display all of the defined resources in a specified resource group, or you can display all the defined resources of a specified resource type.

5.5.5.1 Displaying Resources with the Cluster Manager GUI

The Cluster Manager GUI provides a convenient display of resources through the FailSafe Cluster View. You can launch the FailSafe Cluster View directly, or you can bring it up at any time by clicking on the “FailSafe Cluster View” button at the bottom of the “FailSafe Manager” display.

From the View menu of the FailSafe Cluster View, select Resources to see all defined resources. The status of these resources will be shown in the icon (green indicates online, grey indicates offline). Alternately, you can select “Resources of Type” from the View menu to see resources organized by resource type, or you can select “Resources by Group” to see resources organized by resource group.

5.5.5.2 Displaying Resources with the Cluster Manager CLI

Use the following command to view the parameters of a defined resource:

```
cmgr> show resource A of resource_type B
```

Use the following command to view all of the defined resources in a resource group:

```
cmgr> show resources in resource_group A [in cluster B]
```

If you have specified a default cluster, you do not need to specify a cluster in this command and the CLI will use the default.

Use the following command to view all of the defined resources of a particular resource type in a specified cluster:

```
cmgr> show resources of resource_type A [in cluster B]
```

If you have specified a default cluster, you do not need to specify a cluster in this command and the CLI will use the default.

5.5.6 Defining a Resource Type

The Linux FailSafe software includes many predefined resource types. If these types fit the application you want to make into a highly available service, you can reuse them. If none fits, you can define additional resource types.

Complete information on defining resource types is provided in the *Linux FailSafe Programmer's Guide*. This manual provides a summary of that information.

To define a new resource type, you must have the following information:

- Name of the resource type, with a maximum length of 255 characters.
- Name of the cluster to which the resource type will apply.
- Node on which the resource type will apply, if the resource type is to be restricted to a specific node.
- Order of performing the action scripts for resources of this type in relation to resources of other types:
  - Resources are started in the increasing order of this value
  - Resources are stopped in the decreasing order of this value

    See the *Linux FailSafe Programmer's Guide* for a full description of the order ranges available.
- Restart mode, which can be one of the following values:
  - 0 = Do not restart on monitoring failures
  - 1 = Restart a fixed number of times
- Number of local restarts (when restart mode is 1).
- Location of the executable script. This is always `/usr/lib/failsafe/resources_types/`*rtname*, where *rtname* is the resource type name.
- Monitoring interval, which is the time period (in milliseconds) between successive executions of the `monitor` action script; this is only valid for the `monitor` action script.
- Starting time for monitoring. When the resource group is made in online in a cluster node, Linux FailSafe will start monitoring the resources after the specified time period (in milliseconds).
- Action scripts to be defined for this resource type, You must specify scripts for `start`, `stop`, `exclusive`, and `monitor`, although the `monitor` script may contain only a

return-success function if you wish. If you specify 1 for the restart mode, you must specify a `restart` script.

- Type-specific attributes to be defined for this resource type. The action scripts use this information to start, stop, and monitor a resource of this resource type. For example, NFS requires the following resource keys:
  - `export-point`, which takes a value that defines the export disk name. This name is used as input to the `exportfs` command. For example:

    ```
    export-point = /this_disk
    ```

  - `export-info`, which takes a value that defines the export options for the filesystem. These options are used in the `exportfs` command. For example:

    ```
    export-info = rw,sync,no_root_squash
    ```

  - `filesystem`, which takes a value that defines the raw filesystem. This name is used as input to the `mount`) command. For example:

    ```
    filesystem = /dev/sda1
    ```

To define a new resource type, you use the Cluster Manager GUI or the Cluster Manager CLI.

5.5.6.1 Defining a Resource Type with the Cluster Manager GUI

To define a resource type with the Cluster Manager GUI, perform the following steps:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the “Resources & Resource Types” category.
3. On the right side of the display click on the “Define a Resource Type” task link to launch the task.
4. Enter the selected inputs.
5. Click on “OK” at the bottom of the screen to complete the task.

5.5.6.2 Defining a Resource Type with the Cluster Manager CLI

The following steps show the use of `cluster_mgr` interactively to define a resource type called `test_rt`.

1. Log in as `root`.
2. Execute the `cluster_mgr` command using the `-p` option to prompt you for information (the command name can be abbreviated to `cmgr`):

   ```
   # /usr/lib/failsafe/bin/cluster_mgr -p
   Welcome to Linux FailSafe Cluster Manager Command-Line Interface

   cmgr>
   ```

3. Use the `set` subcommand to specify the default cluster used for `cluster_mgr` operations. In this example, we use a cluster named `test`:

   ```
   cmgr> set cluster test
   ```

If you prefer, you can specify the cluster name as needed with each subcommand.

---

4. Use the `define resource_type` subcommand. By default, the resource type will apply across the cluster; if you wish to limit the resource_type to a specific node, enter the node name when prompted. If you wish to enable restart mode, enter 1 when prompted.

The following example only shows the prompts and answers for two action scripts (`start` and `stop`) for a new resource type named `test_rt`.

```
cmgr> define resource_type test_rt

(Enter "cancel" at any time to abort)

Node[optional]?
Order ? 300
Restart Mode ? (0)

DEFINE RESOURCE TYPE OPTIONS

        0) Modify Action Script.
        1) Add Action Script.
        2) Remove Action Script.
        3) Add Type Specific Attribute.
        4) Remove Type Specific Attribute.
        5) Add Dependency.
        6) Remove Dependency.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:1

No current resource type actions

Action name ? start
Executable Time? 40000
Monitoring Interval? 0
Start Monitoring Time? 0

        0) Modify Action Script.
        1) Add Action Script.
        2) Remove Action Script.
        3) Add Type Specific Attribute.
        4) Remove Type Specific Attribute.
        5) Add Dependency.
        6) Remove Dependency.
```

```
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:1

Current resource type actions:
        Action - 1: start

Action name stop
Executable Time? 40000
Monitoring Interval? 0
Start Monitoring Time? 0

        0) Modify Action Script.
        1) Add Action Script.
        2) Remove Action Script.
        3) Add Type Specific Attribute.
        4) Remove Type Specific Attribute.
        5) Add Dependency.
        6) Remove Dependency.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:3

No current type specific attributes

Type Specific Attribute ? integer-att
Datatype ? integer
Default value[optional] ? 33

        0) Modify Action Script.
        1) Add Action Script.
        2) Remove Action Script.
        3) Add Type Specific Attribute.
        4) Remove Type Specific Attribute.
        5) Add Dependency.
        6) Remove Dependency.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:3

Current type specific attributes:
        Type Specific Attribute - 1: export-point

Type Specific Attribute ? string-att
Datatype ? string
Default value[optional] ? rw
```

```
        0) Modify Action Script.
        1) Add Action Script.
        2) Remove Action Script.
        3) Add Type Specific Attribute.
        4) Remove Type Specific Attribute.
        5) Add Dependency.
        6) Remove Dependency.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)Enter option:5

No current resource type dependencies

Dependency name ? filesystem

        0) Modify Action Script.
        1) Add Action Script.
        2) Remove Action Script.
        3) Add Type Specific Attribute.
        4) Remove Type Specific Attribute.
        5) Add Dependency.
        6) Remove Dependency.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:7

Current resource type actions:
        Action - 1: start
        Action - 2: stop

Current type specific attributes:
        Type Specific Attribute - 1: integer-att
        Type Specific Attribute - 2: string-att

No current resource type dependencies

Resource dependencies to be added:
        Resource dependency - 1: filesystem

        0) Modify Action Script.
        1) Add Action Script.
        2) Remove Action Script.
        3) Add Type Specific Attribute.
        4) Remove Type Specific Attribute.
        5) Add Dependency.
        6) Remove Dependency.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:9
```

```
Successfully created resource_type test_rt

cmgr> show resource_types

NFS
template
Netscape_web
test_rt
statd
Oracle_DB
MAC_address
IP_address
INFORMIX_DB
filesystem
volume

cmgr> exit
#
```

5.5.7 Defining a Node-Specific Resource Type

You can redefine an existing resource type with a resource definition that applies only to a particular node. Only existing clusterwide resource types can be redefined; resource types already defined for a specific cluster node cannot be redefined.

A resource type that is defined for a node overrides a cluster-wide resource type definition with the same name; this allows an individual node to override global settings from a clusterwide resource type definition. You can use this feature if you want to have different script timeouts for a node or you want to restart a resource on only one node in the cluster.

For example, the `IP_address` resource has local restart enabled by default. If you would like to have an IP address type without local restart for a particular node, you can make a copy of the `IP_address` clusterwide resource type with all of the parameters the same except for restart mode, which you set to 0.

5.5.7.1 Defining a Node-Specific Resource Type with the Cluster Manager GUI

Using the Cluster Manager GUI, you can take an existing clusterwide resource type definition and redefine it for use on a specific node in the cluster. Perform the following tasks:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the “Resources & Resource Types” category.
3. On the right side of the display click on the “Redefine a Resource Type For a Specific Node” task link to launch the task.
4. Enter the selected inputs.
5. Click on “OK” at the bottom of the screen to complete the task.

5.5.7.2 Defining a Node-Specific Resource Type with the Cluster Manager CLI

With the Cluster Manager CLI, you redefine a node-specific resource type just as you define a cluster-wide resource type, except that you specify a node on the `define resource_type` command.

Use the following CLI command to define a node-specific resource type:

```
cmgr> define resource_type A on node B [in cluster C]
```

If you have specified a default cluster, you do not need to specify a cluster in this command and the CLI will use the default.

5.5.8 Adding Dependencies to a Resource Type

Like resources, a resource type can be dependent on one or more other resource types. If such a dependency exists, at least one instance of each of the dependent resource types must be defined. For example, a resource type named `Netscape_web` might have resource type dependencies on a resource type named `IP_address` and `volume`. If a resource named `ws1` is defined with the `Netscape_web` resource type, then the resource group containing `ws1` must also contain at least one resource of the type `IP_address` nd one resource of the type `volume`.

When using the Cluster Manager GUI, you add or remove dependencies for a resource type by selecting the "Add/Remove Dependencies for a Resource Type" from the "Resources & Resource Types" display and providing the indicated input. When using the Cluster Manager CLI, you add or remove dependencies when you define or modify the resource type.

5.5.9 Modifying and Deleting Resource Types

After you have defined a resource types, you can modify and delete them.

5.5.9.1 Modifying and Deleting Resource Types with the Cluster Manager GUI

To modify a resource type with the Cluster Manager GUI, perform the following procedure:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the "Resources & Resource Types" category.
3. On the right side of the display click on the "Modify a Resource Type Definition" task link to launch the task.
4. Enter the selected inputs.
5. Click on "OK" at the bottom of the screen to complete the task, or click on "Cancel" to cancel.

To delete a resource type with the Cluster Manager GUI, perform the following procedure:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the "Resources & Resource Types" category.
3. On the right side of the display click on the "Delete a Resource Type" task link to launch the task.
4. Enter the selected inputs.
5. Click on "OK" at the bottom of the screen to complete the task, or click on "Cancel" to cancel.

5.5.9.2 Modifying and Deleting Resource Types with the Cluster Manager CLI

Use the following CLI command to modify a resource:

```
cmgr> modify resource_type A [in cluster B]
```

Entering this command specifies the resource type you are modifying within a specified cluster. If you have specified a default cluster, you do not need to specify a cluster in this command and the CLI will use the default.

You modify a resource type using the same commands you use to define a resource type.

You can use the following command to delete a resource type:

```
cmgr> delete resource_type A [in cluster B]
```

5.5.10 Installing (Loading) a Resource Type on a Cluster

When you define a cluster, Linux FailSafe installs a set of resource type definitions that you can use that include default values. If you need to install additional standard Silicon Graphics-supplied resource type definitions on the cluster, or if you delete a standard resource type definition and wish to reinstall it, you can load that resource type definition on the cluster.

The resource type definition you are installing cannot exist on the cluster.

5.5.10.1 Installing a Resource Type with the Cluster Manager GUI

To install a resource type using the GUI, select the “Load a Resource” task from the “Resources & Resource Types” task page and enter the resource type to load.

5.5.10.2 Installing a Resource Type with the Cluster Manager CLI

Use the following CLI command to install a resource type on a cluster:

```
cmgr> install resource_type A [in cluster B]
```

If you have specified a default cluster, you do not need to specify a cluster in this command and the CLI will use the default.

5.5.11 Displaying Resource Types

After you have defined a resource types, you can display them.

5.5.11.1 Displaying Resource Types with the Cluster Manager GUI

The Cluster Manager GUI provides a convenient display of resource types through the FailSafe Cluster View. You can launch the FailSafe Cluster View directly, or you can bring it up at any time by clicking on the “FailSafe Cluster View” prompt at the bottom of the “FailSafe Manager” display.

From the View menu of the FailSafe Cluster View, select Types to see all defined resource types. You can then click on any of the resource type icons to view the parameters of the resource type.

5.5.11.2 Displaying Resource Types with the Cluster Manager CLI

Use the following command to view the parameters of a defined resource type in a specified cluster:

```
cmgr> show resource_type A [in cluster B]
```

If you have specified a default cluster, you do not need to specify a cluster in this command and the CLI will use the default.

Use the following command to view all of the defined resource types in a cluster:

```
cmgr> show resource_types [in cluster A]
```

If you have specified a default cluster, you do not need to specify a cluster in this command and the CLI will use the default.

Use the following command to view all of the defined resource types that have been installed:

```
cmgr> show resource_types installed
```

5.5.12 Defining a Failover Policy

Before you can configure your resources into a resource group, you must determine which failover policy to apply to the resource group. To define a failover policy, you provide the following information:

- The name of the failover policy, with a maximum length of 63 characters, which must be unique within the pool.
- The name of an existing failover script.
- The initial failover domain, which is an ordered list of the nodes on which the resource group may execute. The administrator supplies the initial failover domain when configuring the failover policy; this is input to the failover script, which generates the runtime failover domain.
- The failover attributes, which modify the behavior of the failover script.

Complete information on failover policies and failover scripts, with an emphasis on writing your own failover policies and scripts, is provided in the *Linux FailSafe Programmer's Guide*.

5.5.12.1 Failover Scripts

A **failover script** helps determine the node that is chosen for a failed resource group. The failover script takes the initial failover domain and transforms it into the runtime failover domain. Depending upon the contents of the script, the initial and the runtime domains may be identical.

The `ordered` failover script is provided with the Linux FailSafe release. The `ordered` script never changes the initial domain; when using this script, the initial and runtime domains are equivalent.

The `round-robin` failover script is also provided with the Linux FailSafe release. The `round-robin` cript selects the resource group owner in a round-robin (circular) fashion. This policy can be used for resource groups that can be run in any node in the cluster.

Failover scripts are stored in the `/usr/lib/failsafe/policies` directory. If the `ordered` script does not meet your needs, you can define a new failover script and place it in the `/usr/lib/failsafe/policies` directory. When you are using the FailSafe GUI, the GUI automatically detects your script and presents it to you as a choice for you to use. You can configure the Linux FailSafe database to use your new failover script for the required resource groups. For information on defining failover scripts, see the *Linux FailSafe Programmer's Guide*.

5.5.12.2 Failover Domain

A **failover domain** is the ordered list of nodes on which a given resource group can be allocated. The nodes listed in the failover domain must be within the same cluster; however, the failover

domain does not have to include every node in the cluster. The failover domain can be used to statically load balance the resource groups in a cluster.

Examples:

- In a four-node cluster, two nodes might share a volume. The failover domain of the resource group containing the volume will be the two nodes that share the volume.
- If you have a cluster of nodes named venus, mercury, and pluto, you could configure the following initial failover domains for resource groups RG1 and RG2:
  - venus, mercury, pluto for RG1
  - pluto, mercury for RG2

When you define a failover policy, you specify the **initial failover domain**. The initial failover domain is used when a cluster is first booted. The ordered list specified by the initial failover domain is transformed into a **runtime failover domain** by the failover script. With each failure, the failover script takes the current run-time failover domain and potentially modifies it; the initial failover domain is never used again. Depending on the run-time conditions and contents of the failover script, the initial and run-time failover domains may be identical.

Linux FailSafe stores the run-time failover domain and uses it as input to the next failover script invocation.

5.5.12.3 Failover Attributes

A failover attribute is a value that is passed to the failover script and used by Linux FailSafe for the purpose of modifying the run-time failover domain used for a specific resource group. You can specify a failover attribute of `Auto_Failback`, `Controlled_Failback`, `Auto_Recovery`, or `InPlace_Recovery`. `Auto_Failback` and `Controlled_Failback` are mutually exclusive, but you must specify one or the other. `Auto_Recovery` and `InPlace_Recovery` are mutually exclusive, but whether you specify one or the other is optional.

A failover attribute of `Auto_Failback` specifies that the resource group will be run on the first available node in the runtime failover domain. If the first node fails, the next available node will be used; when the first node reboots, the resource group will return to it. This attribute is best used when some type of load balancing is required.

A failover attribute of `Controlled_Failback` specifies that the resource group will be run on the first available node in the runtime failover domain, and will remain running on that node until it fails. If the first node fails, the next available node will be used; the resource group will remain on this new node even after the first node reboots.This attribute is best used when client/server applications have expensive recovery mechanisms, such as databases or any application that uses `tcp` to communicate.

The recovery attributes `Auto_Recovery` and `InPlace_Recovery` determine the node on which a resource group will be allocated when its state changes to online and a member of the group is already allocated (such as when volumes are present). `Auto_Recovery` specifies that the failover policy will be used to allocate the resource group; this is the default recovery attribute if you have specified the `Auto_Failback` attribute. `InPlace_Recovery` specifies that the resource group will be allocated on the node that already contains part of the resource group; this is the default recovery attribute if you have specified the `Controlled_Failback` attribute.

See the *Linux FailSafe Programmer's Guide* for a full discussions of example failover policies.

5.5.12.4 Defining a Failover Policy with the Cluster Manager GUI

To define a failover policy using the GUI, perform the following steps:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the "Failover Policies & Resource Groups" category.
3. On the right side of the display click on the "Define a Failover Policy" task link to launch the task.
4. Enter the selected inputs.
5. Click on "OK" at the bottom of the screen to complete the task.

5.5.12.5 Defining a Failover Policy with the Cluster Manager CLI

To define a failover policy, enter the following command at the `cmgr` prompt to specify the name of the failover policy:

```
cmgr> define failover_policy A
```

The following prompt appears:

```
failover_policy A?
```

When this prompt appears you can use the following commands to specify the components of a failover policy:

```
failover_policy A? set attribute to B
failover policy A? set script to C
failover policy A? set domain to D
failover_policy A?
```

When you define a failover policy, you can set as many attributes and domains as your setup requires, but executing the `add attribute` and `add domain` commands with different values. The CLI also allows you to specify multiple domains in one command of the following format:

```
failover_policy A? set domain to A B C ...
```

The components of a failover policy are described in detail in the *Linux FailSafe Programmer's Guide* and in summary in Section 5.5.12, *Defining a Failover Policy*.

When you are finished defining the failover policy, enter `done` to return to the `cmgr` prompt.

5.5.13 Modifying and Deleting Failover Policies

After you have defined a failover policy, you can modify or delete it.

5.5.13.1 Modifying and Deleting Failover Policies with the Cluster Manager GUI

To modify a failover policy with the Cluster Manager GUI, perform the following procedure:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the "Failover Policies & Resource Groups" category.

3. On the right side of the display click on the “Modify a Failover Policy Definition” task link to launch the task.
4. Enter the selected inputs.
5. Click on “OK” at the bottom of the screen to complete the task, or click on “Cancel” to cancel.

To delete a failover policy with the Cluster Manager GUI, perform the following procedure:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the “Failover Policies & Resource Groups” category.
3. On the right side of the display click on the “Delete a Failover Policy” task link to launch the task.
4. Enter the selected inputs.
5. Click on “OK” at the bottom of the screen to complete the task, or click on “Cancel” to cancel.

5.5.13.2 Modifying and Deleting Failover Policies with the Cluster Manager CLI

Use the following CLI command to modify a failover policy:

```
cmgr> modify failover_policy A
```

You modify a failover policy using the same commands you use to define a failover policy.

You can use the following command to delete a failover policy definition:

```
cmgr> delete failover_policy A
```

5.5.14 Displaying Failover Policies

You can use Linux FailSafe to display any of the following:

- The components of a specified failover policy
- All of the failover policies that have been defined
- All of the failover policy attributes that have been defined
- All of the failover policy scripts that have been defined

5.5.14.1 Displaying Failover Policies with the Cluster Manager GUI

The Cluster Manager GUI provides a convenient display of failover policies through the FailSafe Cluster View. You can launch the FailSafe Cluster View directly, or you can bring it up at any time by clicking on the “FailSafe Cluster View” prompt at the bottom of the “FailSafe Manager” display.

From the View menu of the FailSafe Cluster View, select Failover Policies to see all defined failover policies.

5.5.14.2 Displaying Failover Policies with the Cluster Manager CLI

Use the following command to view the parameters of a defined failover policy:

```
cmgr> show failover_policy A
```

Use the following command to view all of the defined failover policies:

```
cmgr> show failover policies
```

Use the following command to view all of the defined failover policy attributes:

```
cmgr> show failover_policy attributes
```

Use the following command to view all of the defined failover policy scripts:

```
cmgr> show failover_policy scripts
```

5.5.15 Defining Resource Groups

Resources are configured together into **resource group**s. A resource group is a collection of interdependent resources. If any individual resource in a resource group becomes unavailable for its intended use, then the entire resource group is considered unavailable. Therefore, a resource group is the unit of failover for Linux FailSafe.

For example, a resource group could contain all of the resources that are required for the operation of a web node, such as the web node itself, the IP address with which it communicates to the outside world, and the disk volumes containing the content that it serves.

When you define a resource group, you specify a **failover policy**. A failover policy controls the behavior of a resource group in failure situations.

To define a resource group, you provide the following information:

- The name of the resource group, with a maximum length of 63 characters.
- The name of the cluster to which the resource group is available
- The resources to include in the resource group, and their resource types
- The name of the failover policy that determines which node will take over the services of the resource group on failure

Linux FailSafe does not allow resource groups that do not contain any resources to be brought online.

You can define up to 100 resources configured in any number of resource groups.

5.5.15.1 Defining a Resource Group with the Cluster Manager GUI

To define a resource group with the Cluster Manager GUI, perform the following steps:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on “Guided Configuration”.
3. On the right side of the display click on “Set Up Highly Available Resource Groups” to launch the task link.
4. In the resulting window, click each task link in turn, as it becomes available. Enter the selected inputs for each task.
5. When finished, click “OK” to close the taskset window.

5.5.15.2 Defining a Resource Group with the Cluster Manager CLI

To configure a resource group, enter the following command at the `cmgr` prompt to specify the name of a resource group and the cluster to which the resource group is available:

```
cmgr> define resource_group A [in cluster B]
```

Entering this command specifies the name of the resource group you are defining within a specified cluster. If you have specified a default cluster, you do not need to specify a cluster in this command and the CLI will use the default.

The following prompt appears:

```
Enter commands, when finished enter either "done" or "cancel"
resource_group A?
```

When this prompt appears you can use the following commands to specify the resources to include in the resource group and the failover policy to apply to the resource group:

```
resource_group A? add resource B of resource_type C
resource_group A? set failover_policy to D
```

After you have set the failover policy and you have finished adding resources to the resource group, enter `done` to return to the `cmgr` prompt.

For a full example of resource group creation using the Cluster Manager CLI, see Section 5.7, *Resource Group Creation Example*.

5.5.16 Modifying and Deleting Resource Groups

After you have defined resource groups, you can modify and delete the resource groups. You can change the failover policy of a resource group by specifying a new failover policy associated with that resource group, and you can add or delete resources to the existing resource group. Note, however, that since you cannot have a resource group online that does not contain any resources, Linux FailSafe does not allow you to delete all resources from a resource group once the resource group is online. Likewise, Linux FailSafe does not allow you to bring a resource group online if it has no resources. Also, resources must be added and deleted in atomic units; this means that resources which are interdependent must be added and deleted together.

5.5.16.1 Modifying and Deleting Resource Groups with the Cluster Manager GUI

To modify a failure policy with the Cluster Manager GUI, perform the following procedure:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the “Failover Policies & Resource Groups” category.
3. On the right side of the display click on the “Modify a Resource Group Definition” task link to launch the task.
4. Enter the selected inputs.
5. Click on “OK” at the bottom of the screen to complete the task, or click on “Cancel” to cancel.

To add or delete resources to a resource group definition with the Cluster Manager GUI, perform the following procedure:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the “Failover Policies & Resource Groups” category.
3. On the right side of the display click on the “Add/Remove Resources in Resource Group” task link to launch the task.

4. Enter the selected inputs.
5. Click on “OK” at the bottom of the screen to complete the task, or click on “Cancel” to cancel.

To delete a resource group with the Cluster Manager GUI, perform the following procedure:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the “Failover Policies & Resource Groups” category.
3. On the right side of the display click on the “Delete a Resource Group” task link to launch the task.
4. Enter the selected inputs.
5. Click on “OK” at the bottom of the screen to complete the task, or click on “Cancel” to cancel.

5.5.16.2 Modifying and Deleting Resource Groups with the Cluster Manager CLI

Use the following CLI command to modify a resource group:

```
cmgr> modify resource_group A [in cluster B]
```

If you have specified a default cluster, you do not need to specify a cluster in this command and the CLI will use the default. You modify a resource group using the same commands you use to define a failover policy:

```
resource_group A? add resource B of resource_type C
resource_group A? set failover_policy to D
```

You can use the following command to delete a resource group definition:

```
cmgr> delete resource_group A [in cluster B]
```

If you have specified a default cluster, you do not need to specify a cluster in this command and the CLI will use the default.

5.5.17 Displaying Resource Groups

You can display the parameters of a defined resource group, and you can display all of the resource groups defined for a cluster.

5.5.17.1 Displaying Resource Groups with the Cluster Manager GUI

The Cluster Manager GUI provides a convenient display of resource groups through the FailSafe Cluster View. You can launch the FailSafe Cluster View directly, or you can bring it up at any time by clicking on the “FailSafe Cluster View” prompt at the bottom of the “FailSafe Manager” display.

From the View menu of the FailSafe Cluster View, select Groups to see all defined resource groups.

To display which nodes are currently running which groups, select “Groups owned by Nodes.” To display which groups are running which failover policies, select “Groups by Failover Policies.”

5.5.17.2 Displaying Resource Groups with the Cluster Manager CLI

Use the following command to view the parameters of a defined resource group:

```
cmgr> show resource_group A [in cluster B]
```

If you have specified a default cluster, you do not need to specify a cluster in this command and the CLI will use the default.

Use the following command to view all of the defined failover policies:

```
cmgr> show resource_groups [in cluster A]
```

5.6 Linux FailSafe System Log Configuration

Linux FailSafe maintains system logs for each of the Linux FailSafe daemons. You can customize the system logs according to the level of logging you wish to maintain.

A log group is a set of processes that log to the same log file according to the same logging configuration. All Linux FailSafe daemons make one log group each. Linux FailSafe maintains the following log groups:

cli

Commands log

crsd

Cluster reset services (`crsd`) log

diags

Diagnostics log

ha_agent

HA monitoring agents (`ha_ifmx2`) log

ha_cmsd

Cluster membership daemon (`ha_cmsd`) log

ha_fsd

Linux FailSafe daemon (`ha_fsd`) log

ha_gcd

Group communication daemon (`ha_gcd`) log

ha_ifd

network interface monitoring daemon (`ha_ifd`) log

ha_script

Action and Failover policy scripts log

ha_srmd

System resource manager (`ha_srmd`) log

Log group configuration information is maintained for all nodes in the pool for the `cli` and `crsd` log groups or for all nodes in the cluster for all other log groups.You can also customize the log group configuration for a specific node in the cluster or pool.

When you configure a log group, you specify the following information:

- The log level, specified as character strings with the CUI and numerically (1 to 19) with the CLI, as described below
- The log file to log to
- The node whose specified log group you are customizing (optional)

The log level specifies the verbosity of the logging, controlling the amount of log messages that Linux FailSafe will write into an associated log group's file. There are 10 debug level. Table 5–1, *Log Levels*, shows the logging levels as you specify them with the GUI and the CLI.

Table 5–1 Log Levels

GUI level	CLI level	Meaning
`Off`	0	No logging
`Minimal`	1	Logs notification of critical errors and normal operation
`Info`	2	Logs minimal notification plus warning
`Default`	5	Logs all Info messages plus additional notifications
`Debug0`	10	
...		`Debug0` through `Debug9` (11 -19 in CLI) log increasingly more debug information, including data structures. Many megabytes of disk space can be consumed on the server when debug levels are used in a log configuration.
`Debug9`	19	

Notifications of critical errors and normal operations are always sent to `/var/log/failsafe/`. Changes you make to the log level for a log group do not affect `SYSLOG`.

The Linux FailSafe software appends the node name to the name of the log file you specify. For example, when you specify the log file name for a log group as `/var/log/failsafe/cli`, the file name will be `/var/log/failsafe/cli_nodename`.

The default log file names are as follows.

`/var/log/failsafe/cmsd`_*nodename*

log file for cluster membership services daemon in node *nodename*

`/var/log/failsafe/gcd`_*nodename*

log file for group communication daemon in node *nodename*

`/var/log/failsafe/srmd`_*nodename*

log file for system resource manager daemon in node *nodename*

`/var/log/failsafe/failsafe`_ *nodename*

log file for Linux FailSafe daemon, a policy implementor for resource groups, in node *nodename*

`/var/log/failsafe/`*agent_nodename*

log file for monitoring agent named *agent* in node *nodename*. For example, `ifd_`*nodename* is the log file for the interface daemon monitoring agent that monitors interfaces and IP addresses and performs local failover of IP addresses.

`/var/log/failsafe/crsd_`*nodename*

log file for reset daemon in node *nodename*

`/var/log/failsafe/`script_*nodename*

log file for scripts in node *nodename*

`/var/log/failsafe/cli_`*nodename*

log file or internal administrative commands in node *nodename* invoked by the Cluster Manager GUI and Cluster Manager CLI

For information on using log groups in system recovery, see Chapter 9, *Linux FailSafe Recovery*.

5.6.1 Configuring Log Groups with the Cluster Manager GUI

To configure a log group with the Cluster Manager GUI, perform the following steps:

1. Launch the FailSafe Manager.
2. On the left side of the display, click on the “Nodes & Clusters” category.
3. On the right side of the display click on the “Set Log Configuration” task link to launch the task.
4. Enter the selected inputs.
5. Click on “OK” at the bottom of the screen to complete the task.

5.6.2 Configuring Log Groups with the Cluster Manager CLI

You can configure a log group with the following CLI command:

```
cmgr> define log_group A [on node B] [in cluster C]
```

You specify the node if you wish to customize the log group configuration for a specific node only. If you have specified a default cluster, you do not have to specify a cluster in this command; Linux FailSafe will use the default.

The following prompt appears:

```
Enter commands, when finished enter either "done" or "cancel"
log_group A?
```

When this prompt of the node name appears, you enter the log group parameters you wish to modify in the following format:

```
log_group A? set log_level to A
log_group A? add log_file A
log_group A? remove log_file A
```

When you are finished configuring the log group, enter `done` to return to the `cmgr` prompt.

5.6.3 Modifying Log Groups with the Cluster Manager CLI

Use the following CLI command to modify a log group:

```
cmgr> modify log_group A on [node B] [in cluster C]
```

You modify a log group using the same commands you use to define a log group.

5.6.4 Displaying Log Group Definitions with the Cluster Manager GUI

To display log group definitions with the Cluster Manager GUI, run "Set Log Configuration" and choose the log group to display from the rollover menu. The current log level and log file for that log group will be displayed in the task window, where you can change those settings if you desire.

5.6.5 Displaying Log Group Definitions with the Cluster Manager CLI

Use the following command to view the parameters of a defined resource:

```
cmgr> show log_groups
```

This command shows all of the log groups currently defined, with the log group name, the logging levels and the log files.

For information on viewing the contents of the log file, see Chapter 9, *Linux FailSafe Recovery*.

5.7 Resource Group Creation Example

Use the following procedure to create a resource group using the Cluster Manager CLI:

1. Determine the list of resources that belong to the resource group you are defining. The list of resources that belong to a resource group are the resources that move from one node to another as one unit.

   A resource group that provides NFS services would contain a resource of each of the following types:

   - `IP_address`
   - `volume`
   - `filesystem`
   - `NFS`

     All resource and resource type dependencies of resources in a resource group must be satisfied. For example, the `NFS` resource type depends on the `filesystem` resource type, so a resource group containing a resource of `NFS` resource type should also contain a resource of `filesystem` resource type.

2. Determine the failover policy to be used by the resource group.
3. Use the template `cluster_mgr` script available in the `/usr/lib/failsafe/cmgr-templates/cmgr-create-resource_group` file.

This example shows a script that creates a resource group with the following characteristics:

- The resource group is named `nfs-group`
- The resource group is in cluster `HA-cluster`
- The resource group uses the failover policy
- the resource group contains `IP_Address`, `volume`, `filesystem`, and `NFS` resources

The following script can be used to create this resource group:

```
define resource_group nfs-group in cluster HA-cluster
        set failover_policy to n1_n2_ordered
        add resource 192.0.2.34 of resource_type IP_address
        add resource havol1 of resource_type volume
        add resource /hafs1 of resource_type filesystem
        add resource /hafs1 of resource_type NFS
done
```

4. Run this script using the `-f` option of the `cluster_mgr` command.

5.8 Linux FailSafe Configuration Example CLI Script

The following Cluster Manager CLI script provides an example which shows how to configure a cluster in the cluster database. The script illustrates the CLI commands that you execute when you define a cluster. You will use the parameters of your own system when you configure your cluster. After you create a CLI script, you can set the execute permissions and execute the script directly.

For general information on CLI scripts, see Section 4.3.4, *CLI Command Scripts*. For information on the CLI template files that you can use to create your own configuration script, see Section 4.3.5, *CLI Template Scripts*.

```
#!/usr/lib/failsafe/bin/cluster_mgr -f


##############################################################################
#                                                                            #
# Sample cmgr script to create a 2-node cluster in the cluster               #
# database (cdb).                                                            #
# This script is created using cmgr template files under                     #
#          /usr/lib/failsafe/cmgr-scripts directory.                       #
# The cluster has 2 resource groups:                                         #
# 1. nfs-group - Has 2 NFS, 2 filesystem, 2 volume, 1 statd and              #
#                1 IP_address resources.                                     #
# 2. web-group - Has 1 Netscape_web and 1 IP_address resources.              #
#                                                                            #
# NOTE: After running this script to define the cluster in the               #
# cdb, the user has to enable the two resource groups using the              #
# cmgr admin online resource_group command.                                  #
#                                                                            #
##############################################################################

#
```

```
# Create the first node.
# Information to create a node is obtained from template script:
# /usr/lib/failsafe/cmgr-templates/cmgr-create-node
#

#
#
# logical name of the node. It is recommended that logical name of the # node be
output of hostname(1) command.
#
define node sleepy
#
# Hostname of the node. This is optional. If this field is not
# specified,logical name of the node is assumed to be hostname.
# This value has to be
# the output of hostname(1) command.
#
       set hostname to sleepy
#
# Node identifier. Node identifier is a 16 bit integer that uniquely
# identifies the node. This field is optional. If value is
# not provided,cluster software generates node identifier.
# Example value: 1
       set nodeid to 101
#
# Description of the system controller of this node.
# System controller can be ‘‘chalL’’ or ‘‘msc’’ or ‘‘mmsc’’. If the node is a
# Challenge DM/L/XL, then system controller type is ‘‘chalL’’. If the
# node is Origin 200 or deskside Origin 2000, then the system
# controller type is ‘‘msc’’. If the node is rackmount Origin 2000, the
# system controller type is ‘‘mmsc’’.
# Possible values: msc, mmsc, chalL
#
        set sysctrl_type to msc
#
# You can enable or disable system controller definition. Users are
# expected to enable system controller definition after verify the
# serial reset cables connected to this node.
# Possible values: enabled, disabled
#
        set sysctrl_status to enabled
#
# The system controller password for doing privileged system controller
# commands.
# This field is optional.
#
        set sysctrl_password to none
#
# System controller owner. The node name of the machine that is
# connected using serial cables to system controller of this node.
# System controller node also has to be defined in the CDB.
#
        set sysctrl_owner to grumpy
```

```
#
# System controller device. The absolute device path name of the tty
# to which the serial cable is connected in this node.
# Example value: /dev/ttyd2
#
        set sysctrl_device to /dev/ttyd2
#
# Currently, the system controller owner can be connected to the system
# controller on this node using ‘‘tty’’ device.
# Possible value: tty
#
        set sysctrl_owner_type to tty
#
# List of control networks. There can be multiple control networks
# specified for a node. HA cluster software uses these control
# networks for communication between nodes.  At least two control
# networks should be specified for heartbeat messages and one
# control network for failsafe control messages.
# For each control network for the node, please add one more
# control network section.
#
# Name of control network IP address. This IP address must
# be configured on the network interface in /etc/rc.config
# file in the node.
# It is recommended that the IP address in internet dot notation
# is provided.
# Example value: 192.26.50.3
#
        add nic 192.26.50.14
#
# Flag to indicate if the control network can be used for sending
# heartbeat messages.
# Possible values: true, false
#
            set heartbeat to true
#
# Flag to indicate if the control network can be used for sending
# failsafe control messages.
# Possible values: true, false
#
            set ctrl_msgs to true
#
# Priority of the control network. Higher the priority value, lower the
# priority of the control network.
# Example value: 1
#
            set priority to 1
#
# Control network information complete
#
        done
#
# Add more control networks information here.
```

```
#

# Name of control network IP address. This IP address must be
# configured on the network interface in /etc/rc.config
# file in the node.
# It is recommended that the IP address in internet dot
# notation is provided.
# Example value: 192.26.50.3
#
        add nic 150.166.41.60
#
# Flag to indicate if the control network can be used for sending
# heartbeat messages.
# Possible values: true, false
#
            set heartbeat to true
#
# Flag to indicate if the control network can be used for sending
# failsafe control messages.
# Possible values: true, false
#
            set ctrl_msgs to false
#
# Priority of the control network. Higher the priority value, lower the
# priority of the control network.
# Example value: 1
#
            set priority to 2
#
# Control network information complete
#
        done
#
# Node definition complete
#
done


#
# Create the second node.
# Information to create a node is obtained from template script:
#  /usr/lib/failsafe/cmgr-templates/cmgr-create-node
#

#
#
# logical name of the node. It is recommended that logical name of
# the node be output of hostname(1) command.
#
define node grumpy
#
# Hostname of the node. This is optional. If this field is not
# specified,logical name of the node is assumed to be hostname.
```

```
# This value has to be
# the output of hostname(1) command.
#
       set hostname to grumpy
#
# Node identifier. Node identifier is a 16 bit integer that uniquely
# identifies the node. This field is optional. If value is
# not provided,cluster software generates node identifier.
# Example value: 1
       set nodeid to 102
#
# Description of the system controller of this node.
# System controller can be ‘‘chalL’’ or ‘‘msc’’ or ‘‘mmsc’’. If the node is a
# Challenge DM/L/XL, then system controller type is ‘‘chalL’’. If the
# node is Origin 200 or deskside Origin 2000, then the system
# controller type is ‘‘msc’’. If the node is rackmount Origin 2000,
# the system controller type is ‘‘mmsc’’.
# Possible values: msc, mmsc, chalL
#
        set sysctrl_type to msc
#
# You can enable or disable system controller definition. Users are
# expected to enable system controller definition after verify the
# serial reset cables connected to this node.
# Possible values: enabled, disabled
#
        set sysctrl_status to enabled
#
# The system controller password for doing privileged system controller
# commands.
# This field is optional.
#
        set sysctrl_password to none
#
# System controller owner. The node name of the machine that is
# connected using serial cables to system controller of this node.
# System controller node also has to be defined in the CDB.
#
        set sysctrl_owner to sleepy
#
# System controller device. The absolute device path name of the tty
# to which the serial cable is connected in this node.
# Example value: /dev/ttyd2
#
        set sysctrl_device to /dev/ttyd2
#
# Currently, the system controller owner can be connected to the system
# controller on this node using ‘‘tty’’ device.
# Possible value: tty
#
        set sysctrl_owner_type to tty
#
# List of control networks. There can be multiple control networks
```

```
# specified for a node. HA cluster software uses these control
# networks for communication between nodes.  At least two control
# networks should be specified for heartbeat messages and one
# control network for failsafe control messages.
# For each control network for the node, please add one more
# control network section.
#
# Name of control network IP address. This IP address must be
# configured on the network interface in /etc/rc.config
# file in the node.
# It is recommended that the IP address in internet dot notation
# is provided.
# Example value: 192.26.50.3
#
        add nic 192.26.50.15
#
# Flag to indicate if the control network can be used for sending
# heartbeat messages.
# Possible values: true, false
#
            set heartbeat to true
#
# Flag to indicate if the control network can be used for sending
# failsafe control messages.
# Possible values: true, false
#
            set ctrl_msgs to true
#
# Priority of the control network. Higher the priority value, lower the
# priority of the control network.
# Example value: 1
#
            set priority to 1
#
# Control network information complete
#
        done
#
# Add more control networks information here.
#

# Name of control network IP address. This IP address must be
# configured on the network interface in /etc/rc.config
# file in the node.
# It is recommended that the IP address in internet dot notation
# is provided.
# Example value: 192.26.50.3
#
        add nic 150.166.41.61
#
# Flag to indicate if the control network can be used for sending
# heartbeat messages.
# Possible values: true, false
```

```
#
            set heartbeat to true
#
# Flag to indicate if the control network can be used for sending
# failsafe control messages.
# Possible values: true, false
#
            set ctrl_msgs to false
#
# Priority of the control network. Higher the priority value, lower the
# priority of the control network.
# Example value: 1
#
            set priority to 2
#
# Control network information complete
#
        done
#
# Node definition complete
#
done


#
# Define (create) the cluster.
# Information to create the cluster is obtained from template script:
#       /usr/lib/failsafe/cmgr-templates/cmgr-create-cluster
#

#
# Name of the cluster.
#
define cluster failsafe-cluster
#
# Notification command for the cluster. This is optional. If this
# field is not specified,  /usr/bin/mail command is used for
# notification. Notification is sent when there is change in status of
# cluster, node and resource group.
#
       set notify_cmd to /usr/bin/mail
#
# Notification address for the cluster. This field value is passed as
# argument to the notification command. Specifying the notification
# command is optional and user can specify only the notification
# address in order to receive notifications by mail. If address is
# not specified, notification will not be sent.
# Example value: failsafe_alias@sysadm.company.com
       set notify_addr to robinhood@sgi.com princejohn@sgi.com
#
# List of nodes added to the cluster.
# Repeat the following line for each node to be added to the cluster.
# Node should be already defined in the CDB and logical name of the
```

```
# node has to be specified.
        add node sleepy
#
# Add more nodes to the cluster here.
#
        add node grumpy

#
# Cluster definition complete
#
done


#
# Create failover policies
# Information to create the failover policies is obtained from
# template script:
#       /usr/lib/failsafe/cmgr-templates/cmgr-create-cluster
#

#
# Create the first failover policy.
#

#
# Name of the failover policy.
#
define failover_policy sleepy-primary
#
# Failover policy attribute. This field is mandatory.
# Possible values: Auto_Failback, Controlled_Failback, Auto_Recovery,
# InPlace_Recovery
#

        set attribute to Auto_Failback

        set attribute to Auto_Recovery

#
# Failover policy script. The failover policy scripts have to
# be present in
# /usr/lib/failsafe/policies directory. This field is mandatory.
# Example value: ordered (file name not the full path name).
        set script to ordered
#
# Failover policy domain. Ordered list of nodes in the cluster
# separated by spaces. This field is mandatory.
#
        set domain to sleepy grumpy
#
# Failover policy definition complete
#
done
```

```
#
# Create the second failover policy.
#

#
# Name of the failover policy.
#
define failover_policy grumpy-primary
#
# Failover policy attribute. This field is mandatory.
# Possible values: Auto_Failback, Controlled_Failback, Auto_Recovery,
# InPlace_Recovery
#

        set attribute to Auto_Failback

        set attribute to InPlace_Recovery

#
# Failover policy script. The failover policy scripts have
# to be present in
# /usr/lib/failsafe/policies directory. This field is mandatory.
# Example value: ordered (file name not the full path name).
        set script to ordered
#
# Failover policy domain. Ordered list of nodes in the cluster
# separated by spaces. This field is mandatory.
#
        set domain to  grumpy sleepy
#
# Failover policy definition complete
#
done


#
# Create the IP_address resources.
# Information to create an IP_address resource is obtained from:
#       /usr/lib/failsafe/cmgr-templates/cmgr-create-resource-IP_address
#

#
# If multiple resources of resource type IP_address have to be created,
# repeat the following IP_address definition template.
#
# Name of the IP_address resource.  The name of the resource has to
# be IP address in the internet ‘‘.’’ notation. This IP address is used
# by clients to access highly available resources.
# Example value: 192.26.50.140
#
define resource 150.166.41.179 of resource_type IP_address in cluster failsafe-cluster
```

```
#
# The network mask for the IP address. The network mask value is used
# to configure the IP address on the network interface.
# Example value: 0xffffff00
        set NetworkMask to 0xffffff00
#
# The ordered list of interfaces that can be used to configure the IP
# address.The list of interface names are separated by comma.
# Example value: eth0, eth1
        set interfaces to eth1
#
# The broadcast address for the IP address.
# Example value: 192.26.50.255
        set BroadcastAddress to 150.166.41.255

#
# IP_address resource definition for the cluster complete
#
done

#
# Name of the IP_address resource.  The name of the resource has to be
# IP address in the internet ‘‘.’’ notation. This IP address is used by
# clients to access highly available resources.
# Example value: 192.26.50.140
#
define resource 150.166.41.99 of resource_type IP_address in cluster failsafe-cluster

#
# The network mask for the IP address. The network mask value is used
# to configure the IP address on the network interface.
# Example value: 0xffffff00
        set NetworkMask to 0xffffff00
#
# The ordered list of interfaces that can be used to configure the IP
# address.
# The list of interface names are separated by comma.
# Example value: eth0, eth1
        set interfaces to eth1
#
# The broadcast address for the IP address.
# Example value: 192.26.50.255
        set BroadcastAddress to 150.166.41.255

#
# IP_address resource definition for the cluster complete
#
done


#
# Create the volume resources.
# Information to create a volume resource is obtained from:
```

```
#       /usr/lib/failsafe/cmgr-templates/cmgr-create-resource-volume
#

#
# If multiple resources of resource type volume have to be created,
# repeat the following volume definition template.
#
# Name of the volume.  The name of the volume has to be:
# Example value: HA_vol (not /dev/xlv/HA_vol)
#
define resource bagheera of resource_type volume in cluster failsafe-cluster

#
# The user name of the device file name. This field is optional. If
# this field is not specified, value ‘‘root’’ is used.
# Example value: oracle
        set devname-owner to root
#
# The group name of the device file name. This field is optional.
# If this field is not specified, value ‘‘sys’’ is used.
# Example value: oracle
        set devname-group to sys
#
# The device file permissions. This field is optional. If this
# field is not specified, value ‘‘666’’ is used. The file permissions
# have to be specified in octal notation. See chmod(1) for more
# information.
# Example value: 666
        set devname-mode to 666

#
# Volume resource definition for the cluster complete
#
done

#
# Name of the volume.  The name of the volume has to be:
# Example value: HA_vol (not /dev/xlv/HA_vol)
#
define resource bhaloo of resource_type volume in cluster failsafe-cluster

#
# The user name of the device file name. This field is optional. If this
# field is not specified, value ‘‘root’’ is used.
# Example value: oracle
        set devname-owner to root
#
# The group name of the device file name. This field is optional.
# If this field is not specified, value ‘‘sys’’ is used.
# Example value: oracle
        set devname-group to sys
#
# The device file permissions. This field is optional. If this field is
```

```
# not specified, value ‘‘666’’ is used. The file permissions
# have to be specified in octal notation. See chmod(1) for more
# information.
# Example value: 666
        set devname-mode to 666

#
# Volume resource definition for the cluster complete
#
done


#
# Create the filesystem resources.
# Information to create a filesystem resource is obtained from:
#       /usr/lib/failsafe/cmgr-templates/cmgr-create-resource-filesystem
#

#
# filesystem resource type is for XFS filesystem only.

# If multiple resources of resource type filesystem have to be created,
# repeat the following filesystem definition template.
#
# Name of the filesystem.  The name of the filesystem resource has
# to be absolute path name of the filesystem mount point.
# Example value: /shared_vol
#
define resource /haathi of resource_type filesystem in cluster failsafe-cluster

#
# The name of the volume resource corresponding to the filesystem. This
# resource should be the same as the volume dependency, see below.
# This field is mandatory.
# Example value: HA_vol
        set volume-name to bagheera
#
# The options to be used when mounting the filesystem. This field is
# mandatory. For the list of mount options, see fstab(4).
# Example value: ‘‘rw’’
        set mount-options to rw
#
# The monitoring level for the filesystem. This field is optional. If
# this field is not specified, value ‘‘1’’ is used.
# Monitoring level can be
# 1 - Checks if filesystem exists in the mtab file (see mtab(4)). This
# is a lightweight check compared to monitoring level 2.
# 2 - Checks if the filesystem is mounted using stat(1m) command.
#
        set monitoring-level to 2
done

#
```

```
# Add filesystem resource type dependency
#
modify resource /haathi of resource_type filesystem in cluster failsafe-cluster
#
# The filesystem resource type definition also contains a resource
# dependency on a volume resource.
# This field is mandatory.
# Example value: HA_vol
        add dependency bagheera of type volume
#
# filesystem resource definition for the cluster complete
#
done

#
# Name of the filesystem.  The name of the filesystem resource has
# to be absolute path name of the filesystem mount point.
# Example value: /shared_vol
#
define resource /sherkhan of resource_type filesystem in cluster failsafe-cluster

#
# The name of the volume resource corresponding to the filesystem. This
# resource should be the same as the volume dependency, see below.
# This field is mandatory.
# Example value: HA_vol
        set volume-name to bhaloo
#
# The options to be used when mounting the filesystem. This field is
# mandatory.For the list of mount options, see fstab(4).
# Example value: ‘‘rw’’
        set mount-options to rw
#
# The monitoring level for the filesystem. This field is optional. If
# this field is not specified, value ‘‘1’’ is used.
# Monitoring level can be
# 1 - Checks if filesystem exists in the mtab file (see mtab(4)). This
# is a lightweight check compared to monitoring level 2.
# 2 - Checks if the filesystem is mounted using stat(1m) command.
#
        set monitoring-level to 2
done

#
# Add filesystem resource type dependency
#
modify resource /sherkhan of resource_type filesystem in cluster failsafe-cluster
#
# The filesystem resource type definition also contains a resource
# dependency on a volume resource.
# This field is mandatory.
# Example value: HA_vol
        add dependency bhaloo of type volume
```

```
#
# filesystem resource definition for the cluster complete
#
done


#
# Create the statd resource.
# Information to create a filesystem resource is obtained from:
#       /usr/lib/failsafe/cmgr-templates/cmgr-create-resource-statd
#

#
# If multiple resources of resource type statd have to be created,
# repeat the following filesystem definition template.
#
# Name of the statd.  The name of the resource has to be the location
# of the NFS/lockd directory.
# Example value: /disk1/statmon
#

define resource /haathi/statmon of resource_type statd in cluster failsafe-cluster

#
# The IP address on which the NFS clinets connect, this resource should
# be the same as the IP_address dependency, see below.
# This field is mandatory.
# Example value: 128.1.2.3
        set InterfaceAddress to 150.166.41.99
done


#
# Add the statd resource type dependencies
#
modify resource /haathi/statmon of resource_type statd in cluster failsafe-cluster
#
# The statd resource type definition also contains a resource
# dependency on a IP_address resource.
# This field is mandatory.
# Example value: 128.1.2.3
        add dependency 150.166.41.99 of type IP_address
#
# The statd resource type definition also contains a resource
# dependency on a filesystem resource. It defines the location of
# the NFS lock directory filesystem.
# This field is mandatory.
# Example value: /disk1
        add dependency /haathi of type filesystem
#
# statd resource definition for the cluster complete
#
done
```

```
#
# Create the NFS resources.
# Information to create a NFS resource is obtained from:
#       /usr/lib/failsafe/cmgr-templates/cmgr-create-resource-NFS
#

#
# If multiple resources of resource type NFS have to be created, repeat
# the following NFS definition template.
#
# Name of the NFS export point. The name of the NFS resource has to be
# export path name of the filesystem mount point.
# Example value: /disk1
#
define resource /haathi of resource_type NFS in cluster failsafe-cluster

#
# The export options to be used when exporting the filesystem. For the
# list of export options, see exportfs(1M).
# This field is mandatory.
# Example value: ‘‘rw,wsync,anon=root’’
        set export-info to rw
#
# The name of the filesystem resource corresponding to the export
# point. This resource should be the same as the filesystem dependency,
# see below.
# This field is mandatory.
# Example value: /disk1
        set filesystem to /haathi
done

#
# Add the resource type dependency
#
modify resource /haathi of resource_type NFS in cluster failsafe-cluster
#
# The NFS resource type definition also contains a resource dependency
# on a filesystem resource.
# This field is mandatory.
# Example value: /disk1
        add dependency /haathi of type filesystem
#
# The NFS resource type also contains a pseudo resource dependency
# on a statd resource. You really must have a statd resource associated
# with a NFS resource, so the NFS locks can be failed over.
# This field is mandatory.
# Example value: /disk1/statmon
        add dependency /haathi/statmon of type statd

#
# NFS resource definition for the cluster complete
```

```
#
done

#
# Name of the NFS export point. The name of the NFS resource has to be
# export path name of the filesystem mount point.
# Example value: /disk1
#
define resource /sherkhan of resource_type NFS in cluster failsafe-cluster

#
# The export options to be used when exporting the filesystem. For the
# list of export options, see exportfs(1M).
# This field is mandatory.
# Example value: ‘‘rw,wsync,anon=root’’
       set export-info to rw
#
# The name of the filesystem resource corresponding to the export
# point. This
# resource should be the same as the filesystem dependency, see below.
# This field is mandatory.
# Example value: /disk1
       set filesystem to /sherkhan
done

#
# Add the resource type dependency
#
modify resource /sherkhan of resource_type NFS in cluster failsafe-cluster
#
# The NFS resource type definition also contains a resource dependency
# on a filesystem resource.
# This field is mandatory.
# Example value: /disk1
       add dependency /sherkhan of type filesystem
#
# The NFS resource type also contains a pseudo resource dependency
# on a statd resource. You really must have a statd resource associated
# with a NFS resource, so the NFS locks can be failed over.
# This field is mandatory.
# Example value: /disk1/statmon
       add dependency /haathi/statmon of type statd

#
# NFS resource definition for the cluster complete
#
done


#
# Create the Netscape_web resource.
# Information to create a Netscape_web resource is obtained from:
#       /usr/lib/failsafe/cmgr-templates/cmgr-create-resource-Netscape_web
```

```
#

#
# If multiple resources of resource type Netscape_web have to be
# created, repeat the following filesystem definition template.
#
# Name of the Netscape WEB server.  The name of the resource has to be
# a unique identifier.
# Example value: ha80
#
define resource web-server of resource_type Netscape_web in cluster failsafe-cluster

#
# The locations of the servers startup and stop scripts.
# This field is mandatory.
# Example value: /usr/ns-home/ha86
        set admin-scripts to /var/netscape/suitespot/https-control3
#
# the TCP port number with the server listens on.
# This field is mandatory.
# Example value: 80
        set port-number to 80
#
# The desired monitoring level, the user can specify either;
#       1 - checks for process existence
#       2 - issues an HTML query to the server.
# This field is mandatory.
# Example value: 2
        set monitor-level to 2
#
# The locations of the WEB servers initial HTML page
# This field is mandatory.
# Example value: /var/www/htdocs
        set default-page-location to /var/www/htdocs
#
# The WEB servers IP address, this must be a configured IP_address
# resource.
# This resource should be the same as the IP_address dependency, see
# below.
# This field is mandatory.
# Example value: 28.12.9.5
        set web-ipaddr to 150.166.41.179
done

#
# Add the resource dependency
#
modify resource web-server of resource_type Netscape_web in cluster failsafe-cluster
#
# The Netscape_web resource type definition also contains a resource
# dependency on a IP_address resource.
# This field is mandatory.
# Example value: 28.12.9.5
```

```
        add dependency 150.166.41.179 of type IP_address
#
# Netscape_web resource definition for the cluster complete
#
done


#
# Create the resource groups.
# Information to create a resource group is obtained from:
#       /usr/lib/failsafe/cmgr-templates/cmgr-create-resource_group
#

#
# Name of the resource group. Name of the resource group must be unique
# in the cluster.
#
define resource_group nfs-group in cluster failsafe-cluster
#
# Failover policy for the resource group. This field is mandatory.
# Failover policy should be already defined in the CDB.
#
        set failover_policy to sleepy-primary
#
# List of resources in the resource group.
# Repeat the following line for each resource to be added to the
# resource group.
        add resource 150.166.41.99 of resource_type IP_address
#
# Add more resources to the resource group here.
#
        add resource bagheera of resource_type volume

        add resource bhaloo of resource_type volume

        add resource /haathi of resource_type filesystem

        add resource /sherkhan of resource_type filesystem

        add resource /haathi/statmon of resource_type statd

        add resource /haathi of resource_type NFS

        add resource /sherkhan of resource_type NFS

#
# Resource group definition complete
#
done

#
# Name of the resource group. Name of the resource group must be unique
# in the cluster.
```

```
#
define resource_group web-group in cluster failsafe-cluster
#
# Failover policy for the resource group. This field is mandatory.
# Failover policy should be already defined in the CDB.
#
        set failover_policy to grumpy-primary
#
# List of resources in the resource group.
# Repeat the following line for each resource to be added to the
# resource group.
        add resource 150.166.41.179 of resource_type IP_address

#
# Add more resources to the resource group here.
#

  add resource web-server of resource_type Netscape_web

#
# Resource group definition complete
#
done


#
# Script complete. This should be last line of the script
#
quit
```

6 Configuration Examples

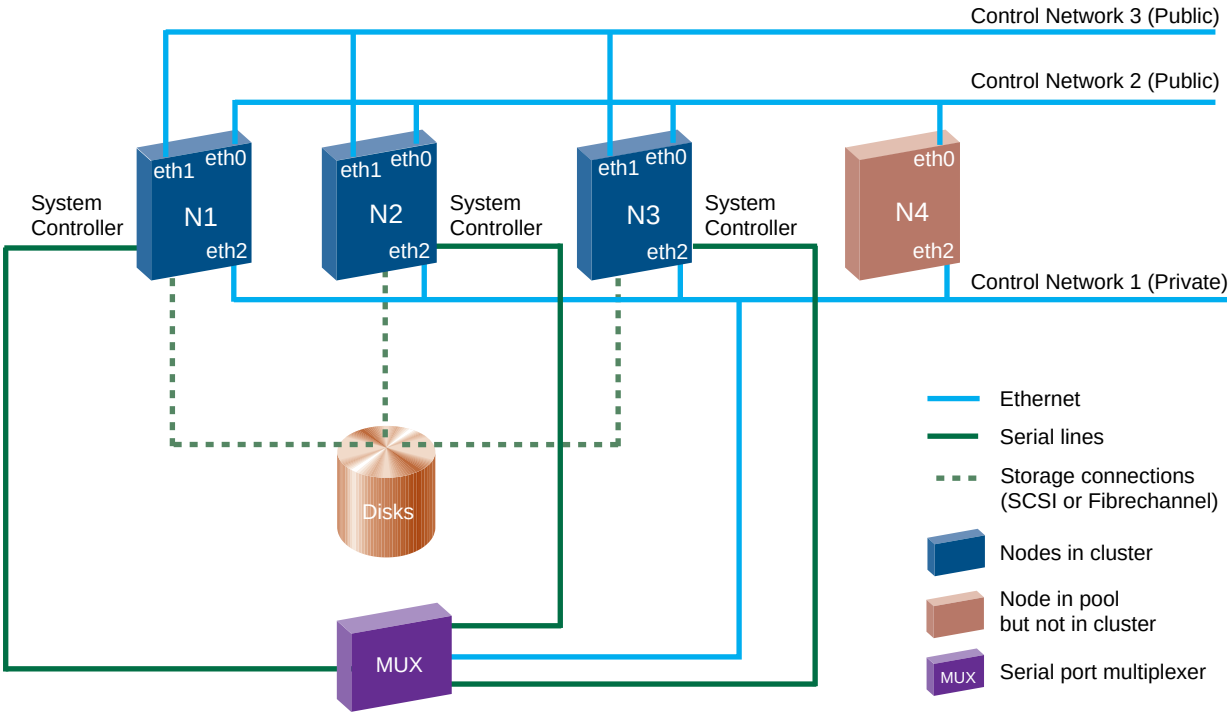
This chapter provides an example of a Linux FailSafe configuration that uses a three-node cluster, and some variations of that configuration. It includes the following sections:

- Section 6.1, *Linux FailSafe Example with Three-Node Cluster*
- Section 6.2, *cmgr Script*
- Section 6.3, *Local Failover of an IP Address*

6.1 Linux FailSafe Example with Three-Node Cluster

The following illustration shows a three-node Linux FailSafe cluster. This configuration consists of a pool containing nodes `N1`, `N2`, `N3`, and `N4`. Nodes `N1`, `N2`, and `N3` make up the Linux FailSafe cluster. The nodes in this cluster share disks, and are connected to a serial port multiplexer, which is also connected to the private control network.

Figure 6–1 Configuration Example



6.2 cmgr Script

This section provides an example `cmgr` script that defines a Linux FailSafe three-node cluster as shown in Section 6.1, *Linux FailSafe Example with Three-Node Cluster*. For general information on CLI scripts, see Section 4.3.4, *CLI Command Scripts*. For information on the CLI template files that you can use to create your own configuration script, see Section 4.3.5, *CLI Template Scripts*.

This cluster has two resource groups, `RG1` and `RG2`.

Resource group RG1 contains the following resources:

IP_address

192.26.50.1

filesystem

/ha1

volume

ha1_vol

NFS

/ha1/export

Resource group RG1 has a failover policy of FP1. FP1 has the following components:

script

ordered

attributes

Auto_Failback

Auto_Recovery

failover domain

N1, N2, N3

Resource group RG2 contains the following resources:

IP_address

192.26.50.2

filesystem

/ha2

volume

ha2_vol

NFS

/ha2/export

Resource group RG2 has a failover policy of FP2. FP2 has the following components:

script

round-robin

attributes

Controlled_Failback

`Inplace_Recovery`

failover domain

`N2, N3`

The `cmgr` script to define this configuration is as follows:

```
#!/usr/cluster/bin/cluster_mgr -f
define node N1
        set hostname to N1
        set sysctrl_type to msc
        set sysctrl_status to enabled
        set sysctrl_password to none
        set sysctrl_owner to N4
        set sysctrl_device to /dev/ttydn001
        set sysctrl_owner_type to tty
        add nic ef2-N1
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 1
        done
        add nic eth0-N1
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 2
        done
        add nic eth1-N1
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 3
        done
done


define node N2
        set hostname to N2
        set sysctrl_type to msc
        set sysctrl_status to enabled
        set sysctrl_password to none
        set sysctrl_owner to N4
        set sysctrl_device to /dev/ttydn002
        set sysctrl_owner_type to tty
        add nic ef2-N2
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 1
        done
        add nic eth0-N2
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 2
        done
        add nic eth1-N2
```

```
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 3
        done
done

define node N3
        set hostname to N3
        set sysctrl_type to msc
        set sysctrl_status to enabled
        set sysctrl_password to none
        set sysctrl_owner to N4
        set sysctrl_device to /dev/ttydn003
        set sysctrl_owner_type to tty
        add nic ef2-N3
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 1
        done
        add nic eth0-N3
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 2
        done
        add nic eth1-N3
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 3
        done
done


define node N4
        set hostname to N4
         add nic ef2-N4
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 1
        done
        add nic eth0-N4
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 2
        done
done
define cluster TEST
        set notify_cmd to /usr/bin/mail
        set notify_addr to failsafe_sysadm@company.com
        add node N1
        add node N2
        add node N3
done
```

```
define failover_policy fp1
        set attribute to Auto_Failback
        set attribute to Auto_Recovery
        set script to ordered
        set domain to N1 N2 N3
done


define failover_policy fp2
        set attribute to Controlled_Failback
        set attribute to Inplace_Recovery
        set script to round-robin
        set domain to N2 N3
done

define resource 192.26.50.1 of resource_type IP_address in cluster TEST
        set NetworkMask to 0xffffff00
        set interfaces to eth0,eth1
        set BroadcastAddress to 192.26.50.255
done

define resource ha1_vol of resource_type volume in cluster TEST
        set devname-owner to root
        set devname-group to sys
        set devname-mode to 666
done

define resource /ha1 of resource_type filesystem in cluster TEST
        set volume-name to ha1_vol
        set mount-options to rw,noauto
        set monitoring-level to 2
done

modify resource /ha1 of resource_type filesystem in cluster TEST
        add dependency ha1_vol of type volume
done

define resource /ha1/export of resource_type NFS in cluster TEST
        set export-info to rw,wsync
        set filesystem to /ha1
done

modify resource /ha1/export of resource_type NFS in cluster TEST
        add dependency /ha1 of type filesystem
done
define resource_group RG1 in cluster TEST
        set failover_policy to fp1
        add resource 192.26.50.1 of resource_type IP_address
        add resource ha1_vol of resource_type volume
        add resource /ha1 of resource_type filesystem
        add resource /ha1/export of resource_type NFS
done
```

```
define resource 192.26.50.2 of resource_type IP_address in cluster TEST
        set NetworkMask to 0xffffff00
        set interfaces to eth0
        set BroadcastAddress to 192.26.50.255
done

define resource ha2_vol of resource_type volume in cluster TEST
        set devname-owner to root
        set devname-group to sys
        set devname-mode to 666
done

define resource /ha2 of resource_type filesystem in cluster TEST
        set volume-name to ha2_vol
        set mount-options to rw,noauto
        set monitoring-level to 2
done

modify resource /ha2 of resource_type filesystem in cluster TEST
        add dependency ha2_vol of type volume
done

define resource /ha2/export of resource_type NFS in cluster TEST
        set export-info to rw,wsync
        set filesystem to /ha2
done

modify resource /ha2/export of resource_type NFS in cluster TEST
        add dependency /ha2 of type filesystem
done

define resource_group RG2 in cluster TEST
        set failover_policy to fp2
        add resource 192.26.50.2 of resource_type IP_address
        add resource ha2_vol of resource_type volume
        add resource /ha2 of resource_type filesystem
        add resource /ha2/export of resource_type NFS
done


quit
```

6.3 Local Failover of an IP Address

You can configure a Linux FailSafe system to fail over an IP address to a second interface within the same host. To do this, specify multiple interfaces for resources of `IP_address` resource type. You can also specify different interfaces for supporting a heterogeneous cluster. For information on specifying IP address resources, see Section 5.5.1.1, *IP Address Resource Attributes*.

The following example configures local failover of an IP address. It uses the configuration illustrated in Section 6.1, *Linux FailSafe Example with Three-Node Cluster*.

1. Define an IP address resource with two interfaces:

```
define resource 192.26.50.1 of resource_type IP_address in cluster TEST
          set NetworkMask to 0xffffff00
          set interfaces to eth0,eth1
          set BroadcastAddress to 192.26.50.255
done
```

   IP address 192.26.50.1 will be locally failed over from interface `eth0` to interface `eth1` when there is an `eth0` interface failure.

   In nodes `N1`, `N2`, and `N3`, either `eth0` or `eth1` should configure up automatically, when the node boots up. Both `eth0` and `eth1` are physically connected to the same subnet 192.26.50. Only one network interface connected to the same network should be configured up in a node.

2. Modify the `/etc/conf/netif.options` file to configure the `eth0` and `eth1` interfaces:

```
if1name-eth0 if1addr=192.26.50.10  if2name=eth1 if2addr=192.26.50.11
```

3. The `etc/init.d/network` script should configure the network interface `eth1` down in all nodes `N1`, `N2`, and `N3`. Add the following line to the file:

```
ifconfig eth1 down
```

7 Linux FailSafe System Operation

This chapter describes administrative tasks you perform to operate and monitor a Linux FailSafe system. It describes how to perform tasks using the FailSafe Cluster Manager Graphical User Interface (GUI) and the FailSafe Cluster Manager Command Line Interface (CLI). The major sections in this chapter are as follows:

- Section 7.1, *Setting System Operation Defaults*
- Section 7.2, *System Operation Considerations*
- Section 7.3, *Activating (Starting) Linux FailSafe*
- Section 7.4, *System Status*
- Section 7.5, *Resource Group Failover*
- Section 7.6, *Deactivating (Stopping) Linux FailSafe*
- Section 7.7, *Resetting Nodes*
- Section 7.8, *Backing Up and Restoring Configuration With Cluster Manager CLI*

7.1 Setting System Operation Defaults

Several commands that you perform on a running system allow you the option of specifying a node or cluster. You can specify a node or a cluster to use as the default if you do not specify the node or cluster explicitly.

7.1.1 Setting Default Cluster with Cluster Manager GUI

The Cluster Manager GUI prompts you to enter the name of the default cluster when you have not specified one. Alternately, you can set the default cluster by clicking the “Select Cluster...” button at the bottom of the FailSafe Manager window.

When using the Cluster Manager GUI, there is no need to set a default node.

7.1.2 Setting Defaults with Cluster Manager CLI

When you are using the Cluster Manager CLI, you can use the following commands to specify default values. Use either of the following commands to specify a default cluster:

```
cmgr> set cluster A
cmgr> set node A
```

7.2 System Operation Considerations

Once a Linux FailSafe command is started, it may partially complete even if you interrupt the command by typing [Ctrl-c]. If you halt the execution of a command this way, you may leave the cluster in an indeterminate state and you may need to use the various status commands to determine the actual state of the cluster and its components.

7.3 Activating (Starting) Linux FailSafe

After you have configured your Linux FailSafe system and run diagnostic tests on its components, you can activate the highly available services by starting Linux FailSafe. You can start Linux FailSafe on a systemwide basis, on all of the nodes in a cluster, or on a specified node only.

When you start HA services on a subset of the nodes, you should make sure that resource groups are not running in other nodes in the cluster. For example, if a cluster contains nodes N1, N2, and N3 and HA services are started on nodes N1 and N2 but not on node N3, you should make sure that resource groups are not running on node N3. Linux FailSafe will not perform exclusivity checks on nodes where HA services are not started.

When you start HA services, the following actions are performed:

1. All nodes in the cluster in the CDB are enabled
2. Linux FailSafe returns success to the user after modifying the CDB
3. The local CMOND gets notification from `cdbd`
4. The local CMOND starts all HA processes (CMSD, GCD, SRMD, FSD) and IFD.
5. CMOND sets `failsafe2` `chkconfig` flag to `on`.

7.3.1 Activating Linux FailSafe with the Cluster Manager GUI

To start Linux FailSafe services using the Cluster Manager GUI, perform the following steps:

1. On the left side of the display, click on the "Nodes & Cluster" category.
2. On the right side of the display click on the "Start FailSafe HA Services" task link to launch the task.
3. Enter the selected inputs.
4. Click on "OK" at the bottom of the screen to complete the task.

7.3.2 Activating Linux FailSafe with the Cluster Manager CLI

To activate Linux FailSafe in a cluster, use the following command:

```
cmgr> start ha_services [on node A] [for cluster B]
```

7.4 System Status

While the Linux FailSafe system is running, you can monitor the status of the Linux FailSafe components to determine the state of the component. Linux FailSafe allows you to view the system status in the following ways:

- You can keep continuous watch on the state of a cluster using the FailSafe Cluster View of the Cluster Manager GUI.
- You can query the status of an individual resource group, node, or cluster using either the Cluster Manager GUI or the Cluster Manager CLI.
- You can use the `haStatus` script provided with the Cluster Manager CLI to see the status of all clusters, nodes, resources, and resource groups in the configuration.

The following sections describe the procedures for performing each of these tasks.

7.4.1 Monitoring System Status with the Cluster Manager GUI

The easiest way to keep a continuous watch on the state of a cluster is to use the FailSafe Cluster View of the Cluster Manager GUI.

In the FailSafe Cluster View window, problems system components are experiencing appear as blinking red icons. Components in transitional states also appear as blinking icons. If there is a problem in a resource group or node, the FailSafe Cluster View icon for the cluster turns red and blinks, as well as the resource group or node icon.

The full color legend for component states in the FailSafe Cluster View is as follows:

grey

healthy but not online or active

green

healthy and active or online

blinking green

transitioning to green

blinking red

problems with component

black and white outline

resource type

grey with yellow wrench

maintenance mode, may or may not be currently monitored by Linux FailSafe

If you minimize the FailSafe Cluster View window, the minimized-icon shows the current state of the cluster. When the cluster has Linux FailSafe HA services active and there is no error, the icon shows a green cluster. When the cluster goes into error state, the icon shows a red cluster. When the cluster has Linux FailSafe HA services inactive, the icon shows a grey cluster.

7.4.2 Monitoring Resource and Reset Serial Line with the Cluster Manager CLI

You can use the CLI to query the status of a resource or to ping the system controller at a node, as described in the following subsections.

7.4.2.1 Querying Resource Status with the Cluster Manager CLI

To query a resource status, use the following CLI command:

```
cmgr> show status of resource A of resource_type B [in cluster C]
```

If you have specified a default cluster, you do not need to specify a cluster when you use this command and it will show the status of the indicated resource in the default cluster.

7.4.2.2 Pinging a System Controller with the Cluster Manager CLI

To perform a ping operation on a system controller by providing the device name, use the following CLI command:

```
cmgr> admin ping dev_name A of dev_type B with sysctrl_type C
```

7.4.3 Resource Group Status

To query the status of a resource group, you provide the name of the resource group and the cluster which includes the resource group. Resource group status includes the following components:

- Resource group state
- Resource group error state
- Resource owner

These components are described in the following subsections.

If a node that contains a resource group online has a status of UNKNOWN, the status of the resource group will not be available or ONLINE-READY.

7.4.3.1 Resource Group State

A resource group state can be one of the following:

ONLINE

Linux FailSafe is running on the local nodes. The resource group is allocated on a node in the cluster and is being monitored by Linux FailSafe. It is fully allocated if there is no error; otherwise, some resources may not be allocated or some resources may be in error state.

ONLINE-PENDING

Linux FailSafe is running on the local nodes and the resource group is in the process of being allocated. This is a transient state.

OFFLINE

The resource group is not running or the resource group has been detached, regardless of whether Linux FailSafe is running. When Linux FailSafe starts up, it will not allocate this resource group.

OFFLINE-PENDING

Linux FailSafe is running on the local nodes and the resource group is in the process of being released (becoming offline). This is a transient state.

ONLINE-READY

Linux FailSafe is not running on the local node. When Linux FailSafe starts up, it will attempt to bring this resource group online. No Linux FailSafe process is running on the current node is this state is returned.

ONLINE-MAINTENANCE

The resource group is allocated in a node in the cluster but it is not being monitored by Linux FailSafe. If a node failure occurs while a resource group in ONLINE-MAINTENANCE state resides on that node, the resource group will be moved to another node and monitoring will resume. An administrator may move a resource group to an ONLINE-MAINTENANCE state for upgrade or testing purposes, or if there is any reason that Linux FailSafe should not act on that resource for a period of time.

INTERNAL ERROR

An internal Linux FailSafe error has occurred and Linux FailSafe does not know the state of the resource group. Error recovery is required.

DISCOVERY (EXCLUSIVITY)

The resource group is in the process of going online if Linux FailSafe can correctly determine whether any resource in the resource group is already allocated on all nodes in the resource group's application failure domain. This is a transient state.

INITIALIZING

Linux FailSafe on the local node has yet to get any information about this resource group. This is a transient state.

7.4.3.2 Resource Group Error State

When a resource group is ONLINE, its error status is continually being monitored. A resource group error status can be one of the following:

NO ERROR

Resource group has no error.

INTERNAL ERROR - NOT RECOVERABLE

Notify Silicon Graphics if this condition arises.

NODE UNKNOWN

Node that had the resource group online is in unknown state. This occurs when the node is not part of the cluster. The last known state of the resource group is ONLINE, but the system cannot talk to the node.

SRMD EXECUTABLE ERROR

The start or stop action has failed for a resource in the resource group.

SPLIT RESOURCE GROUP (EXCLUSIVITY)

Linux FailSafe has determined that part of the resource group was running on at least two different nodes in the cluster.

NODE NOT AVAILABLE (EXCLUSIVITY)

Linux FailSafe has determined that one of the nodes in the resource group's application failure domain was not in the membership. Linux FailSafe cannot bring the resource group online until that node is removed from the application failure domain or HA services are started on that node.

MONITOR ACTIVITY UNKNOWN

In the process of turning maintenance mode on or off, an error occurred. Linux FailSafe can no longer determine if monitoring is enabled or disabled. Retry the operation. If the error continues, report the error to Silicon Graphics.

NO AVAILABLE NODES

A monitoring error has occurred on the last valid node in the cluster's membership.

7.4.3.3 Resource Owner

The resource owner is the logical node name of the node that currently owns the resource.

7.4.3.4 Monitoring Resource Group Status with the Cluster Manager GUI

You can use the FailSafe ClusterView to monitor the status of the resources in a Linux FailSafe configuration. You can launch the FailSafe Cluster View directly, or you can bring it up at any time by clicking on "FailSafe Cluster View" at the bottom of the "FailSafe Manager" display.

From the View menu, select "Resources in Groups" to see the resources organized by the groups they belong to, or select "Groups owned by Nodes" to see where the online groups are running. This view lets you observe failovers as they occur.

7.4.3.5 Querying Resource Group Status with the Cluster Manager CLI

To query a resource group status, use the following CLI command:

```
cmgr> show status of resource_group A [in cluster B]
```

If you have specified a default cluster, you do not need to specify a cluster when you use this command and it will show the status of the indicated resource group in the default cluster.

7.4.4 Node Status

To query the status of a node, you provide the logical node name of the node. The node status can be one of the following:

UP

This node is part of cluster membership.

DOWN

This node is not part of cluster membership (no heartbeats) and this node has been reset. This is a transient state.

UNKNOWN

This node is not part of cluster membership (no heartbeats) and this node has not been reset (reset attempt has failed).

INACTIVE

HA services have not been started on this node.

When you start HA services, node states transition from `INACTIVE` to `UP`. It may happen that a node state may transition from `INACTIVE` to `UNKNOWN` to `UP`.

7.4.4.1 Monitoring Cluster Status with the Cluster Manager GUI

You can use the FailSafe Cluster View to monitor the status of the clusters in a Linux FailSafe configuration. You can launch the FailSafe Cluster View directly, or you can bring it up at any time by clicking on “FailSafe Cluster View” at the bottom of the “FailSafe Manager” display.

From the View menu, select “Groups owned by Nodes” to monitor the health of the default cluster, its resource groups, and the group’s resources.

7.4.4.2 Querying Node Status with the Cluster Manager CLI

To query node status, use the following CLI command:

```
cmgr> show status of node A
```

7.4.4.3 Pinging the System Controller with the Cluster Manager CLI

When Linux FailSafe is running, you can determine whether the system controller on a node is responding with the following Cluster Manger CLI command:

```
cmgr> admin ping node A
```

This command uses the Linux FailSafe daemons to test whether the system controller is responding.

You can verify reset connectivity on a node in a cluster even when the Linux FailSafe daemons are not running by using the `standalone` option of the `admin ping` command of the CLI:

```
cmgr> admin ping standalone node A
```

This command does not go through the Linux FailSafe daemons, but calls the `ping` command directly to test whether the system controller on the indicated node is responding.

7.4.5 Cluster Status

To query the status of a cluster, you provide the name of the cluster. The cluster status can be one of the following:

- `ACTIVE`
- `INACTIVE`

7.4.5.1 Querying Cluster Status with the Cluster Manager GUI

You can use the Cluster View of the Cluster Manager GUI to monitor the status of the clusters in a Linux FailSafe system.

7.4.5.2 Querying Cluster Status with the Cluster Manager CLI

To query node and cluster status, use the following CLI command:

```
cmgr> show status of cluster A
```

7.4.6 Viewing System Status with the haStatus CLI Script

The `haStatus` script provides status and configuration information about clusters, nodes, resources, and resource groups in the configuration. This script is installed in the

/var/cluster/cmgr-scripts directory. You can modify this script to suit your needs. See the haStatus (1M) man page for further information about this script.

The following examples show the output of the different options of the haStatus script.

```
# haStatus -help
Usage: haStatus [-a|-i] [-c clustername]
where,
 -a prints detailed cluster configuration information and cluster
status.
 -i prints detailed cluster configuration information only.
 -c can be used to specify a cluster for which status is to be printed.
 ‘‘clustername’’ is the name of the cluster for which status is to be
printed.
# haStatus
Tue Nov 30 14:12:09 PST 1999
Cluster test-cluster:
        Cluster state is ACTIVE.
Node hans2:
        State of machine is UP.
Node hans1:
        State of machine is UP.
Resource_group nfs-group1:
        State: Online
        Error: No error
        Owner: hans1
        Failover Policy: fp_h1_h2_ord_auto_auto
        Resources:
                /hafs1  (type: NFS)
                /hafs1/nfs/statmon      (type: statd)
                150.166.41.95   (type: IP_address)
                /hafs1  (type: filesystem)
                havol1  (type: volume)
# haStatus -i
Tue Nov 30 14:13:52 PST 1999
Cluster test-cluster:
Node hans2:
        Logical Machine Name: hans2
        Hostname: hans2.engr.sgi.com
        Is FailSafe: true
        Is Cellular: false
        Nodeid: 32418
        Reset type: powerCycle
        System Controller: msc
        System Controller status: enabled
        System Controller owner: hans1
        System Controller owner device: /dev/ttyd2
        System Controller owner type: tty
        ControlNet Ipaddr: 192.26.50.15
        ControlNet HB: true
        ControlNet Control: true
        ControlNet Priority: 1
        ControlNet Ipaddr: 150.166.41.61
        ControlNet HB: true
```

```
        ControlNet Control: false
        ControlNet Priority: 2
Node hans1:
        Logical Machine Name: hans1
        Hostname: hans1.engr.sgi.com
        Is FailSafe: true
        Is Cellular: false
        Nodeid: 32645
        Reset type: powerCycle
        System Controller: msc
        System Controller status: enabled
        System Controller owner: hans2
        System Controller owner device: /dev/ttyd2
        System Controller owner type: tty
        ControlNet Ipaddr: 192.26.50.14
        ControlNet HB: true
        ControlNet Control: true
        ControlNet Priority: 1
        ControlNet Ipaddr: 150.166.41.60
        ControlNet HB: true
        ControlNet Control: false
        ControlNet Priority: 2
Resource_group nfs-group1:
        Failover Policy: fp_h1_h2_ord_auto_auto
                Version: 1
                Script: ordered
                Attributes: Auto_Failback Auto_Recovery
                Initial AFD: hans1 hans2
        Resources:
                /hafs1  (type: NFS)
                /hafs1/nfs/statmon      (type: statd)
                150.166.41.95   (type: IP_address)
                /hafs1  (type: filesystem)
                havol1  (type: volume)
Resource /hafs1 (type NFS):
        export-info: rw,wsync
        filesystem: /hafs1
        Resource dependencies
        statd /hafs1/nfs/statmon
        filesystem /hafs1
Resource /hafs1/nfs/statmon (type statd):
        InterfaceAddress: 150.166.41.95
        Resource dependencies
        IP_address 150.166.41.95
        filesystem /hafs1
Resource 150.166.41.95 (type IP_address):
        NetworkMask: 0xffffff00
        interfaces: ef1
        BroadcastAddress: 150.166.41.255
        No resource dependencies
Resource /hafs1 (type filesystem):
        volume-name: havol1
        mount-options: rw,noauto
```

```
        monitoring-level: 2
        Resource dependencies
        volume havol1
Resource havol1 (type volume):
        devname-group: sys
        devname-owner: root
        devname-mode: 666
        No resource dependencies
Failover_policy fp_h1_h2_ord_auto_auto:
        Version: 1
        Script: ordered
        Attributes: Auto_Failback Auto_Recovery
        Initial AFD: hans1 hans2
# haStatus -a
Tue Nov 30 14:45:30 PST 1999
Cluster test-cluster:
        Cluster state is ACTIVE.
Node hans2:
        State of machine is UP.
        Logical Machine Name: hans2
        Hostname: hans2.engr.sgi.com
        Is FailSafe: true
        Is Cellular: false
        Nodeid: 32418
        Reset type: powerCycle
        System Controller: msc
        System Controller status: enabled
        System Controller owner: hans1
        System Controller owner device: /dev/ttyd2
        System Controller owner type: tty
        ControlNet Ipaddr: 192.26.50.15
        ControlNet HB: true
        ControlNet Control: true
        ControlNet Priority: 1
        ControlNet Ipaddr: 150.166.41.61
        ControlNet HB: true
        ControlNet Control: false
        ControlNet Priority: 2
Node hans1:
        State of machine is UP.
        Logical Machine Name: hans1
        Hostname: hans1.engr.sgi.com
        Is FailSafe: true
        Is Cellular: false
        Nodeid: 32645
        Reset type: powerCycle
        System Controller: msc
        System Controller status: enabled
        System Controller owner: hans2
        System Controller owner device: /dev/ttyd2
        System Controller owner type: tty
        ControlNet Ipaddr: 192.26.50.14
        ControlNet HB: true
```

```
        ControlNet Control: true
        ControlNet Priority: 1
        ControlNet Ipaddr: 150.166.41.60
        ControlNet HB: true
        ControlNet Control: false
        ControlNet Priority: 2
Resource_group nfs-group1:
        State: Online
        Error: No error
        Owner: hans1
        Failover Policy: fp_h1_h2_ord_auto_auto
                Version: 1
                Script: ordered
                Attributes: Auto_Failback Auto_Recovery
                Initial AFD: hans1 hans2
        Resources:
                /hafs1  (type: NFS)
                /hafs1/nfs/statmon      (type: statd)
                150.166.41.95   (type: IP_address)
                /hafs1  (type: filesystem)
                havol1  (type: volume)
Resource /hafs1 (type NFS):
        State: Online
        Error: None
        Owner: hans1
        Flags: Resource is monitored locally
        export-info: rw,wsync
        filesystem: /hafs1
        Resource dependencies
        statd /hafs1/nfs/statmon
        filesystem /hafs1
Resource /hafs1/nfs/statmon (type statd):
        State: Online
        Error: None
        Owner: hans1
        Flags: Resource is monitored locally
        InterfaceAddress: 150.166.41.95
        Resource dependencies
        IP_address 150.166.41.95
        filesystem /hafs1
Resource 150.166.41.95 (type IP_address):
        State: Online
        Error: None
        Owner: hans1
        Flags: Resource is monitored locally
        NetworkMask: 0xffffff00
        interfaces: ef1
        BroadcastAddress: 150.166.41.255
        No resource dependencies
Resource /hafs1 (type filesystem):
        State: Online
        Error: None
        Owner: hans1
```

```
        Flags: Resource is monitored locally
        volume-name: havol1
        mount-options: rw,noauto
        monitoring-level: 2
        Resource dependencies
        volume havol1
Resource havol1 (type volume):
        State: Online
        Error: None
        Owner: hans1
        Flags: Resource is monitored locally
        devname-group: sys
        devname-owner: root
        devname-mode: 666
        No resource dependencies
# haStatus -c test-cluster
Tue Nov 30 14:42:04 PST 1999
Cluster test-cluster:
        Cluster state is ACTIVE.
Node hans2:
        State of machine is UP.
Node hans1:
        State of machine is UP.
Resource_group nfs-group1:
        State: Online
        Error: No error
        Owner: hans1
        Failover Policy: fp_h1_h2_ord_auto_auto
        Resources:
                /hafs1  (type: NFS)
                /hafs1/nfs/statmon      (type: statd)
                150.166.41.95   (type: IP_address)
                /hafs1  (type: filesystem)
                havol1  (type: volume)
```

7.5 Resource Group Failover

While a Linux FailSafe system is running, you can move a resource group online to a particular node, or you can take a resource group offline. In addition, you can move a resource group from one node in a cluster to another node in a cluster. The following subsections describe these tasks.

7.5.1 Bringing a Resource Group Online

Before you bring a resource group online for the first time, you should run the diagnostic tests on that resource group. Diagnostics check system configurations and perform some validations that are not performed when you bring a resource group online.

To bring a resource group online, you specify the name of the resource and the name of the cluster which contains the node.

You cannot bring a resource group online if the resource group has no members.

To bring a resource group fully online, HA services must be active. When HA services are active, an attempt is made to allocate the resource group in the cluster. However, you can also execute a command to bring the resource group online when HA services are not active. When HA services are not active, the resource group is marked to be brought online when HA services become active.

Before bringing a resource group online in the cluster, you must be sure that the resource group is not running on a disabled node (where HA services are not running). Bringing a resource group online while it is running on a disabled node could cause data corruption. For information on detached resource groups, see Section 7.5.2, *Taking a Resource Group Offline*.

7.5.1.1 Bringing a Resource Group Online with the Cluster Manager GUI

To bring a resource group online using the Cluster Manager GUI, perform the following steps:

1. On the left side of the display, click on the "Failover Policies & Resource Groups" category.
2. On the right side of the display click on the "Bring a Resource Group Online" task link to launch the task.
3. Enter the selected inputs.
4. Click on "OK" at the bottom of the screen to complete the task.

7.5.1.2 Bringing a Resource Group Online with the Cluster Manager CLI

To bring a resource group online, use the following CLI command:

```
cmgr> admin online resource_group A [in cluster B]
```

If you have specified a default cluster, you do not need to specify a cluster when you use this command.

7.5.2 Taking a Resource Group Offline

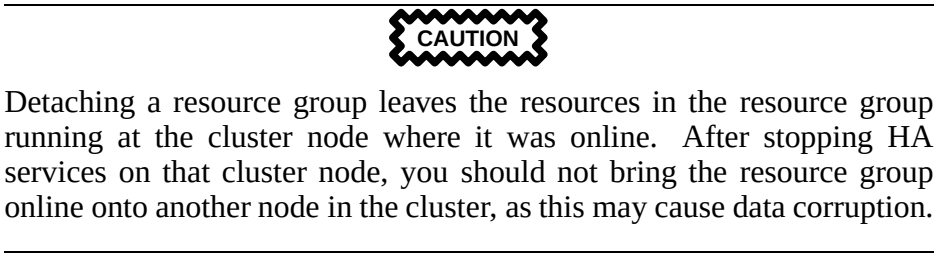
When you take a resource group offline, FaillSafe takes each resource in the resource group offline in a predefined order. If any single resource gives an error during this process, the process stops, leaving all remaining resources allocated.

You can take a Linux FailSafe resource group offline in any of three ways:

- Take the resource group offline. This physically stops the processes for that resource group and does not reset any error conditions. If this operation fails, the resource group will be left online in an error state.
- Force the resource group offline. This physically stops the processes for that resource group but resets any error conditions. This operation cannot fail.
- Detach the resource groups. This causes Linux FailSafe to stop monitoring the resource group, but does not physically stop the processes on that group. Linux FailSafe will report

the status as offline and will not have any control over the group. This operation should rarely fail.

If you do not need to stop the resource group and do not want Linux FailSafe to monitor the resource group while you make changes but you would still like to have administrative control over the resource group (for instance, to move that resource group to another node), you can put the resource group in maintenance mode using the "Suspend Monitoring a Resource Group" task on the GUI or the `admin maintenance_on` command of the CLI, as described in Section 7.5.4, *Stop Monitoring of a Resource Group (Maintenance Mode)*.



7.5.2.1 Taking a Resource Group Offline with the Cluster Manager GUI

To take a resource group offline using the Cluster Manager GUI, perform the following steps:

1. Launch the Cluster Manager.
2. On the left side of the display, click on the "Failover Policies & Resource Groups" category.
3. On the right side of the display click on the "Take a Resource Group Offline" task link to launch the task.
4. Enter the selected inputs.
5. Click on "OK" at the bottom of the screen to complete the task.

7.5.2.2 Taking a Resource Group Offline with the Cluster Manager CLI

To take a resource group offline, use the following CLI command:

```
cmgr> admin offline resource_group A [in cluster B]
```

If you have specified a default cluster, you do not need to specify a cluster in this command and the CLI will use the default.

To take a resource group offline with the force option in effect, use the following CLI command:

```
cmgr> admin offline_force resource_group A [in cluster B]
```

To detach a resource group, use the following CLI command:

```
cmgr> admin offline_detach resource_group A [in cluster B]
```

7.5.3 Moving a Resource Group

While Linux FailSafe is active, you can move a resource group to another node in the same cluster. When you move a resource group, you specify the following:

- The name of the resource group.

- The logical name of the destination node (optional). When you do not provide a logical destination name, Linux FailSafe chooses the destination based on the failover policy.
- The name of the cluster that contains the nodes.

7.5.3.1 Moving a Resource Group with the Cluster Manager GUI

To move a resource group using the Cluster Manager GUI, perform the following steps:

1. On the left side of the display, click on the “Failover Policies & Resource Groups” category.
2. On the right side of the display click on the “Move a Resource Group” task link to launch the task.
3. Enter the selected inputs.
4. Click on “OK” at the bottom of the screen to complete the task.

7.5.3.2 Moving a Resource Group with the Cluster Manager CLI

To move a resource group to another node, use the following CLI command:

```
cmgr> admin move resource_group A [in cluster B] [to node C]
```

7.5.4 Stop Monitoring of a Resource Group (Maintenance Mode)

You can temporarily stop Linux FailSafe from monitoring a specific resource group, which puts the resource group in maintenance mode. The resource group remains on its same node in the cluster but is no longer monitored by Linux FailSafe for resource failures.

You can put a resource group into maintenance mode if you do not want Linux FailSafe to monitor the group for a period of time. You may want to do this for upgrade or testing purposes, or if there is any reason that Linux FailSafe should not act on that resource group. When a resource group is in maintenance mode, it is not being monitored and it is not highly available. If the resource group’s owner node fails, Linux FailSafe will move the resource group to another node and resume monitoring.

When you put a resource group into maintenance mode, resources in the resource group are in `ONLINE-MAINTENANCE` state. The `ONLINE-MAINTENANCE` state for the resource is seen only on the node that has the resource online. All other nodes will show the resource as ONLINE. The resource group, however, should appear as being in `ONLINE-MAINTENANCE` state in all nodes.

7.5.4.1 Putting a Resource Group into Maintenance Mode with the Cluster Manager GUI

To put a resource group into maintenance mode using the Cluster Manager GUI, perform the following steps:

1. On the left side of the display, click on the “Failover Policies & Resource Groups” category.
2. On the right side of the display click on the “Suspend Monitoring a Resource Group” task link to launch the task.
3. Enter the selected inputs.

7.5.4.2 Resume Monitoring of a Resource Group with the Cluster Manager GUI

To resume monitoring a resource group using the Cluster Manager GUI, perform the following steps:

1. On the left side of the display, click on the “Failover Policies & Resource Groups” category.
2. On the right side of the display click on the “Resume Monitoring a Resource Group” task link to launch the task.
3. Enter the selected inputs.

7.5.4.3 Putting a Resource Group into Maintenance Mode with the Cluster Manager CLI

To put a resource group into maintenance mode, use the following CLI command:

```
cmgr> admin maintenance_on resource_group A [in cluster B]
```

If you have specified a default cluster, you do not need to specify a cluster when you use this command.

7.5.4.4 Resume Monitoring of a Resource Group with the Cluster Manager CLI

To move a resource group back online from maintenance mode, use the following CLI command:

```
cmgr> admin maintenance_off resource_group A [in cluster B]
```

7.6 Deactivating (Stopping) Linux FailSafe

You can stop the execution of Linux FailSafe on a systemwide basis, on all the nodes in a cluster, or on a specified node only.

Deactivating a node or a cluster is a complex operation that involves several steps and can take several minutes. Aborting a deactivate operation can leave the nodes and the resources in an intended state.

When deactivating HA services on a node or for a cluster, the operation may fail if any resource groups are not in a stable clean state. Resource groups which are in transition will cause any deactivate HA services command to fail. In many cases, the command may succeed at a later time after resource groups have settled into a stable state.

After you have successfully deactivated a node or a cluster, the node or cluster should have no resource groups and all HA services should be gone.

Serially stopping HA services on every node in a cluster is not the same as stopping HA services for the entire cluster. If the former case, an attempt is made to keep resource groups online and highly available while in the latter case resource groups are moved offline, as described in the following sections.

When you stop HA services, the Linux FailSafe daemons perform the following actions:

1. A shutdown request is sent to Linux FailSafe (FSD)
2. FSD releases all resource groups and puts them in `ONLINE-READY` state
3. All nodes in the cluster in the configuration database are disabled (one node at a time and the local node last)
4. Linux FailSafe waits until the node is removed from cluster membership before disabling the node
5. The shutdown is successful only when all nodes are not part of cluster membership

6. CMOND receives notification from the configuration database when nodes are disabled
7. The local CMOND sends SIGTERM to all HA processes and IFD.
8. All HA processes clean up and exit with “don’t restart” code
9. All other CMSD daemons remove the disabled node from the cluster membership

7.6.1 Deactivating HA Services on a Node

The operation of deactivating a node tries to move all resource groups from the node to some other node and then tries to disable the node in the cluster, subsequently killing all HA processes.

When HA services are stopped on a node, all resource groups owned by the node are moved to some other node in the cluster that is capable of maintaining these resource groups in a highly available state. This operation will fail if there is no node that can take over these resource groups. This condition will always occur if the last node in a cluster is shut down when you deactivate HA services on that node.

In this circumstance, you can specify the `force` option to shut down the node even if resource groups cannot be moved or released. This will normally leave resource groups allocated in a non-highly-available state on that same node. Using the `force` option might result in the node getting reset. In order to guarantee that the resource groups remain allocated on the last node in a cluster, all online resource groups should be detached.

If you wish to move resource groups offline that are owned by the node being shut down, you must do so prior to deactivating the node.

7.6.2 Deactivating HA Services in a Cluster

The operation of deactivating a cluster attempts to release all resource groups and disable all nodes in the cluster, subsequently killing all HA processes.

When a cluster is deactivated and the Linux FailSafe HA services are stopped on that cluster, resource groups are moved offline or deallocated. If you want the resource groups to remain allocated, you must detach the resource groups before attempting to deactivate the cluster.

Serially stopping HA services on every node in a cluster is not the same as stopping HA services for the entire cluster. If the former case, an attempt is made to keep resource groups online and highly available while in the latter case resource groups are moved offline.

7.6.3 Deactivating Linux FailSafe with the Cluster Manager GUI

To stop Linux FailSafe services using the Cluster Manager GUI, perform the following steps:

1. On the left side of the display, click on the “Nodes & Cluster” category.
2. On the right side of the display click on the “Stop FailSafe HA Services” task link to launch the task.
3. Enter the selected inputs.
4. Click on “OK” at the bottom of the screen to complete the task.

7.6.4 Deactivating Linux FailSafe with the Cluster Manager CLI

To deactivate Linux FailSafe in a cluster and stop Linux FailSafe processing, use the following command:

```
cmgr> stop ha_services [on node A] [for cluster B][force]
```

7.7 Resetting Nodes

You can use Linux FailSafe to reset nodes in a cluster. This sends a reset command to the system controller port on the specified node. When the node is reset, other nodes in the cluster will detect this and remove the node from the active cluster, reallocating any resource groups that were allocated on that node onto a backup node. The backup node used depends on how you have configured your system.

Once the node reboots, it will rejoin the cluster. Some resource groups might move back to the node, depending on how you have configured your system.

7.7.1 Resetting a Node with the Cluster Manager GUI

To reset a Linux FailSafe node using the Cluster Manager GUI, perform the following steps:

1. On the left side of the display, click on the “Nodes & Cluster” category.
2. On the right side of the display click on the “Reset a Node” task link to launch the task.
3. Enter the node to reset.
4. Click on “OK” at the bottom of the screen to complete the task.

7.7.2 Resetting a Node with the Cluster Manager CLI

When Linux FailSafe is running, you can reboot a node with the following Cluster Manger CLI command:

```
cmgr> admin reset node A
```

This command uses the Linux FailSafe daemons to reset the specified node.

You can reset a node in a cluster even when the Linux FailSafe daemons are not running by using the `standalone` option of the `admin reset` command of the CLI:

```
cmgr> admin reset standalone node A
```

This command does not go through the Linux FailSafe daemons.

7.8 Backing Up and Restoring Configuration With Cluster Manager CLI

The Cluster Manager CLI provides scripts that you can use to backup and restore your configuration: `cdbDump` and `cdbRestore`. These scripts are installed in the `/var/cluster/cmgr-scripts` directory. You can modify these scripts to suit your needs.

The `cdbDump` script, as provided, creates compressed tar files of the `/var/cluster/cdb/cdb.db#` directory and the `/var/cluster/cdb.db` file.

The `cdbRestore` script, as provided, restores the compressed tar files of the `/var/cluster/cdb/cdb.db#` directory and the `/var/cluster/cdb.db` file.

When you use the `cdbDump` and `cdbRestore` scripts, you should follow the following procedures:

- Run the `cdbDump` and `cdbRestore` scripts only when no administrative commands are running. This could result in an inconsistent backup.
- You must backup the configuration of each node in the cluster separately. The configuration information is different for each node, and all node-specific information is stored locally only.
- Run the backup procedure whenever you change your configuration.
- The backups of all nodes in the pool taken at the same time should be restored together.
- Cluster and Linux FailSafe process should not be running when you restore your configuration.

In addition to the above restrictions, you should not perform a `cdbDump` while information is changing in the CDB. Check SYSLOG for information to help determine when CDB activity is occurring. As a rule of thumb, you should be able to perform a `cdbDump` if at least 15 minutes have passed since the last node joined the cluster or the last administration command was run.

---

8 Testing Linux FailSafe Configuration

This chapter explains how to test the Linux FailSafe system configuration using the Cluster Manager GUI and the Cluster Manager CLI. For general information on using the Cluster Manager GUI and the Cluster Manager CLI, see Chapter 4, *Linux FailSafe Administration Tools*.

The sections in this chapter are as follows:

- Section 8.1, *Overview of FailSafe Diagnostic Commands*
- Section 8.2, *Performing Diagnostic Tasks with the Cluster Manager GUI*
- Section 8.3, *Performing Diagnostic Tasks with the Cluster Manager CLI*

8.1 Overview of FailSafe Diagnostic Commands

Table 8–1, *FailSafe Diagnostic Test Summary* shows the tests you can perform with Linux FailSafe diagnostic commands:

Table 8–1 FailSafe Diagnostic Test Summary

Diagnostic Test	Checks Performed
resource	Checks that the resource type parameters are set
	Check that the parameters are syntactically correct
	Validates that the parameters exist
resource group	Tests all resources defined in the resource group
failover policy	Checks that the failover policy exists
	Checks that the failover domain contains a valid list of hosts
network connectivity	Checks that the control interfaces are on the same network
	Checks that the nodes can communicate with each other
serial connection	Checks that the nodes can reset each other

All transactions are logged to the diagnostics file `diags_`*nodename* in the log directory.

You should test resource groups before starting FailSafe HA services or starting a resource group. These tests are designed to check for resource inconsistencies which could prevent the resource group from starting successfully.

8.2 Performing Diagnostic Tasks with the Cluster Manager GUI

To test the components of a FailSafe system using the Cluster Manager GUI, perform the following steps:

1. Select Task Manager on the FailSafe Toolchest.
2. On the left side of the display, click on the “Diagnostics” category.

3. Select one of the diagnostics tasks that appear on the right side of the display: “Test Connectivity,” “Test Resources,” or “Test Failover Policy.”

8.2.1 Testing Connectivity with the Cluster Manager GUI

When you select the “Test Connectivity” task from the Diagnostics display, you can test the network and serial connections on the nodes in your cluster by entering the requested inputs. You can test all of the nodes in the cluster at one time, or you can specify an individual node to test.

8.2.2 Testing Resources with the Cluster Manager GUI

When you select the “Test Resources” task from the Diagnostics display, you can test the resources on the nodes in your cluster by entering the requested inputs. You can test resources by type and by group. You can test the resources of a resource type or in a resource group on all of the nodes in the cluster at one time, or you can specify an individual node to test. Resource tests are performed only on nodes in the resource group’s application failover domain.

8.2.3 Testing Failover Policies with the Cluster Manager GUI

When you select the “Test Failover Policy” task from the Diagnostics display, you can test whether a failover policy is defined correctly. This test checks the failover policy by validating the policy script, failover attributes, and whether the application failover domain consists of valid nodes from the cluster.

8.3 Performing Diagnostic Tasks with the Cluster Manager CLI

The following subsections described how to perform diagnostic tasks on your system using the Cluster Manager CLI commands.

8.3.1 Testing the Serial Connections with the Cluster Manager CLI

You can use the Cluster Manager CLI to test the serial connections between the Linux FailSafe nodes. This test pings each specified node through the serial line and produces an error message if the ping is not successful. Do not execute this command while FailSafe is running.

When you are using the Cluster Manager CLI, use the following command to test the serial connections for the machines in a cluster

```
cmgr> test serial in cluster A [on node B node C ...]
```

This test yields an error message when it encounters its first error, indicating the node that did not respond. If you receive an error message after executing this test, verify the cable connections of the serial cable from the indicated node’s serial port to the remote power control unit or the system controller port of the other nodes and run the test again.

The following shows an example of the `test serial` CLI command:

```
# cluster_mgr
Welcome to Linux FailSafe Cluster Manager Command-Line Interface

cmgr> test serial in cluster eagan on node cm1
```

```
Success: testing serial...
Success: Ensuring Node Can Get IP Addresses For All Specified Hosts
Success: Number of IP addresses obtained for <cm1> = 1
Success:        The first IP address for <cm1> = 128.162.19.34
Success: Checking serial lines via crsd (crsd is running)
Success: Successfully checked serial line
Success: Serial Line OK
Success: overall exit status:success, tests failed:0, total tests executed:1
```

The following shows an example of an attempt to run the `test serial` CLI command while FailSafe is running (causing the command to fail to execute):

```
cmgr> test serial in cluster eagan on node cm1
Error: Cannot run the serial tests, diagnostics has detected FailSafe (ha_cmsd) is running

Failed to execute FailSafe tests/diagnostics ha

test command failed
cmgr>
```

8.3.2 Testing Network Connectivity with the Cluster Manager CLI

You can use the Cluster Manager CLI to test the network connectivity in a cluster. This test checks if the specified nodes can communicate with each other through each configured interface in the nodes. This test will not run if FailSafe is running.

When you are using the Cluster Manager CLI, use the following command to test the network connectivity for the machines in a cluster

```
cmgr> test connectivity in cluster A [on node B node C ...]
```

The following shows an example of the `test connectivity` CLI command:

```
cmgr> test connectivity in cluster eagan on node cm1
Success: testing connectivity...
Success: checking that the control IP_addresses are on the same networks
Success: pinging address cm1-priv interface ef0 from host cm1
Success: pinging address cm1 interface ef1 from host cm1
Success: overall exit status:success, tests failed:0, total tests
executed:1
```

This test yields an error message when it encounters its first error, indicating the node that did not respond. If you receive an error message after executing this test, verify that the network interface has been configured up, using the `ifconfig` command, for example:

/usr/etc/ifconfig ec3

```
ec3: flags=c63<UP,BROADCAST,NOTRAILERS,RUNNING,FILTMULTI,MULTICAST>
        inet 190.0.3.1 netmask 0xffffff00 broadcast 190.0.3.255
```

The UP in the first line of output indicates that the interface is configured up.

If the network interface is configured up, verify that the network cables are connected properly and run the test again.

8.3.3 Testing Resources with the Cluster Manager CLI

You can use the Cluster Manager CLI to test any configured resource by resource name or by resource type.

The Cluster Manager CLI uses the following syntax to test a resource by name:

```
cmgr> test resource A of resource_type B in cluster C [on node D node E ...]
```

The following shows an example of testing a resource by name:

```
cmgr> test resource /disk1 of resource_type filesystem in cluster eagan on machine cm1
Success: *** testing node resources on node cm1 ***
Success: *** testing all filesystem resources on node cm1 ***
Success: testing resource /disk1 of resource type filesystem on node cm1
Success: overall exit status:success, tests failed:0, total tests executed:1
```

The Cluster Manager CLI uses the following syntax to test a resource by resource type:

```
cmgr> test resource_type A in cluster B [on node C node D...]
```

The following shows an example of testing resources by resource type:

```
cmgr> test resource_type filesystem in cluster eagan on machine cm1
Success: *** testing node resources on node cm1 ***
Success: *** testing all filesystem resources on node cm1 ***
Success: testing resource /disk4 of resource type filesystem on node cm1
Success: testing resource /disk5 of resource type filesystem on node cm1
Success: testing resource /disk2 of resource type filesystem on node cm1
Success: testing resource /disk3 of resource type filesystem on node cm1
Success: testing resource /disk1 of resource type filesystem on node cm1
Success: overall exit status:success, tests failed:0, total tests executed:5
```

You can use the CLI to test volume and filesystem resources in *destructive* mode. This provides a more thorough test of filesystems and volumes. CLI tests will not run in destructive mode if FailSafe is running.

The Cluster Manager CLI uses the following syntax for the commands that test resources in destructive mode:

```
cmgr> test resource A of resource_type B in cluster C [on node D node C ...] destructive
```

The following sections describe the diagnostic tests available for resources.

8.3.3.1 Testing Logical Volumes

You can use the Cluster Manager CLI to test the logical volumes in a cluster. This test checks if the specified volume is configured correctly.

When you are using the Cluster Manager CLI, use the following command to test a logical volume:

```
cmgr> test resource A of resource_type volume on cluster B [on node C node D ...]
```

The following example tests a logical volume:

```
cmgr> test resource alternate of resource_type volume on cluster eagan
Success: *** testing node resources on node cm1 ***
```

```
Success: *** testing all volume resources on node cm1 ***
Success: running resource type volume tests on node cm1
Success: *** testing node resources on node cm2 ***
Success: *** testing all volume resources on node cm2 ***
Success: running resource type volume tests on node cm2
Success: overall exit status:success, tests failed:0, total tests executed:2
cmgr>
```

The following example tests a logical volume in destructive mode:

```
cmgr> test resource alternate of resource_type volume on cluster eagan destructive
Warning: executing the tests in destructive mode
Success: *** testing node resources on node cm1 ***
Success: *** testing all volume resources on node cm1 ***
Success: running resource type volume tests on node cm1
Success: successfully assembled volume: alternate
Success: *** testing node resources on node cm2 ***
Success: *** testing all volume resources on node cm2 ***
Success: running resource type volume tests on node cm2
Success: successfully assembled volume: alternate
Success: overall exit status:success, tests failed:0, total tests executed:2
cmgr>
```

8.3.3.2 Testing Filesystems

You can use the Cluster Manager CLI to test the filesystems configured in a cluster. This test checks if the specified filesystem is configured correctly and, in addition, checks whether the volume the filesystem will reside on is configured correctly.

When you are using the Cluster Manager CLI, use the following command to test a filesystem:

```
cmgr> test resource A of resource_type filesystems on cluster B [on node C node D ...]
```

The following example tests a filesystem. This example first uses a CLI `show` command to display the filesystems that have been defined in a cluster.

```
cmgr> show resources of resource_type filesystem in cluster eagan
/disk4 type filesystem
/disk5 type filesystem
/disk2 type filesystem
/disk3 type filesystem
/disk1 type filesystem
cmgr> test resource /disk4 of resource_type filesystem in cluster eagan on node cm1
Success: *** testing node resources on node cm1 ***
Success: *** testing all filesystem resources on node cm1 ***
Success: successfully mounted filesystem: /disk4
Success: overall exit status:success, tests failed:0, total tests executed:1
cmgr>
```

The following example tests a filesystem in destructive mode:

```
cmgr> test resource /disk4 of resource_type filesystem in cluster eagan on node cm1 destructive
Warning: executing the tests in destructive mode
Success: *** testing node resources on node cm1 ***
Success: *** testing all filesystem resources on node cm1 ***
Success: successfully mounted filesystem: /disk4
```

```
Success: overall exit status:success, tests failed:0, total tests executed:1
cmgr>
```

8.3.3.3 Testing NFS Filesystems

You can use the Cluster Manager CLI to test the NFS filesystems configured in a cluster. This test checks if the specified NFS filesystem is configured correctly and, in addition, checks whether the volume the NFS filesystem will reside on is configured correctly.

When you are using the Cluster Manager CLI, use the following command to test an NFS filesystem:

```
cmgr> test resource A of resource_type NFS on cluster B [on node C node D ...]
```

The following example tests an NFS filesystem:

```
cmgr> test resource /disk4 of resource_type NFS in cluster eagan
Success: *** testing node resources on node cm1 ***
Success: *** testing all NFS resources on node cm1 ***
Success: *** testing node resources on node cm2 ***
Success: *** testing all NFS resources on node cm2 ***
Success: overall exit status:success, tests failed:0, total tests executed:2
cmgr>
```

8.3.3.4 Testing statd Resources

You can use the Cluster Manager CLI to test the statd resources configured in a cluster. When you are using the Cluster Manager CLI, use the following command to test an NFS filesystem:

```
cmgr> test resource A of resource_type statd on cluster B [on node C node D ...]
```

The following example tests a statd resource:

```
cmgr> test resource /disk1/statmon of resource_type statd in cluster eagan
Success: *** testing node resources on node cm1 ***
Success: *** testing all statd resources on node cm1 ***
Success: *** testing node resources on node cm2 ***
Success: *** testing all statd resources on node cm2 ***
Success: overall exit status:success, tests failed:0, total tests executed:2
cmgr>
```

8.3.3.5 Testing Netscape-web Resources

You can use the Cluster Manager CLI to test the Netscape Web resources configured in a cluster.

When you are using the Cluster Manager CLI, use the following command to test a Netscape-web resource:

```
cmgr> test resource A of resource_type Netscape_web on cluster B [on node C  node D ...]
```

The following example tests a Netscape-web resource. In this example, the Netscape-web resource on node cm2 failed the diagnostic test.

```
cmgr> test resource nss-enterprise of resource_type Netscape_web in cluster eagan
Success: *** testing node resources on node cm1 ***
Success: *** testing all Netscape_web resources on node cm1 ***
Success: *** testing node resources on node cm2 ***
Success: *** testing all Netscape_web resources on node cm2 ***
```

```
Warning: resource nss-enterprise has invaild script /var/netscape/suitespot/https-ha85 location
Warning: /var/netscape/suitespot/https-ha85/config/magnus.conf must contain the
"Port" parameter
Warning: /var/netscape/suitespot/https-ha85/config/magnus.conf must contain the
"Address" parameter
Warning: resource nss-enterprise of type Netscape_web failed
Success: overall exit status:failed, tests failed:1, total tests executed:2
Failed to execute FailSafe tests/diagnostics ha
test command failed
cmgr>
```

8.3.3.6 Testing Resource Groups

You can use the Cluster Manager CLI to test a resource group. This test cycles through the resource tests for all of the resources defined for a resource group. Resource tests are performed only on nodes in the resource group's application failover domain.

The Cluster Manager CLI uses the following syntax for the commands that test resource groups:

```
cmgr> test resource_group A in cluster B [on node C node D ...]
```

The following example tests a resource group. This example first uses a CLI `show` command to display the resource groups that have been defined in a cluster.

```
cmgr> show resource_groups in cluster eagan
Resource Groups:
        nfs2
        informix
cmgr> test resource_group nfs2 in cluster eagan on machine cm1
Success: *** testing node resources on node cm1 ***
Success: testing resource /disk4 of resource type NFS on node cm1
Success: testing resource /disk3 of resource type NFS on node cm1
Success: testing resource /disk3/statmon of resource type statd on node cm1
Success: testing resource 128.162.19.45 of resource type IP_address on node cm1
Success: testing resource /disk4 of resource type filesystem on node cm1
Success: testing resource /disk3 of resource type filesystem on node cm1
Success: testing resource dmf1 of resource type volume on node cm1
Success: testing resource dmfjournals of resource type volume on node cm1
Success: overall exit status:success, tests failed:0, total tests executed:16
cmgr>
```

8.3.4 Testing Failover Policies with the Cluster Manager CLI

You can use the Cluster Manager CLI to test whether a failover policy is defined correctly. This test checks the failover policy by validating the policy script, failover attributes, and whether the application failover domain consists of valid nodes from the cluster.

The Cluster Manager CLI uses the following syntax for the commands that test a failover policy:

```
cmgr> test failover_policy A in cluster B [on node C node D ...]
```

The following example tests a failover policy. This example first uses a CLI `show` command to display the failover policies that have been defined in a cluster.

```
cmgr> show failover_policies
Failover Policies:
```

```
        reverse
        ordered-in-order
cmgr> test failover_policy reverse in cluster eagan
Success: *** testing node resources on node cm1 ***
Success: testing policy reverse on node cm1
Success: *** testing node resources on node cm2 ***
Success: testing policy reverse on node cm2
Success: overall exit status:success, tests failed:0, total tests executed:2
cmgr>
```

9 Linux FailSafe Recovery

This chapter provides information on FailSafe system recovery, and includes sections on the following topics:

- Section 9.1, *Overview of FailSafe System Recovery*
- Section 9.2, *FailSafe Log Files*
- Section 9.3, *Node Membership and Resets*
- Section 9.4, *Status Monitoring*
- Section 9.5, *Dynamic Control of FailSafe Services*
- Section 9.6, *Recovery Procedures*

9.1 Overview of FailSafe System Recovery

When a FailSafe system experiences problems, you can use some of the FailSafe features and commands to determine where the problem is.

FailSafe provides the following tools to evaluate and recover from system failure:

- Log files
- Commands to monitor status of system components
- Commands to start, stop, and fail over highly available services

Keep in mind that the FailSafe logs may not detect system problems that do not translate into FailSafe problems. For example, if a CPU goes bad, or hardware maintenance is required, FailSafe may not be able to detect and log these failures.

In general, when evaluating system problems of any nature on a FailSafe configuration, you should determine whether you need to shut down a node to address those problems. When you shut down a node, perform the following steps:

1. Stop FailSafe services on that node
2. Shut down the node to perform needed maintenance and repair
3. Start up the node
4. Start FailSafe services on that node

It is important that you explicitly stop FailSafe services before shutting down a node, where possible, so that FailSafe does not interpret the node shutdown as node failure. If FailSafe interprets the service interruption as node failure, there could be unexpected ramifications, depending on how you have configured your resource groups and your application failover domain.

When you shut down a node to perform maintenance, you may need to change your FailSafe configuration to keep your system running.

9.2 FailSafe Log Files

Linux FailSafe maintains system logs for each of the FailSafe daemons. You can customize the system logs according to the level of logging you wish to maintain.

For information on setting up log configurations, see Section 5.6, *Linux FailSafe System Log Configuration*.

Log messages can be of the following types:

Normal

Normal messages report on the successful completion of a task. An example of a normal message is as follows:

```
Wed Sep 2 11:57:25.284 <N ha_gcd cms 10185:0>
Delivering TOTAL membership (S# 1, GS# 1)
```

Error/Warning

Error or warning messages indicate that an error has occurred or may occur soon. These messages may result from using the wrong command or improper syntax. An example of a warning message is as follows:

```
Wed Sep 2 13:45:47.199 <W crsd crs 9908:0
crs_config.c:634> CI_ERR_NOTFOUND, safer - no
such node
```

Syslog Messages

All normal and error messages are also logged to `syslog`. Syslog messages include the symbol `<CI>` in the header to indicate they are cluster-related messages. An example of a syslog message is as follows:

```
Wed Sep 2 12:22:57 6X:safe syslog: <<CI>
ha_cmsd misc 10435:0> CI_FAILURE, I am not part
of the enabled cluster anymore
```

Debug

Debug messages appear in the log group file when the logging level is set to debug0 or higher (using the GUI) or 10 or higher (using the CLI).

Many megabytes of disk space can be consumed on the server when debug levels are used in a log configuration.

Examining the log files should enable you to see the nature of the system error. Noting the time of the error and looking at the log files to note the activity of the various daemons immediately before error occurred, you may be able to determine what situation existed that caused the failure.

9.3 Node Membership and Resets

In looking over the actions of a FailSafe system on failure to determine what has gone wrong and how processes have transferred, it is important to consider the concept of node membership. When failover occurs, the runtime failover domain can include only those nodes that are in the cluster membership.

9.3.1 Node Membership and Tie-Breaker Node

Nodes can enter into the cluster membership only when they are not disabled and they are in a known state. This ensures that data integrity is maintained because only nodes within the cluster membership can access the shared storage. If nodes outside the membership and not controlled by FailSafe were able to access the shared storage, two nodes might try to access the same data at the same time, a situation that would result in data corruption. For this reason, disabled nodes do not participate in the membership computation. Note that no attempt is made to reset nodes that are configured disabled before confirming the cluster membership.

Node membership in a cluster is based on a quorum majority. For a cluster to be enabled, more than 50% of the nodes in the cluster must be in a known state, able to talk to each other, using heartbeat control networks. This quorum determines which nodes are part of the cluster membership that is formed.

If there are an even number of nodes in the cluster, it is possible that there will be no majority quorum; there could be two sets of nodes, each consisting of 50% of the total number of node, unable to communicate with the other set of nodes. In this case, FailSafe uses the node that has been configured as the tie-breaker node when you configured your FailSafe parameters. If no tie-breaker node was configured, FailSafe uses the enabled node with the lowest node id number.

For information on setting tie-breaker nodes, see Section 5.4.4, *Linux FailSafe HA Parameters*.

The nodes in a quorum attempt to reset the nodes that are not in the quorum. Nodes that can be reset are declared `DOWN` in the membership, nodes that could not be reset are declared `UNKNOWN`. Nodes in the quorum are `UP`.

If a new majority quorum is computed, a new membership is declared whether any node could be reset or not.

If at least one node in the current quorum has a current membership, the nodes will proceed to declare a new membership if they can reset at least one node.

If all nodes in the new tied quorum are coming up for the first time, they will try to reset and proceed with a new membership only if the quorum includes the tie-breaker node.

If a tied subset of nodes in the cluster had no previous membership, then the subset of nodes in the cluster with the tie-breaker node attempts to reset nodes in the other subset of nodes in the cluster. If at least one node reset succeeds, a new membership is confirmed.

If a tied subset of nodes in the cluster had previous membership, the nodes in one subset of nodes in the cluster attempt to reset nodes in the other subset of nodes in the cluster. If at least one node reset succeeds, a new membership is confirmed. The subset of nodes in the cluster with the tie-breaker node resets immediately, the other subset of nodes in the cluster attempts to reset after some time.

Resets are done through system controllers connected to tty ports through serial lines. Periodic serial line monitoring never stops. If the estimated serial line monitoring failure interval and the estimated heartbeat loss interval overlap, we suspect a power failure at the node being reset.

9.3.2 No Membership Formed

When no cluster membership is formed, you should check the following areas for possible problems:

- Is the cluster membership daemon, `ha_cmsd` running? Is the database daemon, `cdbd`, running?
- Can the nodes communicate with each other?
  - Are the control networks configured as heartbeat networks?
- Can the control network addresses be pinged from peer nodes?
- Are the quorum majority or tie rules satisfied?

  Look at the `cmsd` log to determine membership status.
- If a reset is required, are the following conditions met?
  - Is the node control daemon, `crsd`, up and running?
  - Is the reset serial line in good health?

    You can look at the `crsd` log for the node you are concerned with, or execute an `admin ping` and `admin reset` command on the node to check this.

9.3.3 No Membership Formed

When no cluster membership is formed, you should check the following areas for possible problems:

- Is the cluster membership daemon, `ha_cmsd` running? Is the database daemon, `cdbd`, running?
- Can the nodes communicate with each other?
  - Are the control networks configured as heartbeat networks?
- Can the control network addresses be pinged from peer nodes?
- Are the quorum majority or tie rules satisfied?

  Look at the `cmsd` log to determine membership status.
- If a reset is required, are the following conditions met?
  - Is the node control daemon, `crsd`, up and running?
  - Is the reset serial line in good health?

    You can look at the `crsd` log for the node you are concerned with, or execute an `admin ping` and `admin reset` command on the node to check this.

9.4 Status Monitoring

FailSafe allows you to monitor and check the status of specified clusters, nodes, resources, and resource groups. You can use this feature to isolate where your system is encountering problems.

With the FailSafe Cluster Manager GUI Cluster View, you can monitor the status of the FailSafe components continuously through their visual representation. Using the FailSafe Cluster Manger CLI, you can display the status of the individual components by using the `show` command.

For information on status monitoring and on the meaning of the states of the FailSafe components, see Section 7.4, *System Status*.

9.5 Dynamic Control of FailSafe Services

FailSafe allows you to perform a variety of administrative tasks that can help you troubleshoot a system with problems without bringing down the entire system. These tasks include the following:

- You can add or delete nodes from a cluster without affecting the FailSafe services and the applications running in the cluster
- You can add or delete a resource group without affecting other online resource groups
- You can add or delete resources from a resource group while it is still online
- You can change FailSafe parameters such as the heartbeat interval and the node timeout and have those values take immediate affect while the services are up and running
- You can start and stop FailSafe services on specified nodes
- You can move a resource group online, or take it offline
- You can stop the monitoring of a resource group by putting the resource group into maintenance mode. This is not an expensive operation, as it does not stop and start the resource group, it just puts the resource group in a state where it is not available to FailSafe.
- You can reset individual nodes

For information on how to perform these tasks, see Chapter 5, *Linux FailSafe Cluster Configuration*, and Chapter 7, *Linux FailSafe System Operation*.

9.6 Recovery Procedures

The following sections describe various recovery procedures you can perform when different failsafe components fail. Procedures for the following situations are provided:

- Section 9.6.1, *Cluster Error Recovery*
- Section 9.6.2, *Node Error recovery*
- Section 9.6.3, *Resource Group Maintenance and Error Recovery*
- Section 9.6.4, *Resource Error Recovery*
- Section 9.6.5, *Control Network Failure Recovery*
- Section 9.6.6, *Serial Cable Failure Recovery*
- Section 9.6.7, *CDB Maintenance and Recovery*
- Section 9.6.8, *FailSafe Cluster Manager GUI and CLI Inconsistencies*

9.6.1 Cluster Error Recovery

Follow this procedure if status of the cluster is UNKNOWN in all nodes in the cluster.

1. Check to see if there are control networks that have failed (see Section 9.6.5, *Control Network Failure Recovery*).
2. At least 50% of the nodes in the cluster must be able to communicate with each other to have an active cluster (Quorum requirement). If there are not sufficient nodes in the cluster

that can communicate with each other using control networks, stop HA services on some of the nodes so that the quorum requirement is satisfied.

3. If there are no hardware configuration problems, detach all resource groups that are online in the cluster (if any), stop HA services in the cluster, and restart HA services in the cluster.

The following `cluster_mgr` command detaches the resource group `web-rg` in cluster `web-cluster`:

```
cmgr> admin detach resource_group web-rg in cluster web-cluster
```

To stop HA services in the cluster `web-cluster` and ignore errors (`force` option), use the following `cluster_mgr` command:

```
cmgr> stop ha_services for cluster web-cluster force
```

To start HA services in the cluster `web-cluster`, use the following `cluster_mgr` command:

```
cmgr> start ha_services for cluster web-cluster
```

9.6.2 Node Error recovery

Follow this procedure if the status of a node is UNKNOWN in an active cluster:

1. Check to see if the control networks in the node are working (see Section 9.6.5, *Control Network Failure Recovery*).
2. Check to see if the serial reset cables to reset the node are working (see Section 9.6.6, *Serial Cable Failure Recovery*).
3. If there are no hardware configuration problems, stop HA services in the node and restart HA services.

   To stop HA services in the node `web-node3` in the cluster `web-cluster`, ignoring errors (`force` option), use the following `cluster_mgr` command

   ```
   cmgr> stop ha_services in node web-node3 for cluster web-cluster
   force
   ```

   To start HA services in the node `web-node3` in the cluster `web-cluster`, use the following `cluster_mgr` command:

   ```
   cmgr> start ha_services in node web-node3 for cluster web-cluster
   ```

9.6.3 Resource Group Maintenance and Error Recovery

To do simple maintenance on an application that is part of the resource group, use the following procedure. This procedure stops monitoring the resources in the resource group when maintenance mode is on. You need to turn maintenance mode off when application maintenance is done.

If there is node failure on the node where resource group maintenance is being performed, the resource group is moved to another node in the failover policy domain.

---

1. To put a resource group `web-rg` in maintenance mode, use the following `cluster_mgr` command:

```
cmgr> admin maintenance_on resource_group web-rg in cluster web-cluster
```

2. The resource group state changes to `ONLINE_MAINTENANCE`. Do whatever application maintenance is required. (Rotating application logs is an example of simple application maintenance).

3. To remove a resource group `web-rg` from maintenance mode, use the following `cluster_mgr` command:

```
cmgr> admin maintenance_off resource_group web-rg in cluster
web-cluster
```

   The resource group state changes back to `ONLINE`.

You perform the following procedure when a resource group is in an `ONLINE` state and has an SRMD `EXECUTABLE ERROR`.

1. Look at the SRM logs (default location: `/var/log/failsafe/srmd_node name`) to determine the cause of failure and the resource that has failed.

2. Fix the cause of failure. This might require changes to resource configuration or changes to resource type stop/start/failover action timeouts.

3. After fixing the problem, move the resource group offline with the `force` option and then move the resource group online.

   The following `cluster_mgr` command moves the resource group `web-rg` in the cluster `web-cluster` offline and ignores any errors:

```
cmgr> admin offline resource_group web-rg in cluster web-cluster
force
```

   The following `cluster_mgr` command moves the resource group `web-rg` in the cluster `web-cluster` online:

```
cmgr> admin online resource_group web-rg in cluster web-cluster
```

   The resource group `web-rg` should be in an `ONLINE` state with no error.

You use the following procedure when a resource group is not online but is in an error state. Most of these errors occur as a result of the exclusivity process. This process, run when a resource group is brought online, determines if any resources are already allocated somewhere in the failure domain of a resource group. Note that exclusivity scripts return that a resource is allocated on a node if the script fails in any way. In other words, unless the script can determine that a resource is not present, it returns a value indicating that the resource is allocated.

Some possible error states include: `SPLIT RESOURCE GROUP (EXCLUSIVITY)`, `NODE NOT AVAILABLE (EXCLUSIVITY)`, `NO AVAILABLE NODES` in failure domain. See Section 7.4.3, *Resource Group Status*, for explanations of resource group error codes.

1. Look at the `failsafe` and SRM logs (default directory: `/var/log/failsafe`, files: `failsafe_`*nodename*, `srmd_`*nodename*) to determine the cause of the failure and the resource that failed.

For example, say the task of moving a resource group online results in a resource group with error state `SPLIT RESOURCE GROUP (EXCLUSIVITY)`. This means that parts of a resource group are allocated on at least two different nodes. One of the failsafe logs will have the description of which nodes are believed to have the resource group partially allocated.

At this point, look at the `srmd` logs on each of these machines to see what resources are believed to be allocated. In some cases, a misconfigured resource will show up as a resource which is allocated. This is especially true for `Netscape_web` resources.

2. Fix the cause of the failure. This might require changes to resource configuration or changes to resource type start/stop/exclusivity timeouts.
3. After fixing the problem, move the resource group offline with the `force` option and then move the resource group online.

There are a few double failures that can occur in the cluster which will cause resource groups to remain in a non-highly-available state. At times a resource group might get stuck in an offline state. A resource group might also stay in an error state on a node even when a new node joins the cluster and the resource group can migrate to that node to clear the error.

When these circumstances arise, the correct action should be as follows:

1. Try to move the resource group online if it is offline.
2. If the resource group is stuck on a node, detach the resource group, then bring it online again. This should clear many errors.
3. If detaching the resource group does not work, force the resource group offline, then bring it back online.
4. If commands appear to be hanging or not working properly, detach all resource groups, then shut down the cluster and bring all resource groups back online.

See Section 7.5.2, *Taking a Resource Group Offline*, for information on detaching resource groups and forcing resource groups offline.

9.6.4 Resource Error Recovery

You use this procedure when a resource that is not part of a resource group is in an `ONLINE` state with error. This can happen when the addition or removal of resources from a resource group fails.

1. Look at the SRM logs (default location: `/var/log/failsafe/srmd_`*nodename*) to determine the cause of failure and the resource that has failed.
2. Fix the cause of failure. This might require changes to resource configuration or changes to resource type stop/start/failover action timeouts.
3. After fixing the problem, move the resource offline with the `force` option of the Cluster Manager CLI `admin offline` command:

```
cmgr> admin offline_force resource web-srvr of resource_type
Netscape_Web in cluster web-cluster
```

Executing this command removes the error state of resource `web-srvr` of type `Netscape_Web`, making it available to be added to a resource group.

You can also use the Cluster Manager GUI to clear the error state for the resource. To do this, you select the "Recover a Resource" task from the "Resources and Resource Types" category of the FailSafe Manager.

9.6.5 Control Network Failure Recovery

Control network failures are reported in `cmsd` logs. The default location of `cmsd` log is `/var/log/failsafe/cmsd_`*nodename*. Follow this procedure when the control network fails:

1. Use the `ping` command to check whether the control network IP address is configured in the node.
2. Check node configuration to see whether the control network IP addresses are correctly specified.

   The following `cluster_mgr` command displays node configuration for `web-node3:`

   ```
   cmgr> show node web-node3
   ```

3. If IP names are specified for control networks instead of IP addresses in XX.XX.XX.XX notation, check to see whether IP names can be resolved using DNS. It is recommended that IP addresses are used instead of IP names.
4. Check whether the heartbeat interval and node timeouts are correctly set for the cluster. These HA parameters can seen using `cluster_mgr show ha_parameters` command.

9.6.6 Serial Cable Failure Recovery

Serial cables are used for resetting a node when there is a node failure. Serial cable failures are reported in `crsd` logs. The default location for the `crsd` log is `/var/log/failsafe/crsd_`*nodename*.

1. Check the node configuration to see whether serial cable connection is correctly configured.

   The following `cluster_mgr` command displays node configuration for `web-node3`

   ```
   cmgr> show node web-node3
   ```

   Use the `cluster_mgr admin ping` command to verify the serial cables.

   ```
   cmgr> admin ping node web-node3
   ```

The above command reports serial cables problems in node `web-node3`.

9.6.7 CDB Maintenance and Recovery

When the entire configuration database (CDB) must be reinitialized, execute the following command:

```
# /usr/cluster/bin/cdbreinit /var/cluster/cdb/cdb.db
```

This command will restart all cluster processes. The contents of the configuration database will be automatically synchronized with other nodes if other nodes in the pool are available.

Otherwise, the CDB will need to be restored from backup at this point. For instructions on backing up and restoring the CDB, see Section 7.8, *Backing Up and Restoring Configuration With Cluster Manager CLI*.

9.6.8 FailSafe Cluster Manager GUI and CLI Inconsistencies

If the FailSafe Cluster Manager GUI is displaying information that is inconsistent with the FailSafe `cluster_mgr` command, restart cad process on the node to which Cluster Manager GUI is connected to by executing the following command:

```
# killall cad
```

The cluster administration daemon is restarted automatically by the `cmond` process.

10 Upgrading and Maintaining Active Clusters

When a Linux FailSafe system is running, you may need to perform various administration procedures without shutting down the entire cluster. This chapter provides instructions for performing upgrade and maintenance procedures on active clusters. It includes the following procedures:

- Section 10.1, *Adding a Node to an Active Cluster*
- Section 10.2, *Deleting a Node from an Active Cluster*
- Section 10.4, *Upgrading OS Software in an Active Cluster*
- Section 10.5, *Upgrading FailSafe Software in an Active Cluster*
- Section 10.6, *Adding New Resource Groups or Resources in an Active Cluster*
- Section 10.7, *Adding a New Hardware Device in an Active Cluster*

10.1 Adding a Node to an Active Cluster

Use the following procedure to add a node to an active cluster. This procedure begins with the assumption that `cluster_admin`, `cluster_control`, `cluster_ha` and `failsafe2` products are already installed in this node.

1. Check control network connections from the node to the rest of the cluster using `ping` command. Note the list of control network IP addresses.
2. Check the serial connections to reset this node. Note the name of the node that can reset this node.
3. Run node diagnostics. For information on FailSafe diagnostic commands, see Chapter 8, *Testing Linux FailSafe Configuration*.
4. Make sure `sgi-cad`, `sgi-crsd`, `sgi-cmsd`, and `sgi-gcd` entries are present in the `/etc/services` file. The port numbers for these processes should match the port numbers in other nodes in the cluster.

   Example entries:

   ```
   sgi-cad         7200/tcp      # SGI cluster admin daemon
   sgi-crsd        7500/udp      # SGI cluster reset services daemon
   sgi-cmsd        7000/udp      # SGI cluster membership Daemon
   sgi-gcd         8000/udp      # SGI group communication Daemon
   ```

5. Check if cluster processes (`cad`, `cmond`, `crsd`) are running.

   ```
   # ps -ef | grep cad
   ```

   If cluster processes are not running, run the `cdbreinit` command.

   ```
   # /usr/lib/failsafe/bin/cdbreinit /var/lib/failsafe/cdb/cdb.db
    Killing cdbd...
    Removing database header file /var/lib/failsafe/cdb/cdb.db...
    Preparing to delete database directory /var/lib/failsafe/cdb/cdb.db# !!
    Continue[y/n]y
    Removing database directory /var/lib/failsafe/cdb/cdb.db#...
   ```

```
Deleted CDB database at /var/lib/failsafe/cdb/cdb.db
Recreating new CDB database at /var/lib/failsafe/cdb/cdb.db with cdb-exitop...
 cdbd
 Created standard CDB database in /var/lib/failsafe/cdb/cdb.db

 Please make sure that "sgi-cad" service is added to /etc/services file
 If not, add the entry and restart cluster processes.
 Please refer to FailSafe administration manual for more
 information.

 Modifying CDB database at /var/lib/failsafe/cdb/cdb.db with cluster_ha-exitop...
 Modified standard CDB database in /var/lib/failsafe/cdb/cdb.db

 Please make sure that "sgi-cmsd" and "sgi-gcd" services are added
 to /etc/services file before starting HA services.
 Please refer to FailSafe administration manual for more
 information.

 Starting cluster control processes with cluster_control-exitop...

 Please make sure that "sgi-crsd" service is added to /etc/services file
 If not, add the entry and restart cluster processes.
 Please refer to FailSafe administration manual for more
 information.

 Started cluster control processes
 Restarting cluster admin processes with failsafe-exitop...
```

6. Use `cluster_mgr` template (`/usr/lib/failsafe/cmgr-templates/cmgr-create-node`) or `cluster_mgr` command to define the node.

   This node must be defined from one of nodes that is already in the cluster.

   ---

7. Use the `cluster_mgr` command to add the node to the cluster.

   For example: The following `cluster_mgr` command adds the node `web-node3` to the cluster `web-cluster`:

   ```
   cmgr> modify cluster web-cluster
   Enter commands, when finished enter either "done" or "cancel"

   web-cluster ? add node web-node3
   web-cluster ? done
   ```

8. You can start HA services on this node using the `cluster_mgr` command. For example, the following `cluster_mgr` command starts HA services on node `web-node3` in cluster `web-cluster`:

   ```
   cmgr> start ha_services on node web-node3 in cluster web-cluster
   ```

9. Remember to add this node to the failure domain of the relevant failover policy. In order to do this, the entire failover policy must be re-defined, including the additional node in the failure domain.

10.2 Deleting a Node from an Active Cluster

Use the following procedure to delete a node from an active cluster. This procedure begins with the assumption that the node status is UP.

1. If resource groups are online on the node, use the `cluster_mgr` command to move them to another node in the cluster.

   To move the resource groups to another node in the cluster, there should be another node available in the failover policy domain of the resource group. If you want to leave the resource groups running in the same node, use the `cluster_mgr` command to detach the resource group. For example, the following command would leave the resource group `web-rg` running in the same node in the cluster `web-cluster`.

   ```
   cmgr> admin detach resource_group "web-rg" in cluster web-cluster
   ```

2. Delete the node from the failure domains of any failover policies which use the node. In order to do this, the entire failover policy must be re-defined, deleting the affected node from the failure domain.

3. To stop HA services on the node `web-node3`, use the following `cluster_mgr` command. This command will move all the resource groups online on this node to other nodes in the cluster if possible.

   ```
   cmgr> stop ha_services on node web-node3 for cluster web-cluster
   ```

   If it is not possible to move resource groups that are online on node `web-node3`, the above command will fail. The `force` option is available to stop HA services in a node even in the case of an error. Should there be any resources which can not be moved offline or deallocated properly, a side-effect of the offline force command will be to leave these resources allocated on the node.

Perform Steps 4, 5, 6, and 7 if the node must be deleted from the configuration database.

4. Delete the node from the cluster. To delete node `web-node3` from `web-cluster` configuration, use the following `cluster_mgr` command:

   ```
   cmgr> modify cluster web-cluster
   Enter commands, when finished enter either "done" or "cancel"
   web-cluster ? remove node web-node3
   web-cluster ? done
   ```

5. Remove node configuration from the configuration database.

   The following `cluster_mgr` command deletes the `web-node3` node definition from the configuration database.

   ```
   cmgr> delete node web-node3
   ```

6. Stop all cluster processes and delete the configuration database.

   The following commands stop cluster processes on the node and delete the configuration database.

```
# /etc/rc.d/init.d/failsafe stop
# killall cdbd
# cdbdelete /var/lib/failsafe/cdb/cdb.db
```

7. Disable cluster and HA processes from starting when the node boots. The following commands perform those tasks:

```
# fsconfig failsafe off
```

10.3 Changing Control Networks in a Cluster

Use the following procedure to change the control networks in a currently active cluster. This procedure is valid for a two-node cluster consisting of nodes `node1` and `node2`. In this procedure, you must complete each step before proceeding to the next step.

Do not perform any other administration operations during this procedure.

---

1. From any node, stop HA services on the cluster. Make sure all HA processes have exited on both nodes.
2. From `node2`, stop the cluster processes on `node2`:

```
# /etc/rc.d/init.d/fs_cluster stop
# killall cdbd
```

   Make sure the `cdbd` process have been killed on `node2`.

3. From `node1`, modify the `node1` and `node2` definition. Use the following `cmgr` commands:

```
cmgr> modify node node1
Enter commands, when finished enter either "done" or "cancel"
node1?> remove nic old nic address
node1> add nic nnew nic address
NIC - new nic address set heartbeat to ...
NIC - new nic address set ctrl_msgs to ...
NIC - new nic address set priority to ...
NIC - new nic address done
node1? done
```

   Repeat the same procedure to modify `node2`.

4. From `node1`, check if the `node1` and `node2` definitions are correct. Using `cmgr` on `node1`, execute the following commands to view the node definitions:

```
cmgr> show node node1
cmgr> show node node2
```

5. On both `node1` and `node2`, modify the network interface IP addresses in `/etc/failsafe/config/netif.options` and execute `ifconfig` to configure the new IP addresses on `node1` and `node2`. Verify that the IP addresses match the node definitions in the CDB.
6. From `node1`, stop the cluster process on `node1`:

```
# /etc/rc.d/init.d/fs_cluster stop
# killall cdbd
```

Make sure the `cdbd` process have been killed on `node1`.

7. From `node2`, execute the following command to start cluster process on `node2`:

```
# /usr/cluster/bin/cdbreinit /var/cluster/cdb/cdb.db
```

Answer **`y`** to the prompt the appears.

8. From `node1`, start cluster processes on `node1`:

```
# /etc/rc.d/init.d/fs_cluster start
```

The following messages should appear in the `SYSLOG` on `node2`:

```
Starting to receive CDB sync series from machine node1_node_id>
...
Finished receiving CDB sync series from machine node1_node_id
```

Wait for approximately sixty seconds for the sync to complete.

9. From any node, start HA services in the cluster.

10.4 Upgrading OS Software in an Active Cluster

When you upgrade your OS software in an active cluster, you perform the upgrade on one node at a time.

If the OS software upgrade does not require reboot or does not impact the FailSafe software, there is no need to use the OS upgrade procedure. If you do not know whether the upgrade will impact FailSafe software or if the OS upgrade requires a machine reboot, follow the upgrade procedure described below.

The following procedure upgrades the OS software on node `web-node3`.

1. If resource groups are online on the node, use a `cluster_mgr` command to move them another node in the cluster. To move the resource group to another node in the cluster, there should be another node available in the failover policy domain of the resource group.

   The following `cluster_mgr` command moves resource group `web-rg` to another node in the cluster `web-cluster:`

```
cmgr> admin move resource_group web-rg in cluster web-cluster
```

2. To stop HA services on the node `web-node3`, use the following `cluster_mgr` command. This command will move all the resource groups online on this node to other nodes in the cluster if possible.

```
cmgr> stop ha_services on node web-node3 for cluster web-cluster
```

   If it is not possible to move resource groups that are online on node `web-node3`, the above command will fail. You can use the `force` option to stop HA services in a node even in the case of an error.

3. Perform the OS upgrade in the node `web-node3`.

4. After the OS upgrade, make sure cluster processes (cmond, cad, crsd) are running.
5. Restart HA services on the node. The following cluster_mgr command restarts HA services on the node:

```
cmgr> start ha_services on node web-node3 for cluster web-cluster
```

   Make sure the resource groups are running on the most appropriate node after restarting HA services.

10.5 Upgrading FailSafe Software in an Active Cluster

When you upgrade FailSafe software in an active cluster, you upgrade one node at a time in the cluster.

The following procedure upgrades FailSafe on node web-node3.

1. If resource groups are online on the node, use a cluster_mgr command to move them another node in the cluster. To move the resource group to another node in the cluster, there should be another node available in the failover policy domain of the resource group.

   The following cluster_mgr command moves resource group web-rg to another node in the cluster web-cluster:

```
cmgr> admin move resource_group web-rg in cluster web-cluster
```

2. To stop HA services on the node web-node3, use the following cluster_mgr command. This command will move all the resource groups online on this node to other nodes in the cluster if possible.

```
cmgr> stop ha_services on node web-node3 for cluster web-cluster
```

   If it is not possible to move resource groups that are online on node web-node3, the above command will fail. You can use the force option to stop HA services in a node even in the case of an error.

3. Stop all cluster processes running on the node.

```
# /etc/rc.dinit.d/failsafe stop
```

4. Perform the FailSafe upgrade in the node web-node3.
5. After the FailSafe upgrade, check whether cluster processes (cmond, cad, crsd) are running. If not, restart cluster processes:

```
# fsconfig failsafe on; /etc/rc.d/init.d/failsafe start
```

6. Restart HA services on the node. The following cluster_mgr command restarts HA services on the node:

```
cmgr> start ha_services on node web-node3 for cluster web-cluster
```

   Make sure the resource groups are running on the most appropriate node after restarting HA services.

10.6 Adding New Resource Groups or Resources in an Active Cluster

The following procedure describes how to add a resource group and resources to an active cluster. To add resources to an existing resource group, perform resource configuration (Step 4), resource diagnostics (Step 5) and add resources to the resource group (Step 6).

1. Identify all the resources that have to be moved together. These resources running on a node should be able to provide a service to the client. These resources should be placed in a resource group. For example, Netscape webserver `mfg-web`, its IP address 192.26.50.40, and the filesystem `/shared/mfg-web` containing the web configuration and document pages should be placed in the same resource group (for example, `mfg-web-rg`).
2. Configure the resources in all nodes in the cluster where the resource group is expected to be online. For example, this might involve configuring netscape web server `mfg-web` on nodes `web-node1` and `web-node2` in the cluster.
3. Create a failover policy. Determine the type of failover attribute required for the resource group. The `cluster_mgr` template (`/usr/lib/failsafe/cmgr-templates/cmgr-create-failover_policy`) can be used to create the failover policy.
4. Configure the resources in configuration database. There are `cluster_mgr` templates to create resources of various resource types in `/usr/lib/failsafe/cmgr-templates` directory. For example, the volume resource, the `/shared/mfg-web` filesystem, the `192.26.50.40 IP_address` resource, and the `mfg-web` Netscape_web resource have to be created in the configuration database. Create the resource dependencies for these resources.
5. Run resource diagnostics. For information on the diagnostic commands, see Chapter 8, *Testing Linux FailSafe Configuration*.
6. Create resource group and add resources to the resource group. The `cluster_mgr` template (`/usr/lib/failsafe/cmgr-templates/cmgr-create-resource_group`) can be used to create resource group and add resources to resource group.

   All resources that are dependent on each other should be added to the resource group at the same time. If resources are added to an existing resource group that is online in a node in the cluster, the resources are also made online on the same node.

10.7 Adding a New Hardware Device in an Active Cluster

When you add hardware devices to an active cluster, you add them one node at a time.

To add hardware devices to a node in an active cluster, follow the same procedure as when you upgrade OS software in an active cluster, as described in Section 10.4, *Upgrading OS Software in an Active Cluster*. In summary:

- You must move the resource groups offline and stop HA services in the node before adding the hardware device.
- After adding the hardware device, make sure cluster processes are running and start HA services on the node.

To include the new hardware device in the configuration database, you must modify your resource configuration and your node configuration, where appropriate.

Glossary

action scripts

The set of scripts that determine how a resource is started, monitored, and stopped. There must be a set of action scripts specified for each resource type. The possible set of action scripts is: `probe`, `exclusive`, `start`, `stop`, `monitor`, and `restart`.

cluster

A collection of one or more **cluster nodes** coupled to each other by networks or other similar interconnections. A cluster is identified by a simple name; this name must be unique within the **pool**. A particular node may be a member of only one cluster.

cluster administrator

The person responsible for managing and maintaining a Linux FailSafe cluster.

cluster configuration database

Contains configuration information about all resources, resource types, resource groups, failover policies, nodes, and clusters.

cluster node

A single Linux image. Usually, a cluster node is an individual computer. The term *node* is also used in this guide for brevity.

control messages

Messages that cluster software sends between the cluster nodes to request operations on or distribute information about cluster nodes and resource groups. Linux FailSafe sends control messages for the purpose of ensuring nodes and groups remain highly available. Control messages and heartbeat messages are sent through a node's network interfaces that have been attached to a control network. A node can be attached to multiple control networks.

A node's control networks should not be set to accept control messages if the node is not a dedicated Linux FailSafe node. Otherwise, end users who run non-Linux FailSafe jobs on the machine can have their jobs killed unexpectedly when Linux FailSafe resets the node.

control network

The network that connects nodes through their network interfaces (typically Ethernet) such that Linux FailSafe can maintain a cluster's high availability by sending heartbeat messages and control messages through the network to the attached nodes. Linux FailSafe uses the highest priority network interface on the control network; it uses a network interface with lower priority when all higher-priority network interfaces on the control network fail.

A node must have at least one control network interface for heartbeat messages and one for control messages (both heartbeat and control messages can be configured to use the same interface). A node can have no more than eight control network interfaces.

dependency list

See **resource dependency list** or **resource type dependency list**.

failover

The process of allocating a **resource group** to another **node** to another, according to a **failover policy**. A failover may be triggered by the failure of a resource, a change in the node membership (such as when a node fails or starts), or a manual request by the administrator.

failover attribute

A string that affects the allocation of a resource group in a cluster. The administrator must specify system-defined attributes (such as **AutoFailback** or **ControlledFailback**), and can optionally supply site-specific attributes.

failover domain

The ordered list of **nodes** on which a particular **resource group** can be allocated. The nodes listed in the failover domain must be within the same cluster; however, the failover domain does not have to include every node in the cluster.The administrator defines the **initial failover domain** when creating a failover policy. This list is transformed into the **running failover domain** by the **failover script**; the runtime failover domain is what is actually used to select the failover node. Linux FailSafe stores the runtime failover domain and uses it as input to the next failover script invocation. The initial and runtime failover domains may be identical, depending upon the contents of the failover script. In general, Linux FailSafe allocates a given resource group to the first node listed in the runtime failover domain that is also in the node membership; the point at which this allocation takes place is affected by the **failover attributes**.

failover policy

The method used by Linux FailSafe to determine the destination node of a failover. A failover policy consists of a **failover domain**, **failover attributes**, and a **failover script**. A failover policy name must be unique within the **pool**.

failover script

A failover policy component that generates a **runtime failover domain** and returns it to the Linux FailSafe process. The Linux FailSafe process applies the failover attributes and then selects the first node in the returned failover domain that is also in the current node membership.

heartbeat messages

Messages that cluster software sends between the nodes that indicate a node is up and running. Heartbeat messages and **control messages** are sent through a node's network interfaces that have been attached to a control network. A node can be attached to multiple control networks.

heartbeat interval

Interval between heartbeat messages. The node timeout value must be at least 10 times the heartbeat interval for proper Linux FailSafe operation (otherwise false failovers may be

triggered). The higher the number of heartbeats (smaller heartbeat interval), the greater the potential for slowing down the network. Conversely, the fewer the number of heartbeats (larger heartbeat interval), the greater the potential for reducing availability of resources.

initial failover domain

The ordered list of nodes, defined by the administrator when a failover policy is first created, that is used the first time a cluster is booted.The ordered list specified by the initial failover domain is transformed into a **runtime failover domain** by the **failover script**; the runtime failover domain is used along with failover attributes to determine the node on which a resource group should reside. With each failure, the failover script takes the current runtime failover domain and potentially modifies it; the initial failover domain is never used again. Depending on the runtime conditions and contents of the failover script, the initial and runtime failover domains may be identical. See also **runtime failover domain**.

key/value attribute

A set of information that must be defined for a particular resource type. For example, for the resource type `filesystem`, one key/value pair might be `mount_point=/fs1` where `mount_point` is the key and `fs1` is the value specific to the particular resource being defined. Depending on the value, you specify either a `string` or `integer` data type. In the previous example, you would specify `string` as the data type for the value `fs1`.

log configuration

A log configuration has two parts: a **log level** and a **log file**, both associated with a **log group**. The cluster administrator can customize the location and amount of log output, and can specify a log configuration for all nodes or for only one node. For example, the `crsd` log group can be configured to log detailed level-10 messages to the `/var/log/failsafe/crsd-foo` log only on the node `foo`, and to write only minimal level-1 messages to the `crsd` log on all other nodes.

log file

A file containing Linux FailSafe notifications for a particular **log group**. A log file is part of the **log configuration** for a log group. By default, log files reside in the `/var/log/failsafe` directory, but the cluster administrator can customize this. Note: Linux FailSafe logs both normal operations and critical errors to `/var/log/messages`, as well as to individual logs for specific log groups.

log group

A set of one or more Linux FailSafe processes that use the same log configuration. A log group usually corresponds to one Linux FailSafe daemon, such as `gcd`.

log level

A number controlling the number of log messages that Linux FailSafe will write into an associated log group's log file. A log level is part of the log configuration for a log group.

node

See **cluster node**

node ID

A 16-bit positive integer that uniquely defines a cluster node. During node definition, Linux FailSafe will assign a node ID if one has not been assigned by the cluster administrator. Once assigned, the node ID cannot be modified.

node membership

The list of nodes in a cluster on which Linux FailSafe can allocate resource groups.

node timeout

If no heartbeat is received from a node in this period of time, the node is considered to be dead. The node timeout value must be at least 10 times the heartbeat interval for proper Linux FailSafe operation (otherwise false failovers may be triggered).

notification command

The command used to notify the cluster administrator of changes or failures in the cluster, nodes, and resource groups. The command must exist on every node in the cluster.

offline resource group

A resource group that is not highly available in the cluster. To put a resource group in offline state, Linux FailSafe stops the group (if needed) and stops monitoring the group. An offline resource group can be running on a node, yet not under Linux FailSafe control. If the cluster administrator specifies the *detach only* option while taking the group offline, then Linux FailSafe will not stop the group but will stop monitoring the group.

online resource group

A resource group that is highly available in the cluster. When Linux FailSafe detects a failure that degrades the resource group availability, it moves the resource group to another node in the cluster. To put a resource group in online state, Linux FailSafe starts the group (if needed) and begins monitoring the group. If the cluster administrator specifies the `attach only` option while bringing the group online, then Linux FailSafe will not start the group but will begin monitoring the group.

owner host

A system that can control a Linux FailSafe node remotely, such as power-cycling the node). Serial cables must physically connect the two systems through the node's system controller port. At run time, the owner host must be defined as a node in the Linux FailSafe pool.

owner TTY name

The device file name of the terminal port (TTY) on the **owner host** to which the system controller serial cable is connected. The other end of the cable connects to the Linux FailSafe node with the system controller port, so the node can be controlled remotely by the owner host.

pool

The entire set of **nodes** involved with a group of clusters. The group of clusters are usually close together and should always serve a common purpose. A replicated database is stored on each node in the pool.

port password

The password for the system controller port, usually set once in firmware or by setting jumper wires. (This is not the same as the node's root password.)

powerfail mode

When powerfail mode is turned `on`, Linux FailSafe tracks the response from a node's system controller as it makes reset requests to a cluster node. When these requests fail to reset the node successfully, Linux FailSafe uses heuristics to try to estimate whether the machine has been powered down. If the heuristic algorithm returns with success, Linux FailSafe assumes the remote machine has been reset successfully. When powerfail mode is turned `off`, the heuristics are not used and Linux FailSafe may not be able to detect node power failures.

process membership

A list of process instances in a cluster that form a process group. There can be one or more processes per node.

resource

A single physical or logical entity that provides a service to clients or other resources. For example, a resource can be a single disk volume, a particular network address, or an application such as a web server. A resource is generally available for use over time on two or more **nodes** in a **cluster**, although it can be allocated to only one node at any given time. Resources are identified by a **resource name** and a **resource type**. Dependent resources must be part of the same **resource group** and are identified in a **resource dependency list**.

resource dependency

The condition in which a resource requires the existence of other resources.

resource group

A collection of **resources**. A resource group is identified by a simple name; this name must be unique within a cluster. Resource groups cannot overlap; that is, two resource groups cannot contain the same resource. All interdependent resources must be part of the same resource group. If any individual resource in a resource group becomes unavailable for its intended use, then the entire resource group is considered unavailable. Therefore, a resource group is the unit of failover for Linux FailSafe.

resource keys

Variables that define a resource of a given resource type. The action scripts use this information to start, stop, and monitor a resource of this resource type.

resource name

The simple name that identifies a specific instance of a **resource type**. A resource name must be unique within a cluster.

resource type

A particular class of **resource**. All of the resources in a particular resource type can be handled in the same way for the purposes of **failover**. Every resource is an instance of

exactly one resource type. A resource type is identified by a simple name; this name must be unique within a cluster. A resource type can be defined for a specific node or for an entire cluster. A resource type that is defined for a node overrides a cluster-wide resource type definition with the same name; this allows an individual node to override global settings from a cluster-wide resource type definition.

resource type dependency

A set of resource types upon which a resource type depends. For example, the `filesystem` resource type depends upon the `volume` resource type, and the `Netscape_web` resource type depends upon the `filesystem` and `IP_address` resource types.

runtime failover domain

The ordered set of nodes on which the resource group can execute upon failures, as modified by the **failover script**. The runtime failover domain is used along with failover attributes to determine the node on which a resource group should reside.See also **initial failover domain**.

start/stop order

Each resource type has a start/stop order, which is a non-negative integer. In a resource group, the start/stop orders of the resource types determine the order in which the resources will be started when Linux FailSafe brings the group online and will be stopped when Linux FailSafe takes the group offline. The group's resources are started in increasing order, and stopped in decreasing order; resources of the same type are started and stopped in indeterminate order. For example, if resource type `volume` has order 10 and resource type `filesystem` has order 20, then when Linux FailSafe brings a resource group online, all volume resources in the group will be started before all filesystem resources in the group.

system controller port

A port sitting on a node that provides a way to power-cycle the node remotely. Enabling or disabling a system controller port in the cluster configuration database (CDB) tells Linux FailSafe whether it can perform operations on the system controller port. (When the port is enabled, serial cables must attach the port to another node, the owner host.) System controller port information is optional for a node in the pool, but is required if the node will be added to a cluster; otherwise resources running on that node never will be highly available.

tie-breaker node

A node identified as a tie-breaker for Linux FailSafe to use in the process of computing node membership for the cluster, when exactly half the nodes in the cluster are up and can communicate with each other. If a tie-breaker node is not specified, Linux FailSafe will use the node with the lowest node ID in the cluster as the tie-breaker node.

type-specific attribute

Required information used to define a resource of a particular resource type. For example, for a resource of type `filesystem`, you must enter attributes for the resource's volume

name (where the filesystem is located) and specify options for how to mount the filesystem (for example, as readable and writable).

Index

A

B

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V