SGI CHASE Product Guide

007–4314–001

CONTRIBUTORS

Written by Jenn Jones

Edited by Rick Thompson

Illustrated by Chrystie Danzer, Chris Wengelski, and Dan Young

Production by Glen Traefald

Engineering contributions by Subodh Bhattacharjya, Luc Dauvergne, Daniel Hurtubise, Tony Kavadias, Linda Lait, Herbert Lewis, Padmanabhan Sreenivasan, and Mayank Vasa

Cover design by Sarah Bolles, Sarah Bolles Design, and Dany Galgani, SGI Technical Publications.

Record of Revision

Version	Description
001	December 2000 Original publication

Contents

Figures

Tables

Examples

About This Guide

This publication documents Clustered Highly Available Server Environment (CHASE) release 1.0, CHASE-FILESERVER-1.0, and CHASE-WEBSERVER-1.0 running on SGI 1450 systems. It assumes that SGI ProPack 1.4 has already been installed on the systems in your cluster.

This document explains how to perform configuration, installation, and administrative tasks specific to CHASE.

Related Publications and Web Sites

The following documents and Web sites contain additional information that may be helpful:

- Apache:
  - Apache product Web page: `http://oss.sgi.com/projects/apache`
  - Apache home page: `http://www.apache.org`
- FailSafe:
  - FailSafe product Web page: `http://oss.sgi.com/projects/failsafe`
  - Manuals available from the SGI online Technical Publications Library `http://techpubs.sgi.com`:
    - *Linux FailSafe Administrator's Guide*
    - *Linux FailSafe Programmer's Guide*
- Samba:
  - Samba product Web page: `http://www.sgi.com/software/samba`
  - Samba home page: `http://www.samba.org`
- SGI ProPack:
  - SGI ProPack site: `http://oss.sgi.com/projects`

  - Manual available from the SGI online Technical Publications Library `http://techpubs.sgi.com`:
    - *SGI ProPack 1.4 for Linux Start Here*
- TP9100:
  - Manual available from the SGI online Technical Publications Library `http://techpubs.sgi.com`:
    - *SGI Total Performance 9100 Storage System Owner's Guide*
- VACM:
  - VACM Web site: `http://www.valinux.com/software/vacm`

Obtaining Publications

To obtain SGI documentation, go to the SGI Technical Publications Library at `http://techpubs.sgi.com`.

Conventions

The following conventions are used throughout this document:

Convention	**Meaning**
`command`	This fixed-space font denotes literal items such as commands, files, routines, path names, signals, messages, and programming language structures.
`manpage`(*x*)	Man page section identifiers appear in parentheses after man page names. The following list describes the identifiers:

1	User commands
1B	User commands ported from BSD
2	System calls
3	Library routines, macros, and opdefs
4	Devices (special files)

4P	Protocols
5	File formats
7	Miscellaneous topics
7D	DWB-related information
8	Administrator commands

Some internal routines (for example, the `_assign_asgcmd_info()` routine) do not have man pages associated with them.

variable	Italic typeface denotes variable entries and words or concepts being defined.
`user input`	This bold, fixed-space font denotes literal items that the user enters in interactive sessions. Output is shown in nonbold, fixed-space font.
[ ]	Brackets enclose optional portions of a command or directive line.
...	Ellipses indicate that a preceding element can be repeated.
#	System shell prompt for the superuser (`root`).

Reader Comments

If you have comments about the technical accuracy, content, or organization of this document, please tell us. Be sure to include the title and document number of the manual with your comments. (Online, the document number is located in the front matter of the manual. In printed manuals, the document number is located at the bottom of each page.)

You can contact us in any of the following ways:

- Send e-mail to the following address:

  `techpubs@sgi.com`

- Use the Feedback option on the Technical Publications Library World Wide Web page:

  `http://techpubs.sgi.com`

- Contact your customer service representative and ask that an incident be filed in the SGI incident tracking system.
- Send mail to the following address:

  Technical Publications
  SGI
  1600 Amphitheatre Pkwy., M/S 535
  Mountain View, California 94043–1351

- Send a fax to the attention of "Technical Publications" at +1 650 932 0801.

We value your comments and will respond to them promptly.

Introduction

This document discusses the CHASE product. It explains how to perform configuration and installation.

Terminology

This document uses the following terminology:

- *CHASE-FILESERVER* (CHASE-FS), which refers to CHASE-BASE-CD plus CHASE-FS-CD
- *CHASE-WEBSERVER* (CHASE-WEB), which refers to CHASE-BASE-CD plus CHASE-WEB-CD

Assumptions

The instructions in this document assume that the base Linux operating system (OS) — either Red Hat version 6.2 or later or SuSE version 6.4 or later — and SGI ProPack 1.4 have already been installed on the systems in your cluster. For more information, see the base OS documentation and *SGI ProPack 1.4 for Linux Start Here*.

CHASE Product Contents

CHASE, the Clustered Highly Available (HA) Server Environment, is designed to bring the ease of use and deployment of HA solutions for business critical applications to our customers. It is based on Linux FailSafe, SGI's open sourced, advanced, scalable, and field proven HA software which has been open sourced to benefit Linux overall. Linux FailSafe is ported to Linux from the IRIX version, IRIS FailSafe.

CHASE for fileserver brings together all the pieces and installation program to deploy a highly available fileserver using NFS and Samba in a cluster environment that is extensible as the business needs grow. Even though many of these components may be available from disparate sources in the open source arena, it is clear that well packaged solutions reduce the time and hassles required for users to assemble it all on their own and hence can be quite cost effective. Another reason customers are willing to pay for such packaged solutions is the support they need for a business critical solution which must be up and running 24x7. SGI, with its world class support infrastructure and the best knowledge base for Linux FailSafe, as it is an incarnation of IRIS FailSafe which has been successfully deployed in the field for five years now, is in the position to offer the best support options to our customers who need the cost-effective HA solutions in the Linux environment.

Product Structure

The CHASE product consists of three software and three hardware bundles. Users are expected to buy a software-only bundle and required hardware bundles.

SC4–CHASE-BASE-1.0 is the CD that contains Linux FailSafe product as well as Performance Co-Pilot (PCP) images for Linux FailSafe.

The three software-only bundles are the following:

- SC4–CHASE-CUSTOM-1.0, which contains SC4–CHASE-BASE-1.0 CD. This bundle can be used to build custom agents for FailSafe.
- SC4–CHASE-FILESERVER-1.0, which contains SC4–CHASE-BASE-1.0 CD and another CD containing CHASE agents for NFS v.2 and Samba v.2.0.6.
- SC4–CHASE-WEBSERVER-1.0, which contains SC4–CHASE-BASE-1.0 CD and another CD containing CHASE agent for Apache Web server.

The three hardware bundles available for IA32 machines are the following:

- CHASE-2NODE-IA32 : First two nodes in IA32 Linux HA Cluster

(9290131)	1	Null Ethernet Cable — RJ45
(018–0691–001)	2	Serial Cables

- CHASE-N-NODE : Expansion infrastructure for any greater than two nodes

(018–0700–001)	2	Ethernet Cable — RJ45
(9470235)	1	10BaseT Hub 8–port
(040–1895–003)	1	Rack Mounting Bracket
(9470248)	1	Power Supply

• CHASE-ADD-IA32 : Additional nodes beyond initial two

(018–0700–001)	1	Ethernet Cable — RJ45
(018–0691–001)	1	Serial Cable

Contents for the CHASE Release

CHASE-BASE CD

The CHASE-BASE CD contains the RPMs providing the core FailSafe infrastructure. This consists of Graphical User Interface daemons as well as clustering (highly available) daemons. It also has plug-ins/agents for basic resources like IP addresses and ext2fs filesystems.

Table 1-1 CHASE-BASE CD Contents

Name	RPM	Description
`sysadm_base-lib`	`sysadm_base-lib-1.3.6--1.i386.rpm`	Base libraries used by the server-side portion of Rhino-based applications. When you install it, it will attempt to add its `lib` directory to your `ld.so.conf`.
`cluster_admin`	`cluster_admin-1.0.1--1.i386.rpm`	This RPM contains the cluster database daemon, the reset services daemon and cluster administration daemon. These binaries are installed in `/usr/lib/failsafe/bin`.

Name	RPM	Description
cluster_services	cluster_services-1.0.1--1.i386.rpm	This RPM provides the binaries for the cluster services for CHASE. The binaries are the group communication daemon, the system resources manager daemon, and the cluster memership service daemon. It also provides the agents for basic resource types such as IP address and templates.
failsafe	failsafe-1.0.1--1.i386.rpm	This RPM contains the FailSafe binaries.
sysadm_base-client	sysadm_base-client-1.3.6--1.i386.rpm	This is the base software needed on the client-side by Rhino applications.
sysadm_failsafe-client	sysadm_failsafe-client-0.9.3.i386.rpm	This is the Java-based GUI which you can install on systems which you will use to administer your cluster. (You can use cluster nodes for this if you want.)
sysadm_base-dev	sysadm_base-dev-1.3.6--1.i386.rpm	Headers for building Rhino-based applications.
sysadm_base_tcpmux	sysadm_base_tcpmux-1.3.6--1.i386.rpm	`sysadmd` (the Rhino system administration daemon) is started through `tcpmux`, which is started by `inetd`. This contains the `tcpmux` daemon (really just a wafer-theen layer between `inetd` and `sysadmd`) and a configuration file. When you install it, it will attempt to add a `tcpmux` entry to your `inetd.conf`.
sysadm_base-server	sysadm_base-server-1.3.6--1.i386.rpm	This contains the `sysadmd` daemon, service and protocol modules, and commands for administering privileges. When you install it, it will attempt to add an entry to your `tcpmux.conf` file (installed by `sysadm_base-tcpmux`).

Name	RPM	Description
`sysadm_failsafe-server`	`sysadm_failsafe-server-0.9--3.i386.rpm`	These are the server-side GUI libraries which need to be installed on any node in the cluster which you are going to connect the client to. (You only need to install it on one node in the cluster.)
`sysadm_failsafe-web`	`sysadm_failsafe-web-0.9--3.i386.rpm`	These are HTML files for serving the FailSafe GUI as an applet. Like `sysadm_failsafe-server`, this should be installed on any node in the cluster which you want to connect the GUI to, but you only need this if you do not want to install `sysadm_failsafe-client` on any machines. (This has not really been tested!)
`IBMJava118--JRE`	`IBMJava118--JRE--1.1.8--3.0.i386.rpm`	This is the Java environment RPM. The GUI is qualified to run over this environment.
`Filesystem-plugin`	`ext2fs-plugin-1.0--1.i386.rpm`	This is another basic resource like the IP address. It provides agents to make a filesystem highly available.
	`pcp-2.1.6-1.i386.rpm`	This binary RPM contains the base PCP framework and utilities.
	`pcp-pro-2.1.6-7.i386.rpm`	This binary RPM contains the PCP visualization tools, additional PCP agents and logging tools.
	`pcp-fsafe-2.1.1-1.i386.rpm`	This binary RPM contains the PCP for FailSafe agent and visualization tools for PCP.

CHASE-FILESERVER CD

The CHASE-FILESERVER CD consists of the NFS server, the Samba server, the NFS plug-in, and the Samba plug-in. These plug-ins depend on the IP address and the filesystem plug-ins which come along with the CHASE-BASE CD.

Table 1-2 CHASE-FILESERVER CD Contents

Name	RPM	Description
`samba-2.0.7`		This RPM contains the Samba server version 2.0.7.
`nfs-utils`	`nfs-utils-0.1.6--2`	This RPM contains the NFS server for Red Hat.
`knfsd`	`knfsd-991001-47.i386.rpm`	This RPM contains the NFS server for SuSE.
`NFS-plugin`	`NFS-plugin-1.0--1.i386.rpm`	This plug-in contains the agent used by CHASE to make the NFS daemon highly available.
`Samba-plugin`		This plug-in contains the agent used by CHASE to make the Samba processes highly available.

CHASE-WEBSERVER CD

The CHASE-WEBSERVER CD consists of the Apache Web server and the Apache plug-in. The Apache plug-in depends on the IP address and the filesystem plug-ins which come along with the CHASE-BASE CD.

Table 1-3 CHASE-WEBSERVER CD Contents

Name	RPM	Description
`apache-1.3`	`apache-1.3.12-2.i386.rpm`	This RPM contains the Apache Web server. This Web server can be made highly available. The package is distribution sensitive and hence there are two of them, one for SuSE and the other for Red Hat.
`Apache-plugin`	`Apache-plugin-1.0-1.i386.rpm`	This plug-in contains the agent used by CHASE to make the Apache Web server highly available.

Chapter 2

Hardware Components of a Linux FailSafe Cluster

Linux FailSafe Configurations, Topologies, Reset Model, and Storage Models

Linux FailSafe is expanded to include configurations with up to four nodes. This section explains Linux FailSafe configuration, topologies, and network connections. This section consists of these subsections:

- "Linux FailSafe Configurations: Reset and Failover Models"
- "Configuration Types"
- "Linux FailSafe Networks"

Linux FailSafe Configurations: Reset and Failover Models

This subsection consists of the following:

- "Failover Models"
- "Reset Model"
- "Configuring for Reset and Failover"

Failover Models

The Linux FailSafe software uses heartbeats on an Ethernet network to determine that a node has failed. (This network is discussed in "Heartbeat Network and Ethernet Hub", page 13) If a cluster node fails, the failover model determines how the applications (resources) on the failed node are handled so that they remain available to users:

- Star: failover is to a backup node.
- Hub/switched: failover is to the other node.

Multiple heartbeat networks are recommended.

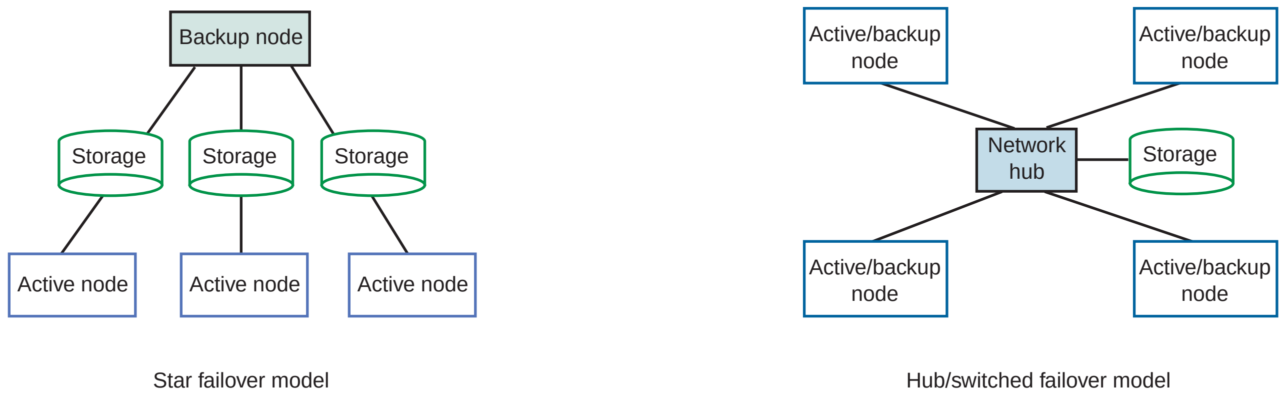

Figure 2-1 Failover Models

Linux FailSafe software is used to determine the node to which the resource fails over, as explained in the *Linux FailSafe Administrator's Guide*.

Reset Model

Linux FailSafe uses serial reset lines to reboot a failed node in the cluster:

- Ring reset: the system control port of each; a node is directly connected to a tty port on another node.

  1. In a two-node system, the system control port on each node is connected to a tty port on the other node.

  2. In a ring system with more than two nodes (up to four nodes), a tty port on each node is cabled to the system control port on the adjacent node; in this model, reset is unidirectional.

Configuring for Reset and Failover

A Linux FailSafe configuration can use either type of failover model. Table 2-1 summarizes Linux FailSafe failover models, reset model, and connections.

Table 2-1 Failover Models, Reset Model, and Connections

Failover	Reset	Nodes	Reset Connections
Star	Ring	2 to 4	Direct link (tty to system control port) between nodes
Hub/switched	Ring	2 to 4	Direct link (tty to system control port) between nodes, unidirectional

Configuration Types

This subsection describes specific Linux FailSafe configurations, as follows:

- "Two-Node Configuration"
- "Four-Node Configuration: Ring Reset and Hub/Switched Failover"

In each section, diagrams show the pertinent networks and connections.

Two-Node Configuration

Figure 2-2 diagrams a Linux FailSafe two-node configuration.

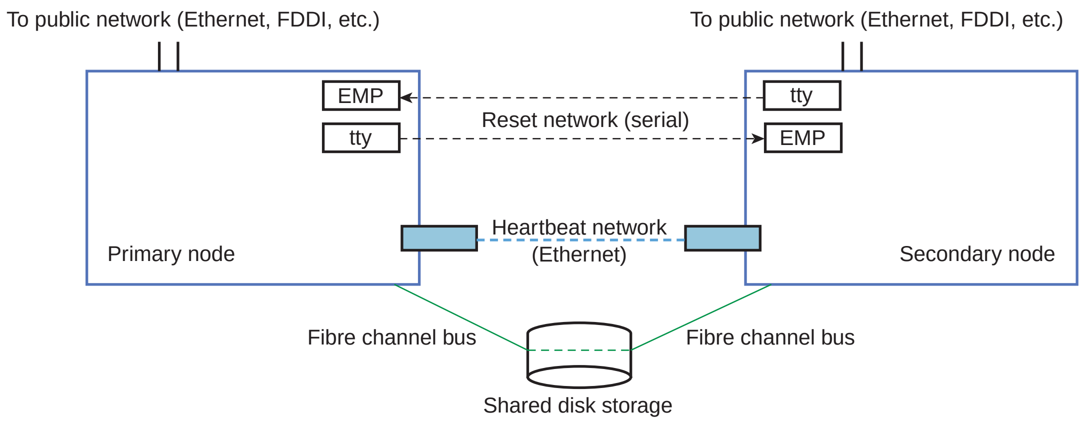

Figure 2-2 Two-Node Configuration, Node-to-Node Reset

In the configuration in Figure 2-2, the two nodes listen to each other's heartbeat on the heartbeat network, which must be Ethernet. Each can power-cycle the other on the reset network's two serial lines.

Four-Node Configuration: Ring Reset and Hub/Switched Failover

Figure 2-3 diagrams a Linux FailSafe four-node ring configuration. This configuration uses hub/switched failover and ring reset.

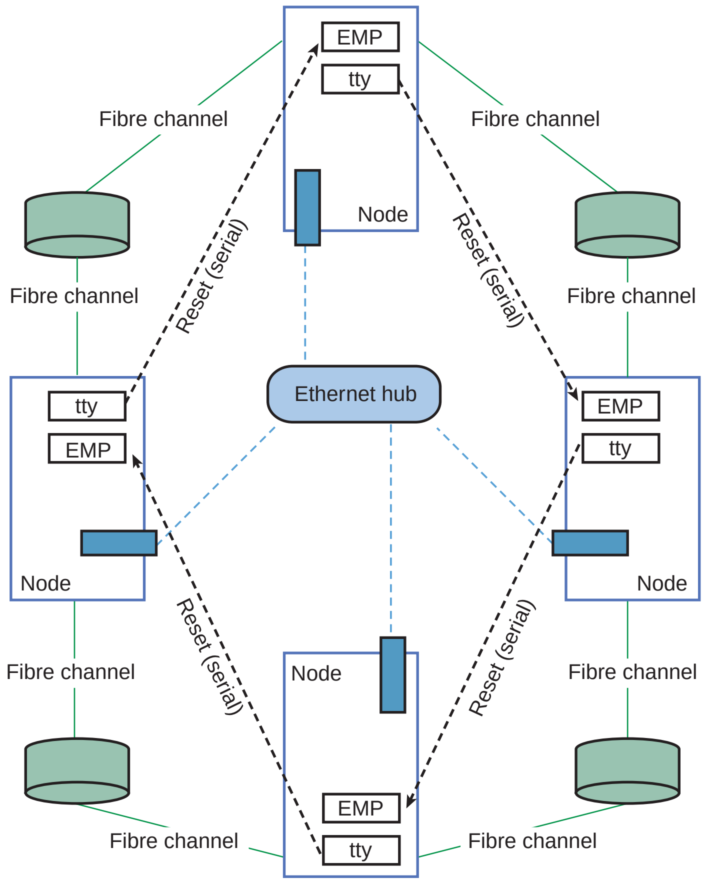

Figure 2-3 Four-Node Configuration: Ring Reset and Hub/Switched Failover

Heartbeat is via an Ethernet connection from the nodes to the Ethernet hub

Linux FailSafe Networks

The Linux FailSafe cluster has these networks:

- Heartbeat network (Ethernet) connection between nodes: the keep-alive heartbeat for monitoring node status and Linux FailSafe control messages.

  Heartbeat is via an Ethernet connection directly between the nodes, or from the nodes to an Ethernet hub.

- Reset network (serial) for power cycling a node: if a node fails, this serial connection provides for reset. This network is also referred to as the control network.

  Depending on the configuration, the failed node is power cycled by a surviving node in the cluster.

- Shared fibre channel connection to storage

  Fibre Channel Hub devices may be part of a Fibre Channel RAID or JBOD connection.

- Public (Ethernet) network interface(s) connecting the cluster node to clients and the outside world

  If a node has multiple public network interfaces, they must be on different subnets.

This subsection discusses these networks (except the public network) further in the following:

- "Heartbeat Network and Ethernet Hub"
- "Reset Network"

Heartbeat Network and Ethernet Hub

The heartbeat network is the keep-alive heartbeat for monitoring node status and Linux FailSafe control messages. This network is an Ethernet connection directly between the nodes, or from the nodes to the Ethernet hub (heartbeat hub).

When the failure of a node is detected on the heartbeat network, one of the other nodes in the cluster uses the serial reset network to power-cycle the failed node. The storage of the failed node is taken over by another node in the cluster.

For the connection to the Ethernet hub or network, RJ45-RJ45 null modem cables (9290131) are included with the option. This cable is 20 feet long; the customer might have ordered the 40-foot optional cable (9290132).

Reset Network

When the heartbeat network carries information that a node has failed, the failed node is power-cycled by a surviving node in the cluster.

The reset network consists of serial connections to each node's system control port.

After a successful reset, one of the surviving nodes in the cluster takes over the resources owned by the failed node. The node that resets the failed node need not be in the cluster.

Determining failover — directing a particular node to take over the function of the failed node and its access to storage — is not necessarily done by the same entity that does the reset.

Cabling the Ethernet Networks

This section explains how to set up the nodes in the Linux FailSafe cluster. It discusses software installation and interface board installation. It explains how to cable the heartbeat Ethernet connection for Linux FailSafe, as well as the public Ethernet connection.

This section consists of these subsections:

- "Cabling the Heartbeat Ethernet Network"
- "Cabling the Public Network"
- "Testing the Public Network Interface"
- "Configuring and Testing Heartbeat Network Connectivity"

Note: Before installing a Linux FailSafe system, make sure that the installation site meets the operating limits and AC power requirements.

The following equipment is required for installation:

- Laptop or ASCII terminal
- Phillips and small flat-blade screwdrivers
- Installation guides for the component systems (see “About This Guide”)

Cabling the Heartbeat Ethernet Network

The network between the cluster nodes supplies the heartbeat of each node to other nodes or equipment monitoring system status, as well as other Linux FailSafe information.

Figure 2-4 shows the Ethernet connector on the SGI 1200 panel. The server might also have one or more ENET boards.

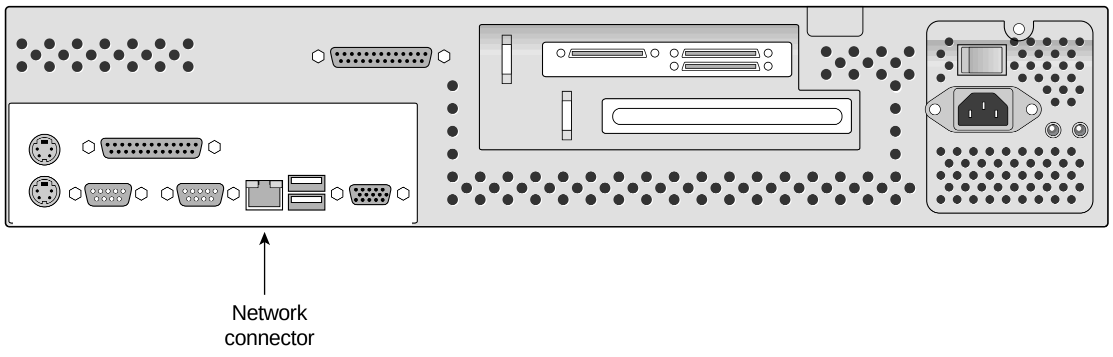

Figure 2-4 SGI 1200 Deskside Server Ethernet Ports (Rear of Chassis)

To cable the heartbeat network, attach an end of the null modem Ethernet cable supplied with the Linux FailSafe system (part number 018-0700-001) to an Ethernet port on each host module. Depending on the configuration, attach the other end to one of the following:

- Two-node configuration: the other node’s Ethernet port
- Other configurations: Ethernet hub

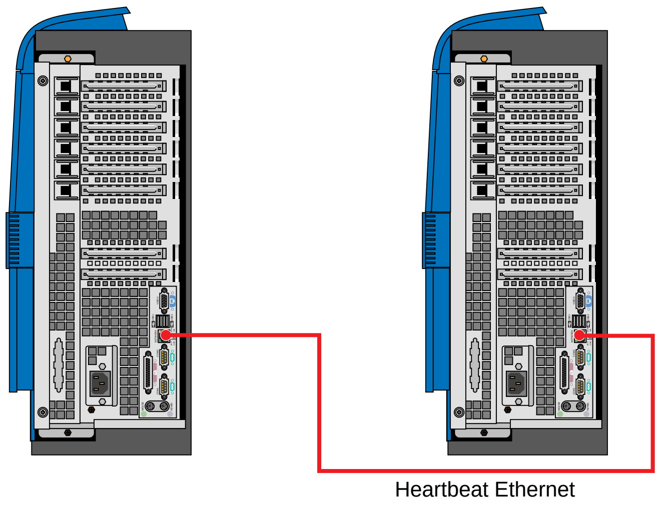

Figure 2-5 Two SGI 1450s Connected Back-to-Back

Figure 2-6 shows cabling for an Ethernet hub.

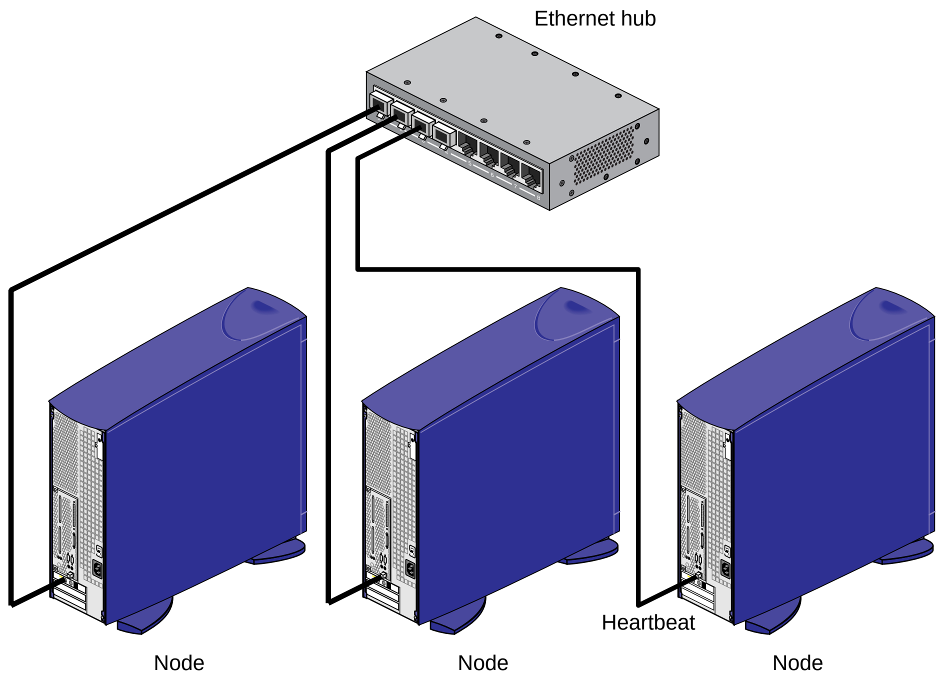

Figure 2-6 Cabling the Ethernet Hub for the Heartbeat Network

Cabling the Public Network

On each node, connect the public network drop cable to an Ethernet port. If the configuration uses an Ethernet hub for the public network, cable the nodes to the hub and attach the public network drop cable to the hub.

Cables for the public Ethernet network are not included in the Linux FailSafe marketing codes. The part number for this cable is 9290131.

Testing the Public Network Interface

For each public network on each node in the cluster, enter

```
# /usr/etc/ping nodeIPaddress
```

where *nodeIPaddress* is the IP address of the node. Typical `ping` output should appear, such as

```
PING IPaddress
64 bytes from 190.x.x.x: icmp_seq=0 tt1=254 time=3 ms
64 bytes from 190.x.x.x: icmp_seq=1 tt1=254 time=2 ms
64 bytes from 190.x.x.x: icmp_seq=2 tt1=254 time=2 ms
```

If `ping` fails, follow these steps:

1. Verify that the network interface was configured up using `ifconfig`; for example:

   ```
   # /usr/etc/ifconfig eth0
   eth0 Link encap:Ethernet HWaddr 00:C0:4F:58:6E:B9
   inet addr: 190.x.x.x Bcast:190.x.x.x Mask:255.255.255.0
   UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
   RX packets:1254523 errors:0 dropped:0 overruns:0 frame:0
   TX packets:1565980 errors:0 dropped:0 overruns:0 carrier:0
   collisions:0 txqueuelen:100
   Interrupt:11 Base address:0xdc00
   ```

   The `UP` in the output indicates that the interface was configured up.

2. Verify that the cables are correctly seated.

Repeat this procedure on each node.

Configuring and Testing Heartbeat Network Connectivity

To configure the heartbeat network interface, follow instructions on network interface and IP address configuration in the *Linux FailSafe Administrator's Guide.* Configure the interface on each node in the cluster.

To test connectivity for a cluster, use `ping` as described in "Testing the Public Network Interface", page 17.

Cabling the Reset Network (Serial Connection)

If a node fails, a serial connection enables another node to power-cycle it. As of now, the serial connection has to be direct.

After you set up the heartbeat and public Ethernet connections for the nodes, as explained in "Cabling the Ethernet Networks", cable the serial reset network. This section explains cabling this reset network for various configurations:

- "Ports for the Serial Connection"
- "Cabling the Serial Connection: Ring Reset"

Ports for the Serial Connection

The serial connection can be done one way:

- Ring reset: system control (EMP) port on each node to tty port on another node

Figure 2-7 and Figure 2-8 show the tty serial ports on the SGI 1200 and 1450.

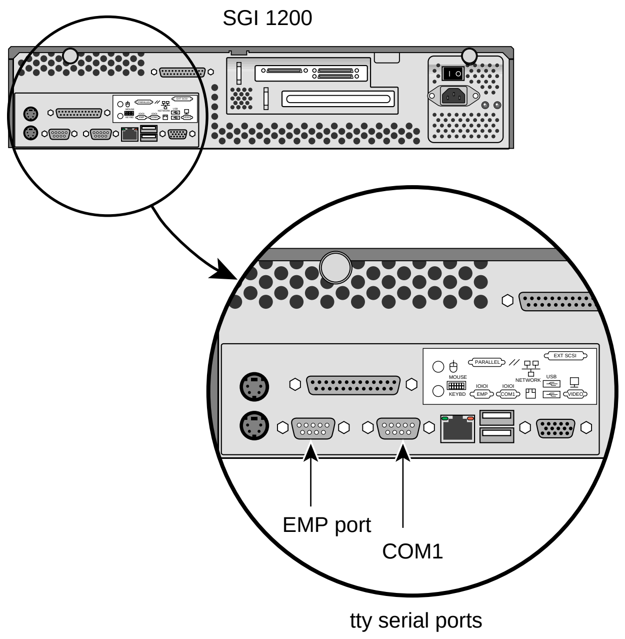

Figure 2-7 SGI 1200 Server tty Ports

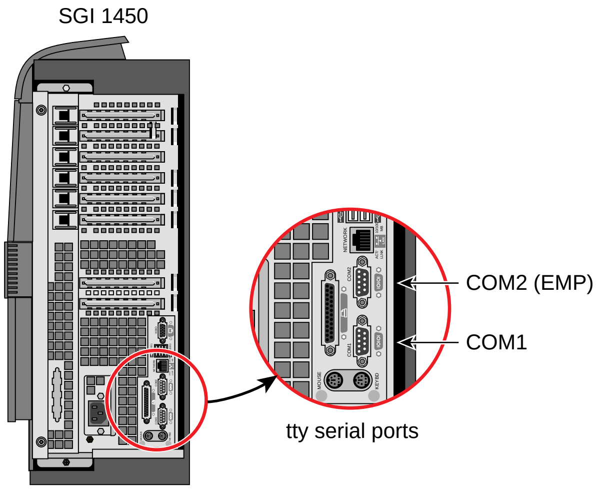

Figure 2-8 SGI 1450 Server tty Ports

Cabling the Serial Connection: Ring Reset

For ring reset, follow these steps to cable the serial connection:

1. For an SGI 1200 or 1450, make sure that nothing is connected to the serial port.
2. Make sure that the nodes are arranged so that the serial cables included in the kits will reach the EMP ports and the tty ports.
3. Cable the serial connection.

   For a two-node cluster, cable each node's EMP port to the other node's tty port.

   For a four-node cluster:

   - Connect the EMP port on the first node to a tty port on the second node.
   - Connect the EMP port on the second node to a tty port on the third node.
   - Connect the EMP port on the third node to a tty port on the fourth node.
   - Connect the EMP port on the fourth node to a tty port on the first node.

Figure 2-9 shows the serial connection between two SGI 1200 servers in a two-node configuration.

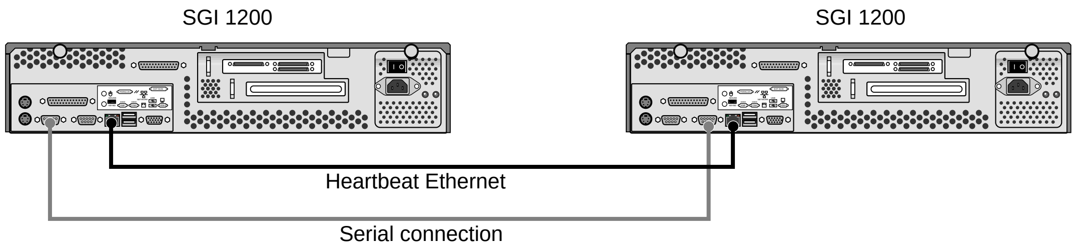

Figure 2-9 Serial Connection and Heartbeat Ethernet Connection, SGI 1200 Servers, Two-Node Configuration

Figure 2-10 shows the serial connection between two SGI 1450 servers in a two-node configuration.

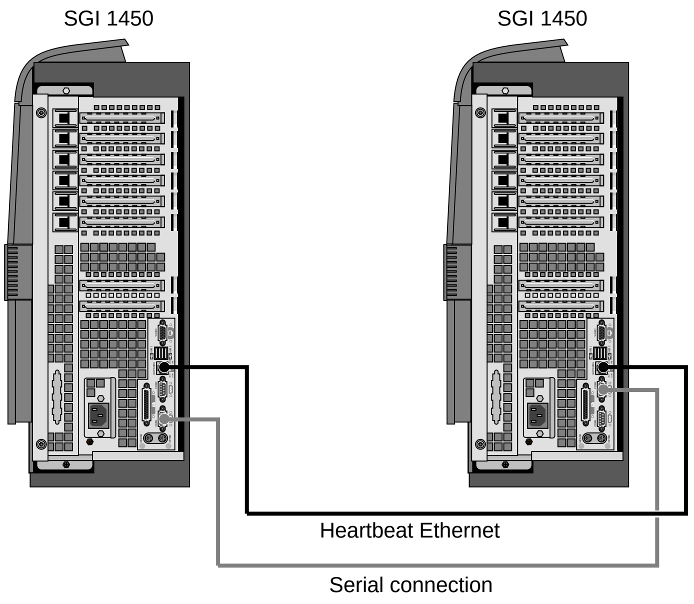

Figure 2-10 Serial Connection and Heartbeat Ethernet Connection, SGI 1450 Servers, Two-Node Configuration

Figure 2-11 shows the serial connection between an SGI 1450 server and an SGI 1200 deskside server in a two-node configuration.

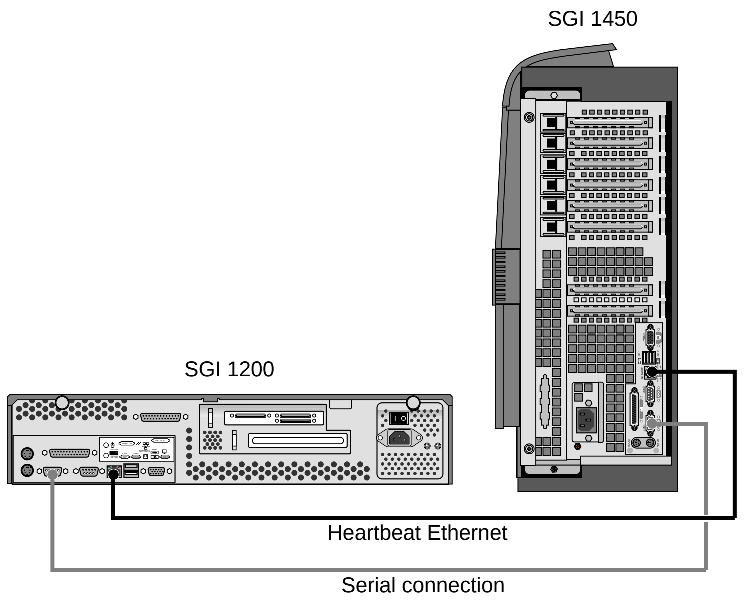

Figure 2-11 Serial Connection and Heartbeat Ethernet Connection, SGI 1200 and SGI 1450 Servers, Two-Node Configuration

Figure 2-12 diagrams a Linux FailSafe four-node cluster with ring reset.

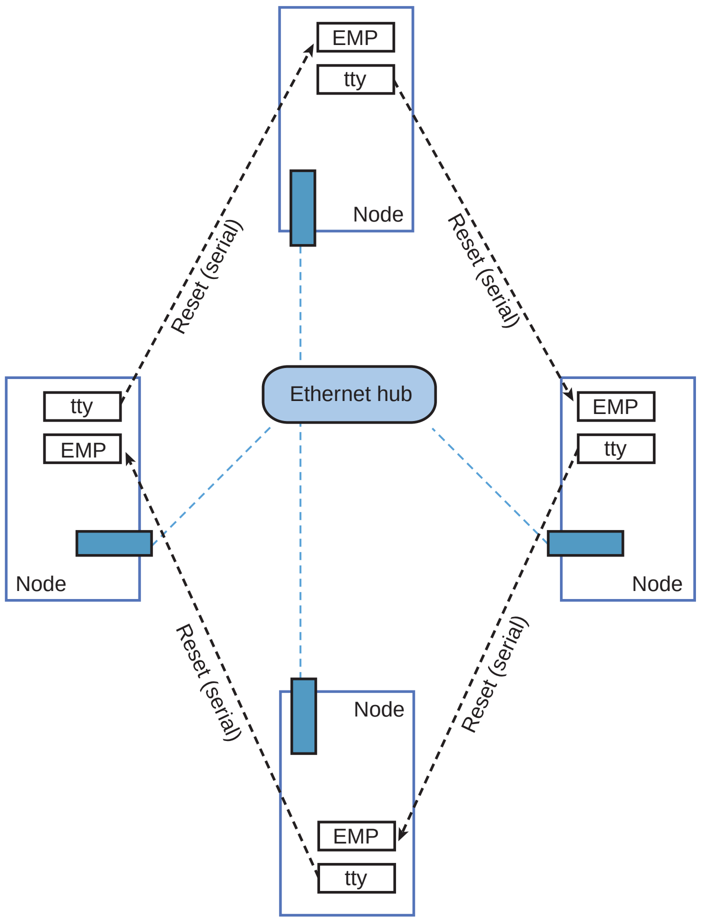

Figure 2-12 Four-Node Cluster Ring Reset

Testing the Serial Connection

The cluster manager has two mechanisms for checking the serial interface.

- For a node using the cluster reset services daemon, use `admin ping node` *nodename*; for example,

```
cmgr> admin ping node fs0
```

- For a node that is not using the cluster reset services daemon, use `admin ping standalone node` *nodename*; for example,

  ```
  cmgr> admin ping standalone node fs0
  ```

Testing the Serial Connection for a Node in a Cluster

To test the serial connection for an individual server, use `admin ping node` *nodename*.

The test exits when it finds the first error and pipes an error message to standard output. Error messages include the name of the server that fails to respond.

Follow these steps:

1. Make sure the cluster nodes are powered on. Power on the serial multiplexer.
2. On a cluster node, start the cluster manager:

   ```
   cmgr
   ```

3. If necessary, stop Linux FailSafe on each node:

   ```
   cmgr> stop ha_services for cluster clustername
   ```

   where *clustername* is the cluster you are testing. Wait until the node has successfully transitioned to standby state and the FailSafe processes have exited. This process can take a few minutes.

4. Enter the following:

   ```
   cmgr> admin ping nodename
   ```

   For example:

   ```
   cmgr> admin ping sys2
   ```

   Sample output follows:

   ```
   ping operation successful
   ```

5. If the command fails, make sure all the cables are seated properly and rerun the command.

Testing the Serial Connection for a Standalone Node

For a server that is not using the cluster reset services daemon, use `admin ping standalone node` *nodename*; for example:

```
cmgr> admin ping standalone node nodename
```

Sample failure output follows:

```
Internal error : crad is running, cannot perform system controller
operation in the standalone mode.

Failed to admin ping

admin command failed
```

Sample successful output follows:

```
ping operation successful
```

Installing and Configuring VACM

The following are the basic steps to enable `crsd` to use VACM for reset, and use cross-over cables for connecting the reset. These steps have to be done on each 1450 node.

1. Bring the machine down to the BIOS.

   a. Ensure that the EMP port is enabled.

   b. If you plan to use the console serial port to control the EMP port, ensure that console redirection is disabled. Failure to do so will prevent booting without disconnecting the serial cable.

   For BIOS details, see Chapter 3, *Configuring Software and Utilities* of the *SGI 1450 Server User's Guide*.

   SGI Linux servers ship configured with EMP on and console redirection on.

2. Install the following VACM RPMs:

   ```
   vacm-lib-2.0.0beta
   vacm-vash-2.0.0beta
   vacm-2.0.0beta
   ```

3. Configure nexxus, the VACM daemon, using `vash` on the controlling node. It is assumed that nexxus is already running, the BIOS is configured correctly, and the serial cable is in place.

   ```
   hostname / nodename / ip address
   host1    node1    150.166.8.57    (Controlling Node)
   host2    node2    150.166.8.58    (Controlled Node)
   ```

   a. Start nexxus the first time (must be `root`).

      ```
      host1# nexxus &
      ```

   b. Login to localhost (can be any user) as default nexxus user `blum` and password `frub`.

      ```
      host1# vash -c localhost -u blum -p frub
        NEXXUS_READY
      vash$
      ```

   c. If you wish to change the login name from `blum` to `barf`, and change the password from `frub` to `foobar`.

      ```
      vash$ ipc localhost nexxus:admin_rename:blum:barf
        NEXXUS:1:JOB_STARTED
        NEXXUS:1:JOB_COMPLETED
      vash$ ipc localhost nexxus:admin_chg_passwd:frub:foobar
        NEXXUS:1:JOB_STARTED
        NEXXUS:1:JOB_COMPLETED
      ```

   d. Enable remote machine(s) to run `vash`/`hoover` commands (by `IP_address:netmask_bits`).

      ```
      vash$ nexxus:admin_add_addr_acl_rule:blum:allow:150.166.8.0:16
        NEXXUS:1:JOB_STARTED
        NEXXUS:1:JOB_COMPLETED
      ```

   e. Add the group failsafe and make `blum` an admin for it.

      ```
      vash$ ipc localhost nexxus:group_add:failsafe
        NEXXUS:3:JOB_STARTED
        NEXXUS:3:JOB_COMPLETED
      vash$ ipc localhost nexxus:group_add_admin:failsafe:blum
        NEXXUS:4:JOB_STARTED
        NEXXUS:4:JOB_COMPLETED
      ```

f. Add the controlled node (host2/node2) to the group failsafe.

```
vash$ ipc localhost nexxus:node_add:node2:failsafe
  NEXXUS:6:JOB_STARTED
  NEXXUS:6:JOB_COMPLETED
```

g. Enable `blum` to run `emp` commands.

```
vash$ ipc localhost nexxus:admin_add_mod_acl_rule:blum:allow:emp:*
  NEXXUS:2:JOB_STARTED
  NEXXUS:2:JOB_COMPLETED
```

h. Add the host node2 to the emp (the IPMI interface). There is no EMO password on node node2 (“NONO”).

```
vash$ ipc akash emp:configuration:node2:/dev/tts/0:NONE
  NEXXUS:5:JOB_STARTED
  NEXXUS:5:JOB_COMPLETED
```

i. Now you can verify the IMPI status (this is what the `crsd ping` command maps to).

```
vash$ ipc localhost emp:node_status:node2
  EMP:7:JOB_STARTED
  EMP:7:NOSTATUS:/dev/tts/0:DETECTED
  EMP:7:JOB_COMPLETED
```

j. Exit vash.

```
vash$ exit
host1#
```

k. To clean up a botched config:

```
root# killall nexxus
root# > /usr/lib/vacm/vacm_configuration
root# nexxus &
```

l. And then start from scratch.

4. After the rest of FailSsafe configuration and setup is completed, configure FailSafe for host1 (node1) to reset host2 (node2).

```
cmgr> show node node2
Logical Machine Name: node2
Hostname: node2
```

```
Is FailSafe: true
Nodeid: 2
Reset type: powerCycle
System Controller: vacm
System Controller status: enabled
System Controller owner: BackupNode
System Controller owner device: host1,blum,frub
System Controller owner type: tty
...
```

Chapter 3

CHASE Software Installation

The CHASE software installation is achieved by installing CHASE-BASE CD and then the other CD. During installation, some RPMs have to be installed before the others since there are some dependencies among them. These dependencies are described below.

The `sysadm_base-lib` RPM has to be installed first. It provides libraries which are used by all other RPMs. The the basic CHASE RPMs `cluster_admin`, `cluster_services`, and `failsafe` are to be installed. Following that, the GUI RPMs (`sysadm_*`) can be installed. Now, depending on which of the other two CDs are to be used, the packages and then the agents are installed. Example: For the CHASE-WEBSERVER CD, we install the `apache-1.3.rpm` for the particular Linux distribution and then the agent.

`INSTALL` Script

The installation described above is carried out by an `INSTALL` executable, that can be invoked by the user. The CHASE-BASE CD `INSTALL` script finishes with the installation from the CHASE-BASE CD and then waits for the user to input the **second** CD, where upon it calls the `INSTALL` from the other CD. `INSTALL` from the second CD can be run independently too; however the RPMs from the first CD need to be installed beforehand. Refer to the above paragraph on dependencies among the RPMs. After the second CD `INSTALL` script finishes installation, it tries to create a new cluster database and delete any existing ones.

Example 3-1 Sample Output from the `INSTALL` Script

```
CASE: Some RPMs are already installed.

#>./INSTALL

        ---- Begin CHASE Installation ----

  Checking for file /sbin/portmap ...
  Checking for package sysadm_base-lib ....
  Checking for package cluster_admin ....
  Checking for package cluster_services ....
  Checking for package failsafe ....
  Checking for package sysadm_base-tcpmux ....
  Checking for package sysadm_base-server ....
  Checking for package sysadm_base-client ....
  Checking for package sysadm_failsafe-client ....
  Checking for package sysadm_failsafe-server ....
  Checking for package IBMJava118-JRE ....

         Packages  sysadm_base-lib cluster_admin cluster_services
failsafe sysadm_base-tcpmux sysadm_base-server sysadm_base-client
sysadm_failsafe-client sysadm_failsafe-server IBMJava118-JRE already installed


        ---- Installing RPMs...

package sysadm_base-lib-1.3.6-1 is already installed
package cluster_admin-1.0.1-1 is already installed
package cluster_services-1.0.1-1 is already installed
package failsafe-1.0.1-1 is already installed
package sysadm_base-tcpmux-1.3.6-1 is already installed
package sysadm_base-server-1.3.6-1 is already installed
package sysadm_base-client-1.3.6-1 is already installed
package sysadm_failsafe-client-0.9-3 is already installed
package sysadm_failsafe-server-0.9-3 is already installed
package IBMJava118-JRE-1.1.8-3.0 is already installed

        ---- Checking packages installation...

  Checking for package sysadm_base-lib ....
  Checking for package cluster_admin ....
  Checking for package cluster_services ....
  Checking for package failsafe ....
  Checking for package sysadm_base-tcpmux ....
  Checking for package sysadm_base-server ....
  Checking for package sysadm_base-client ....
  Checking for package sysadm_failsafe-client ....
  Checking for package sysadm_failsafe-server ....
  Checking for package IBMJava118-JRE ....

        Do You want to install documentation on Failsafe (Y)/N ?
```

n

```
        ---- Updating /etc/services file...


        /etc/services update okay


        ---- Updating startup flags...


        Startup flags update okay


        ---- Creating links in init level directories...

  Creating link /etc/rc.d/rc3.d/S36fs_cluster
  Creating link /etc/rc.d/rc3.d/K39fs_cluster
  Creating link /etc/rc.d/rc3.d/S37failsafe
  Creating link /etc/rc.d/rc3.d/K37failsafe
  Creating link /etc/rc.d/rc5.d/S36fs_cluster
  Creating link /etc/rc.d/rc5.d/K39fs_cluster
  Creating link /etc/rc.d/rc5.d/S37failsafe
  Creating link /etc/rc.d/rc5.d/K37failsafe

        Links creation okay


        Insert 2nd CD now and enter when done ....

        ---- Begin CD 2 Installation ----

  Checking for package perl ....
  Checking for package sysadm_base-lib ....
  Checking for package cluster_admin ....
  Checking for package cluster_services ....
  Checking for package failsafe ....
  Checking for package apache ....

         Packages  apache already installed
```

```
WARNING: Some of the packages are of a different version.

        The Packages are - apache

         Overwrite them (Y)/N ?

n

WARNING: Proper working of software cannot be guaranteed with
                     older version of packages.
  Checking for package Apache-plugin ....

        ---- Installing RPMs...


        ---- Installing RPMs...

Apache-plugin                ############################################
Adding Apache resource type to CDB under cluster space

        ---- Checking installation...

  Checking for package Apache-plugin ....


        ---- Creating an empty database...

Assuming /var/lib/failsafe/cdb/cdb.db as the database
Preparing to delete database at /var/lib/failsafe/cdb/cdb.db! Continue [y]/n

n

Response was not yes, exiting...

        ---- CHASE Installation DONE ----
```

A Brief Description of `INSTALL` Script

Note: 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 the `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.

Brief Description of Second CD `INSTALL` Script

Chase supports two variants: CHASE-Web and CHASE-FS. In either case, the package/application and agents for each have to be installed.

For each distribution, currently only SuSE and Red Hat, and relevant release, find the package in the second CD. Install it using the RPM directive `rpm -i`.

For example, the CHASE-Web CD has the package `apache.*.rpm` in the directory `/dev/cdrom/sgi/packages`. Install it with the command `rpm -i apache.*.rpm`.

For the CHASE-FS CD, the packages are `samba` and `nfs`.

These packages may already exist on the server and may be of a different version. In that case, the older packages may have to be removed and the newer packages from the CHASE CD installed.

Installing Agents

For each CD, we have a list of agents in the directory `/dev/cdrom/sgi/RPMS`. Install them using the RPM command `rpm -i`. These agents are meant for the packages mentioned above.

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 above using the CHASE CD's `INSTALL` script.
2. Configure the system files on each node, as described in "Configuring System Files".
3. Configure the network interfaces on the nodes using the procedure in "Configuring Network Interfaces".
4. 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* of the *Linux FailSafe Administrator's Guide*.

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`

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
```

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.

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 filename.

`-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.

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
```

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 is used in the procedure.

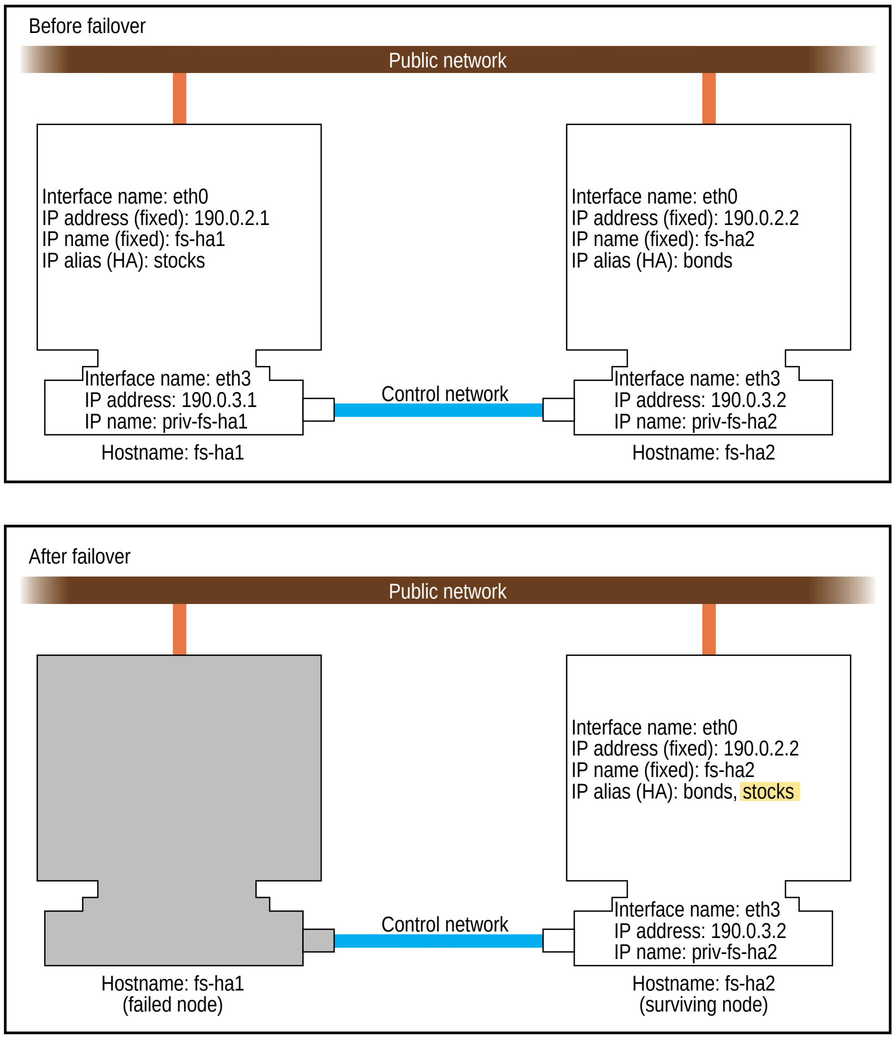

Figure 3-1 Example Interface Configuration

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

   Note: 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 fs-ha1.company.com fs-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, page 42, 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 `fs-ha1` and `fs-ha2`, in `/etc/aliases` on `fs-ha1`, add

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

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

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

Note: 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.

Configuring FailSafe

Configuring FailSafe involves the following steps:

- Defining nodes, as described in Section 5.4.1, *Defining Cluster Nodes* of the *Linux FailSafe Administrator's Guide*.
- Creating a cluster, as described in Section 5.4.5, *Defining a Cluster* of the *Linux FailSafe Administrator's Guide*.
- Creating Failover policies, as described in Section 5.5.12, *Defining a Failover Policy* of the *Linux FailSafe Administrator's Guide*.
- Creating resources, as described in Section 5.5.1, *Defining Resources* of the *Linux FailSafe Administrator's Guide*.
- Creating resource groups, as described in Section 5.5.15, *Defining Resource Groups* of the *Linux FailSafe Administrator's Guide*.
- Starting HA services in the clusters, as described in Section 7.3, *Activating (Starting) Linux FailSafe* of the *Linux FailSafe Administrator's Guide*.

There are three additional resource types present in CHASE plug-in CDs: NFS, Samba, and Apache.

NFS

The NFS resource type is part of the CHASE-FILESERVER CD.

Adding NFS Resource Information to the Configuration Database (CDB)

This section describes procedures that show you how to create an NFS resource type, resource, resource group, and how to test the NFS resource. These procedures assume that a CDB that does not include NFS has already been created, installed, and tested as described in the *Linux FailSafe Administrator's Guide*.

Creating an NFS Resource Type

To create an NFS resource type, this subsection assumes that you are already familiar with the concepts of resource types. The NFS resource type defines the following resource attributes.

- *resource-name*, which defines the name of the resource and is also the export disk name used as input to the `exportfs`(1M) command
- *export-info*, which lists the export options for the file system used in the `exportfs`(1M) command
- *filesystem*, which is the name of the file system that is used as input to the `mount`(1M) command

The NFS resource type is created at cluster creation time. If this automatic resource creation fails, the administrator must create the resource type before an NFS resource is created. The NFS resource type must be installed if you want to add an NFS resource to a cluster that was created before the NFS software was installed.

Creating the NFS Resource Type

You can use one of the following methods to create the NFS resource type:

- Run cluster manager (`cmgr`) and manually create the resource type. For more information, see the *Linux FailSafe Administrator's Guide*.
- Run cluster manager (`cmgr`) and install the resource type, as follws:

```
cmgr> show resource_types installed

template
Netscape_web
statd
Oracle_DB
MAC_address
IP_address
INFORMIX_DB
filesystem
volume

cmgr> install resource_type NFS in cluster eagan

cmgr>
```

- Use the template scripts supplied with Linux FailSafe located in `/usr/lib/failsafe/cmgr-create-resource_type`.
- Execute `/usr/lib/failsafe/resource_types/NFS/create_resource_type` and include the path of the CDB argument and the cluster name.

  OR

  Use the **Load Resource Type** GUI task to load the resource type.

Creating an NFS Resource

After you have defined the resource type, the administrator must define the NFS resources based on the resource type. Each resource requires a unique resource name (for example, the NFS resource type is the NFS instance name). Then, the administrator must supply the resource parameters. To create the resource, either use the cluster manager (`cmgr`), the `cmgr-create-resource-NFS` scripts, or the GUI.

Example 4-1 Creating an NFS Resource using `cmgr`

```
cm2> /usr/lib/failsafe/bin/cluster_mgr
Welcome to SGI Cluster Manager Command-Line Interface

cmgr> define resource /disk5 of resource_type NFS in cluster eagan
Enter commands, when finished enter either "done" or "cancel"

Type specific attributes to create with set command:

Type Specific Attributes - 1: export-info
Type Specific Attributes - 2: filesystem

Resource type dependencies to add:

Resource Dependency Type - 1: filesystem

resource /disk5 ? set export-info to "rw,wsync"
resource /disk5 ? set filesystem to /disk5
resource /disk5 ? done
Successfully created resource /disk5

cmgr> modify resource /disk5 of resource_type NFS in cluster eagan
Enter commands, when finished enter either "done" or "cancel"
```

```
Type specific attributes to modify with set command:

Type Specific Attribute - 1: export-info
Type Specific Attribute - 2: filesystem

No resource type dependencies to add

resource disk5 ? add dependency /disk5 of type filesystem
resource disk5 ? done
Successfully modified resource /disk5

cmgr> show resource /disk5 of resource_type NFS

export-info: rw,wsync,anon=root
filesystem: /disk5

Resource dependencies
filesystem /disk5

cmgr>
```

Creating an NFS Resource Group

To create a resource group, you must first become familiar with the terms and concepts of FailSafe. A resource group can be created either by the GUI or the cluster manager (`cmgr`).

To define an effective resource group, you must include all of the resources that the NFS resource is dependent on, such as file systems, and IP addresses. The following example shows the creation of a typical resource group:

```
cmgr> define resource_group nfsRG in cluster eagan
Enter commands, when finished enter either "done" or "cancel"

resource_group nfs ? set failover_policy to ordered-in-order
resource_group nfs ? add resource /disk5 of resource_type NFS
resource_group nfs ? add resource 128.162.101.20 of resource_type IP_address
resource_group nfs ? add resource /disk5 of resource type filesystem
resource_group nfs ? done
Successfully created resource group nfsRG
```

```
cmgr> show resource_group nfsRG in cluster eagan

Resource Group: nfsRG
        Cluster: eagan
        Failover Policy: ordered-in-order

Resources:
       /disk5 (type: NFS)
       128.162.101.20 (type: IP_Addresses)
       /disk5 (type: filesystem)
```

Samba

The Samba resource type is part of the CHASE-FILESERVER CD.

Adding Samba Resource Information to the Configuration Database (CDB)

This section describes procedures that show you how to create a Samba resource type, resource, resource group, and how to test the Samba resource. These procedures assume that a CDB that does not include Samba has already been created, installed, and tested as described in the *Linux FailSafe Administrator's Guide.*

Creating a Samba Resource Type

To create a Samba resource type, this subsection assumes that you are already familiar with the concepts of resource types. The Samba resource type defines the following resource attributes.

- *resource-name*, which defines the name of the resource and is also the NETBIOS name of the server
- *monitor-level*, user can specify `1` (checks process existence) or `2` (smbclient queries to the server)

The Samba resource type is created at cluster creation time. If this automatic resource creation fails, the administrator must create the resource type before a Samba resource is created. The Samba resource type must be installed if you want to add a Samba resource to a cluster that was created before the Samba software was installed.

Creating the Samba Resource Type

You can use one of the following methods to create the Samba resource type:

- Run cluster manager (`cmgr`) and manually create the resource type. For more information, see the *Linux FailSafe Administrator's Guide*.
- Run cluster manager (`cmgr`) and install the resource type, as follws:

```
cmgr> show resource_types installed

template
Netscape_web
statd
Oracle_DB
MAC_address
IP_address
INFORMIX_DB
filesystem
volume

cmgr> install resource_type Samba in cluster eagan

cmgr>
```

- Use the template scripts supplied with Linux FailSafe located in `/usr/lib/failsafe/cmgr-create-resource_type`.
- Execute `/usr/lib/failsafe/resource_types/Samba/create_resource_type` and include the path of the CDB argument and the cluster name.

  OR

  Use the **Load Resource Type** GUI task to load the resource type.

Creating a Samba Resource

After you have defined the resource type, the administrator must define the Samba resources based on the resource type. Each resource requires a unique resource name (for example, the Samba resource type is the Samba instance name). Then, the administrator must supply the resource parameters. To create the resource, either use the cluster manager (`cmgr`), the `cmgr-create-resource-Samba` scripts, or the GUI.

Example 4-2 Creating a Samba Resource using `cmgr`

```
cm2> /usr/lib/failsafe/bin/cluster_mgr
Welcome to SGI Cluster Manager Command-Line Interface

cmgr> define resource SAMBA/ of resource_type Samba in cluster eagan
Enter commands, when finished enter either "done" or "cancel"

Type specific attributes to create with set command:

Type Specific Attributes - 1: monitor-level

Resource type dependencies to add:

Resource Dependency Type - 1: IP_address

resource SAMBA/ ? set monitor-level to 2
resource SAMBA/ ? done
Successfully created resource /disk5

cmgr> modify resource SAMBA/ of resource_type Samba in cluster eagan
Enter commands, when finished enter either "done" or "cancel"

Type specific attributes to modify with set command:

Type Specific Attribute - 1: monitor-level

No resource type dependencies to add

resource SAMBA/ ? add dependency 128.162.101.22 of type IP_address
resource SAMBA/ ? done
Successfully modified resource SAMBA/

cmgr> show resource SAMBA/ of resource_type Samba

monitor-level:2

Resource dependencies
IP_address 128.162.101.22

cmgr>
```

Creating a Samba Resource Group

To create a resource group, you must first become familiar with the terms and concepts of FailSafe. A resource group can be created either by the GUI or the cluster manager (`cmgr`).

To define an effective resource group, you must include all of the resources that the Samba resource is dependent on, such as file systems, and IP addresses. The following example shows the creation of a typical resource group:

```
cmgr> create resource_group sambaRG in cluster eagan
Enter commands, when finished enter either "done" or "cancel"

resource_group samba ? set failover_policy to ordered-in-order
resource_group samba ? add resource samba/ of resource_type Samba
resource_group samba ? add resource 128.162.101.22 of resource_type IP_address
resource_group samba ? done
Successfully created resource group sambaRG

cmgr> show resource_group sambaRG in cluster eagan

Resource Group: sambaRG
        Cluster: eagan
        Failover Policy: ordered-in-order

Resources:
       samba/ (type: Samba)
       128.162.101.22 (type: IP_Addresses)
```

Apache

The Apache resource type is part of the CHASE-WEBSERVER CD.

Adding Apache Resource Information to the Configuration Database (CDB)

This section describes procedures that show you how to create an Apache resource type, resource, resource group, and how to test the Apache resource. These procedures assume that a CDB that does not include Apache has already been created, installed, and tested as described in the *Linux FailSafe Administrator's Guide.*

Creating an Apache Resource Type

To create an Apache resource type, this subsection assumes that you are already familiar with the concepts of resource types. The Apache resource type defines the following resource attributes.

- *resource-name*, which defines the name of the resource
- *admin-scripts*, which is the location of Web servers start and stop scripts. This field is mandatory.
- *port-number*, which is port number to which the Web server listens to for client requests
- *monitor-level*, which is the desired level of Web server monitoring. User can specify either `1` (checks for existence of the Web server process) or `2` (checks by issuing an HTML request to the Web server)
- *web-ipaddr*, which is the Web server IP address. This is the IP address to which Web server listens to. This IP address should also be an `IP_address` resource.

The Apache resource type is created at cluster creation time. If this automatic resource creation fails, the administrator must create the resource type before an Apache resource is created. The Apache resource type must be installed if you want to add an Apache resource to a cluster that was created before the Apache software was installed.

Creating the Apache Resource Type

You can use one of the following methods to create the Apache resource type:

- Run cluster manager (`cmgr`) and manually create the resource type. For more information, see the *Linux FailSafe Administrator's Guide.*
- Run cluster manager (`cmgr`) and install the resource type, as follws:

  ```
  cmgr> show resource_types installed

  template
  Netscape_web
  statd
  Oracle_DB
  MAC_address
  IP_address
  ```

```
INFORMIX_DB
filesystem
volume

cmgr> install resource_type Apache in cluster eagan

cmgr>
```

- Use the template scripts supplied with Linux FailSafe located in `/usr/lib/failsafe/cmgr-create-resource_type`.
- Execute `/usr/lib/failsafe/resource_types/Apache/create_resource_type` and include the path of the CDB argument and the cluster name.

  OR

  Use the **Load Resource Type** GUI task to load the resource type.

Creating an Apache Resource

After you have defined the resource type, the administrator must define the Apache resources based on the resource type. Each resource requires a unique resource name (for example, the Apache resource type is the Apache instance name). Then, the administrator must supply the resource parameters. To create the resource, either use the cluster manager (`cmgr`), the `cmgr-create-resource-Apache` scripts, or the GUI.

Example 4-3 Creating an Apache Resource using `cmgr`

```
cm2> /usr/lib/failsafe/bin/cluster_mgr
Welcome to SGI Cluster Manager Command-Line Interface

cmgr> define resource ha80 of resource_type Apache in cluster eagan
Enter commands, when finished enter either "done" or "cancel"

Type specific attributes to create with set command:

Type Specific Attributes - 1: admin-scripts
Type Specific Attributes - 2: part-number
Type Specific Attributes - 3: monitor-level
Type Specific Attributes - 4: web-ipaddr
```

```
Resource type dependencies to add:

Resource Dependency Type - 1: IP_address

resource ha80 ? set admin-scripts to /usr/apache/ha80
resource ha80 ? set part-number to 80
resource ha80 ? set monitor-level to 2
resource ha80 ? set web-ipaddr to 128.162.101.2
resource ha80 ? done
Successfully created resource ha80

cmgr> modify resource ha80 of resource_type Apache in cluster eagan
Enter commands, when finished enter either "done" or "cancel"

Type specific attributes to modify with set command:

Type Specific Attributes - 1: admin-scripts
Type Specific Attributes - 2: part-number
Type Specific Attributes - 3: monitor-level
Type Specific Attributes - 4: web-ipaddr

No resource type dependencies to add

resource ha80 ? add dependency 128.162.101.22 type IP-address
resource ha80 ?
resource ha80 ? done
Successfully modified resource ha80

cmgr> show resource ha80 of resource_type Apache

admin-scripts: /usr/apache/ha80
part-number: 80
monitor-level: 2
web-ipaddr: 128.162.101.22

Resource dependencies
IP_address 128.162.101.22

cmgr>
```

Creating an Apache Resource Group

To create a resource group, you must first become familiar with the terms and concepts of FailSafe. A resource group can be created either by the GUI or the cluster manager (`cmgr`).

To define an effective resource group, you must include all of the resources that the Apache resource is dependent on, such as file systems, and IP addresses. The following example shows the creation of a typical resource group:

```
cmgr> create resource_group apacheRG in cluster eagan
Enter commands, when finished enter either "done" or "cancel"

resource_group apache ? set failover_policy to ordered-in-order
resource_group apache ? add resource ha80 of resource_type Apache
resource_group apache ? add resource 128.162.101.22 of resource_type IP_address
resource_group apache ? done
Successfully created resource group apacheRG

cmgr> show resource_group apacheRG in cluster eagan

Resource Group: apacheRG
        Cluster: eagan
        Failover Policy: ordered-in-order

Resources:
       ha80 (type: Apache)
       128.162.101.22 (type: IP_Addresses)
```

Performance Co-Pilot for FailSafe

This chapter tells you how to install and use Performance Co-Pilot (PCP) for FailSafe to monitor the availability of a FailSafe cluster.

PCP provides the following:

- An agent for exporting FailSafe heartbeat and resource monitoring statistics to the PCP framework
- 3-D visualization tools for displaying these statistics in an intuitive presentation

The visualization of statistics provides valuable information about the availability of nodes and resources monitored by FailSafe. For example, it can highlight a reduction in monitoring response times that may indicate problems in availability of services provided by the cluster.

Because PCP for FailSafe is an extension to the PCP framework, you can use other PCP tools to analyze or present FailSafe monitoring statistics, and record PCP for FailSafe metrics as archives for deferred analysis. You can also use PCP to gather statistics about CPU and memory utilization, network and disk activity, and other performance metrics for each node in the cluster.

Installing Performance Co-Pilot Software

You can deploy PCP for FailSafe as a collector agent or as a monitor client:

- Collector agents are installed on *collector hosts*, which are the nodes in the FailSafe cluster itself from which you want to gather statistics. Typically, each node in a FailSafe cluster is designated as a collector host.
- A monitor client is installed on the ***monitor host***, which is typically a workstation that has a display and is running an X Window System server and a window manager.

Installing the Collector Host

To install PCP for FailSafe on the designated collector hosts, the following software components must already be installed:

- The RPM package `failsafe-1.0.1--1` or a later version of it
- The RPM package `pcp-2.1.6-1` or a later version of it
- The RPM package `pcp-pro-2.1.6-7` or a later version of it

After this software is installed, you must install the following subsystems of PCP for FailSafe on each collector host. To install the required RPM packages on a collector host, do the following:

1. Locate the binary RPM package `pcp-fsafe-2.1.1-1.i386.rpm` on the FailSafe CD.

2. Log in as `root`.

3. Issue the `rpm`(1) command to install PCP for FailSafe:

   ```
   # rpm -i <srcpath>/pcp-fsafe-2.1.1-1.i386.rpm
   ```

   where <*srcpath*> is the path (on the local file system, CD-ROM, or URL) to the PCP for FailSafe binary RPM package.

   If `pcp-2.1.6-1` and/or `pcp-pro-2.1.6-7` is not installed, you will get an error from `rpm`(1) saying that pre-requisite packages are not installed. You will need to install them before installing `pcp-fsafe-2.1.1-1`.

   Install them by locating the pcp and pcp-pro RPM packages, and installing them the same way, and in the following order:

   ```
   # rpm -i <srcpath>/pcp-2.1.6-1.i386.rpm
   # rpm -i <srcpath>/pcp-pro-2.1.6-7,i386.rpm
   ```

4. Change to the `/var/pcp/pmdas/fsafe` directory:

   ```
   # cd /var/pcp/pmdas/fsafe
   ```

5. Run the Install utility, which installs the FailSafe performance metrics into the PCP performance metrics namespace:

   ```
   # ./Install
   ```

6. Choose the appropriate configuration for installation of the `fsafe` Performance Metrics Domain Agent (PMDA):

   For Linux FailSafe clusters, since the RPM contains both collector and monitor software, select **both**:

`collector`	Collects performance statistics on this system
`monitor`	Allows this system to monitor local and/or remote systems
`both`	Allows collector and monitor configuration for this system

```
Please enter c(ollector) or m(onitor) or b(oth) {b} b
```

Removing Performance Metrics from a Collector Host

If you wish to remove PCP for FailSafe from a collector host, you will need to remove the PCP for FailSafe metrics from the performance metrics namespace of that host. You can do this before removing the `pcp_fsafe` subsystem by performing the following commands:

1. Change to the `/var/pcp/pmdas/fsafe` directory:

   ```
   # cd /var/pcp/pmdas/fsafe
   ```

2. Run the `Remove` utility:

   ```
   # ./Remove
   ```

Installing the Monitor Host

To install PCP for FailSafe on a designated monitor host, the following software components must already installed on them:

- The `pcp_eoe.sw` subsystem of IRIX 6.5.6 or later, including the subsystem `pcp_eoe.sw.monitor`
- PCP 2.1 or later, including the subsystem `pcp.sw.monitor`

The monitor license (`PCPMON`) must also be installed on the monitor host.

After this software is installed, install the following subsystems of PCP for FailSafe on each collector host. Table 5-1 lists the subsystems required for a collector host, and their approximate sizes:

Table 5-1 PCP for FailSafe Monitor Subsystems

Subsystem	Size in Kbytes
`pcp_fsafe.man.pages`	40
`pcp_fsafe.man.relnotes`	32
`pcp_fsafe.sw.monitor`	516

The instructions for installing on a monitor host is the same as that for a collector host, except that you do not need to install the FailSafe performance metrics into the PCP performance metrics namespace. Please refer to the section "Installing the Collector Host", page 59, disregarding steps 4 to 6.

Using the Visualization Tools

To view statistics about the FailSafe cluster, use the `hbvis`(1) and `rmvis`(1) commands.

The `hbvis`(1) command constructs a display showing the distribution of heartbeat response times for every node in the cluster. Figure 5-1 shows an example display.

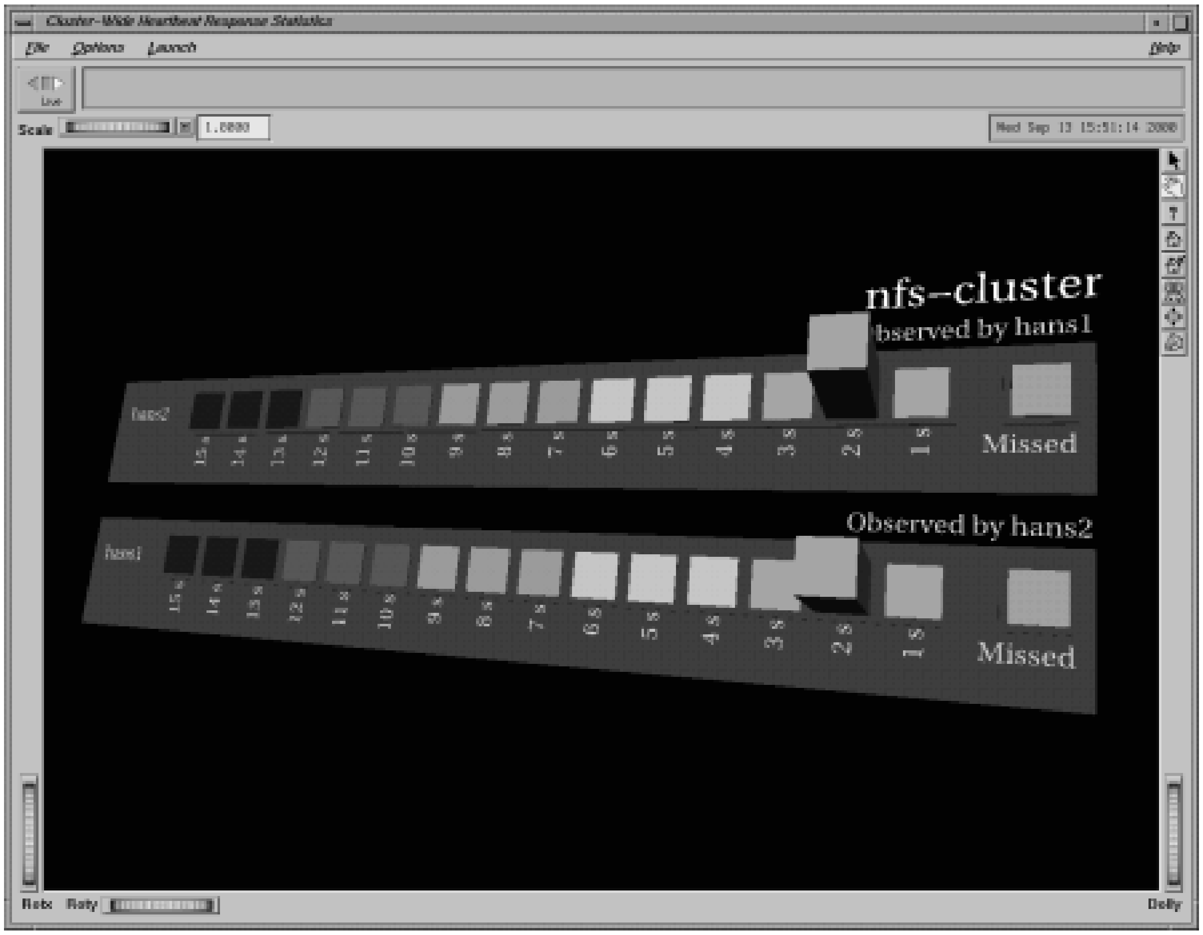

Figure 5-1 Heartbeat Response Statistics

Key features of the display include the frequency of heartbeat responses that arrive at particular intervals within the timeout period, and the frequency of heartbeat responses that have been missed (determined not to have arrived). The bar representing the frequency of missed heartbeat responses changes color to indicate the urgency of problems with availability of a node.

The `rmvis`(1) command constructs a display of the resource monitoring response times for resources monitored on every node of the cluster. Figure 5-2 shows an example display.

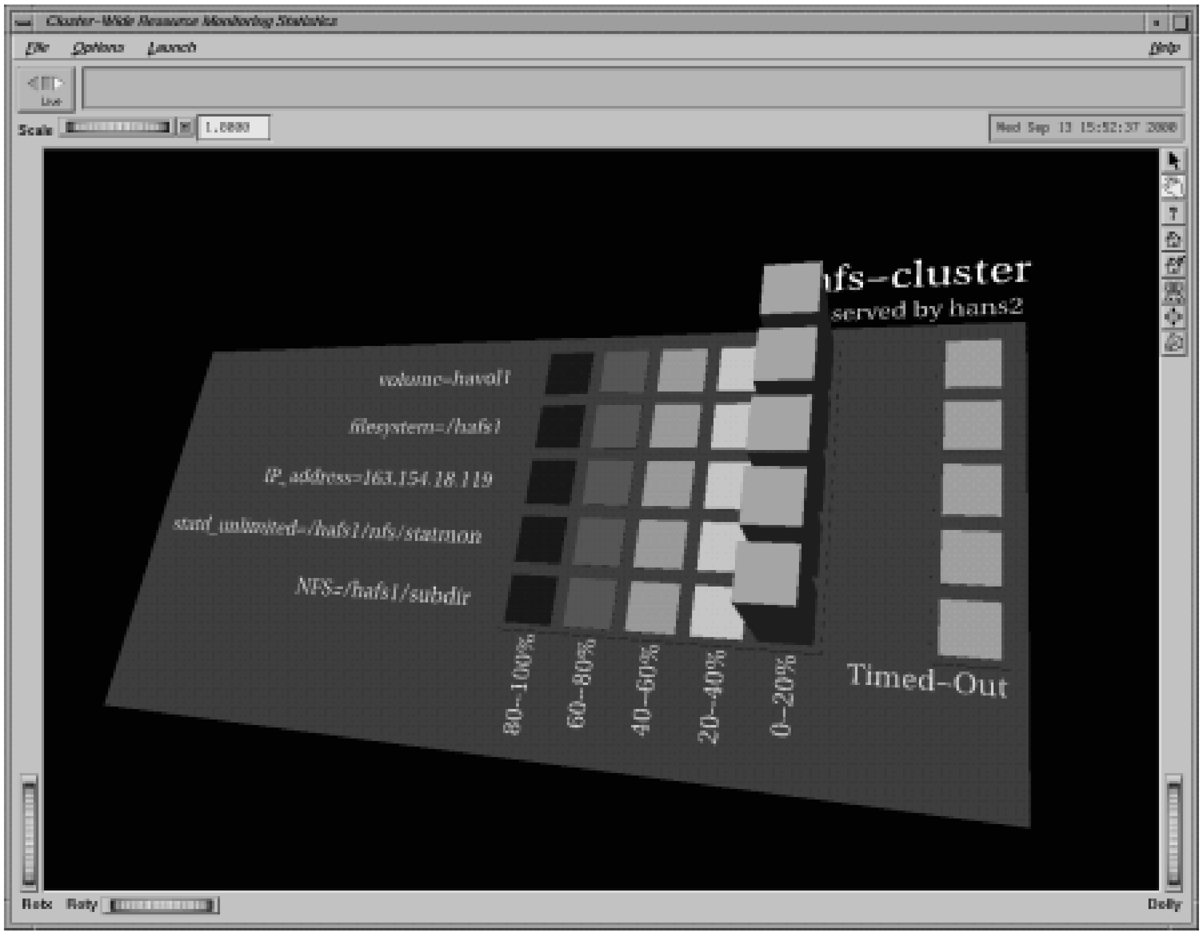

Figure 5-2 Resource Monitoring Statistics

The display is similar in concept to that of `hbvis`(1), showing the frequency of resource monitoring responses that arrive within the timeout period, and the frequency of responses that have timed out. The bar representing the frequency of resource responses that have timed out also changes color to indicate the urgency of problems with the availability of particular resources.

If a node has failed or a resource has failed over, its statistics will disappear from the display.

To run a visualization tool on the monitor host, use the `-h` option to specify an available collector host in the cluster (*host*):

```
% hbvis -h host
```

or

```
% rmvis -h host
```

The collector host specified can be **any** collector host that is a member of the cluster for which you wish to view statistics.

There are various options available to alter the display provided by `hbvis`(1) and `rmvis`(1):

`-H` *hostfile*	Provides a file that lists the nodes that are to appear in the visualization. This is useful in limiting the number of nodes in the display, because it takes more time to construct the display for clusters with more nodes.
`-t` *interval*	Assigns the sampling time of the visualization. There may be circumstances where extending the period of the sampling time may provide better application responsiveness, particularly for clusters with many nodes. Because FailSafe maintains the statistics, `hbvis`(1) and `rmvis`(1) will always show the latest statistics available for the sampling time selected. For details about the *interval* option, see the `pmview`(1) and `PCPIntro`(1) man pages.
`-r`	Selects the FailSafe metrics that present a sampling of statistics taken from the time of the last statistical reset. This enables `hbvis`(1) and `rmvis`(1) to improve the sensitivity of the visualization when abrupt changes appear in the FailSafe monitoring statistics. Without the `-r` option, the statistics presented are from a sampling of FailSafe metrics collected from the time `ha_cmsd`(1m) and/or `ha_srmd`(1m) was last restarted.
`-R`	Starts a new statistical sampling.
`-v`	(`hbvis`(1) only) Provides a visualisation of heartbeat statistics for each node in the cluster, from the point of view of the selected collector host

only. (The collector host is selected using the `-h` option). There is a graphical representation of heartbeat statistics for each node in the cluster as observed by the selected collector host.

`-w`	(`hbvis`(1) only) Provides a visualisation of the aggregate of heartbeat statistics for all nodes in the cluster, from the point of view of the selected collector host only. (The collector host is selected using the `-h` option). There is a only one graphical representation of heartbeat statistics for the entire cluster as observed by the selected collector host.

For a complete description of options, see the `hbvis`(1) and `rmvis`(1) man pages.

`hbvis`(1) and `rmvis`(1) use the command `pmview`(1) to display the 3-D visualization of FailSafe performance metrics. For a description of the various menu commands and controls in the visualization window, consult the man pages for `pmview`(1).

PCP for FailSafe Performance Metrics

PCP tools such as `pmlogger`(1), `pmchart`(1), and `pminfo`(1) can use the metrics exported by PCP for FailSafe.

Appendix A, "Metrics Exported by PCP for FailSafe", page 69, provides a description of PCP for FailSafe metrics. You can also display a description of metrics by using the following command:

```
% pminfo -tT -h host
```

(If you are logged in to a collector host, you can leave out the `-h` option).

Troubleshooting

A grey display (that is, no colored rectangle bars appear on the node's grey baseplane) when using `hbvis`(1) or `rmvis`(1) may indicate one of the following:

- The node is down.

  If you wish to see only the nodes that are up, create a file containing a list of nodes that are to be displayed and pass it as an option to `hbvis`(1)/`rmvis`(1) using the `-H` option (or the environment variable PCP_FSAFE_NODES) so that a new picture of the cluster can be generated. Please refer to the `hbvis`(1)/`rmvis`(1) man pages for more details on the `-H` option.

- The collector daemons have been killed on that node.

  To solve this problem, restart `pmdafsafe`(1) in one of the following ways:

  – If `pmcd`(1) is still running, send `pmcd`(1) the `SIGHUP` signal by entering the following::

  ```
  # killall -HUP pmcd
  ```

  – If `pmcd`(1) is not running, restart PCP by entering the following:

  ```
  # /etc/init.d/pcp start
  ```

- The timeout and sampling settings are too short.

  To change the sampling time, use the time controls available in the `pmview`(1) window. By default, this is 2 seconds; you may need to lengthen the sampling period if you are getting an unsatisfactory display.

  Alternatively, there may be timeout issues between `pmdafsafe`(1) and `pmcd`(1), or between `pmcd`(1) and `pmview`(1). Refer to the man pages for `pmcd`(1) and `PCPIntro`(1) for information on how to change the timeout settings for the various PCP tools.

- The resource has failed over (for `rmvis`(1)).

  In this case, restart `rmvis`(1) so that a new picture of the cluster can be generated.

Metrics Exported by PCP for FailSafe

Table A-1, page 69 lists the metrics implemented by `pmdafsafe`(1).

`fsafe.srm.all.*` metrics are the same as the `fsafe.srm.*` metrics, except that the latest values obtained for all resources will be available, even if `ha_srmd`(1M) or any of the resources themselves are not available.

Table A-1 PCP Metrics

Metric	Description
`fsafe.srm.status` `fsafe.srm.all.status`	Latest status of a monitoring event performed on a resource, for all resources configured to be monitored on this node.
`fsafe.srm.timeout` `fsafe.srm.all.timeout`	The prescribed timeout, in milliseconds, for monitoring a resource.
`fsafe.srm.probes` `fsafe.srm.all.probes`	Number of times a resource has been monitored, for all resources configured to be monitored on this node, since the time `ha_smrd`(1M) has started.
`fsafe.srm.recent.probes`	Number of times a resource has been monitored, for all resources configured to be monitored on this node, since a data collection reset (via `fsafe.control.reset_srm`).
`fsafe.srm.timeouts` `fsafe.srm.all.timeouts`	Number of resource monitoring events that have timed out before declaring that resource as failed, for all resources configured to be monitored on this node, since the time the resources have last been available.
`fsafe.srm.recent.timeouts`	Number of resource monitoring events that have timed out before declaring that resource as failed, for all resources configured to be monitored on this node, since a data collection reset (via `fsafe.control.reset_srm`).

Metric	Description
`fsafe.srm.min_resp` `fsafe.srm.all.min_resp`	Approximate minimum time, in milliseconds, taken to complete a monitoring event on a resource, for all resources configured to be monitored
`fsafe.srm.max_resp` `fsafe.srm.all.max_resp`	Approximate maximum time, in milliseconds, taken to complete a monitoring event on a resource, for all resources configured to be monitored on this node.
`fsafe.srm.last_resp` `fsafe.srm.all.last_resp`	Approximate time, in milliseconds, taken in completing the most recent monitoring event on a resource, for all resources configured to be monitored on this node.
`fsafe.srm.cumm_timeouts` `fsafe.srm.all.cumm_timeouts`	Cumulative number of resource monitoring events that have timed out, for all resources configured to be monitored on this node, since the time `ha_smrd`(1M) has started.
`fsafe.srm.recent.cumm_timeouts`	Cumulative number of resource monitoring events that have timed out, for all resources configured to be monitored on this node, since a data collection reset (via `fsafe.control.reset_srm`).
`fsafe.srm.histo_20` `fsafe.srm.all.histo_20`	Fraction of monitoring events that have been received within 0- 20% of the response time from 0 milliseconds to `fsafe.srm.timeout`, for all resources configured to be monitored on this node, since the time `ha_smrd`(1M) has started.
`fsafe.srm.recent.histo_20`	Fraction of monitoring events that have been received within 0- 20% of the response time from 0 milliseconds to `fsafe.srm.timeout`, for all resources configured to be monitored on this node, since a data collection reset (via `fsafe.control.reset_srm`).
`fsafe.srm.histo_40` `fsafe.srm.all.histo_40`	Fraction of monitoring events that have been received within 20- 40% of the response time from 0 milliseconds to `fsafe.srm.timeout`, for all resources configured to be monitored on this node, since the time `ha_smrd`(1M) has started.

Metric	Description
`fsafe.srm.recent.histo_40`	Fraction of monitoring events that have been received within 20- 40% of the response time from 0 milliseconds to `fsafe.srm.timeout`, for all resources configured to be monitored on this node, since a data collection reset (via `fsafe.control.reset_srm`).
`fsafe.srm.histo_60` `fsafe.srm.all.histo_60`	Fraction of monitoring events that have been received within 40- 60% of the response time from 0 milliseconds to `fsafe.srm.timeout`, for all resources configured to be monitored on this node, since the time `ha_smrd(1M)` has started.
`fsafe.srm.recent.histo_60`	Fraction of monitoring events that have been received within 40- 60% of the response time from 0 milliseconds to `fsafe.srm.timeout`, for all resources configured to be monitored on this node, since a data collection reset (via `fsafe.control.reset_srm`).
`fsafe.srm.histo_80` `fsafe.srm.all.histo_80`	Fraction of monitoring events that have been received within 60- 80% of the response time from 0 milliseconds to `fsafe.srm.timeout`, for all resources configured to be monitored on this node, since the time `ha_smrd`(1M) has started.
`fsafe.srm.recent.histo_80`	Fraction of monitoring events that have been received within 60- 80% of the response time from 0 milliseconds to `fsafe.srm.timeout`, for all resources configured to be monitored on this node, since a data collection reset (via `fsafe.control.reset_srm`).
`fsafe.srm.histo_100` `fsafe.srm.all.histo_100`	Fraction of monitoring events that have been received within 80- 100% of the response time from 0 milliseconds to `fsafe.srm.timeout`, for all resources configured to be monitored on this node, since the time `ha_smrd`(1M) has started.

Metric	Description
`fsafe.srm.recent.histo_100`	Fraction of monitoring events that have been received within 80- 100% of the response time from 0 milliseconds to `fsafe.srm.timeout`, for all resources configured to be monitored on this node, since a data collection reset (via `fsafe.control.reset_srm`).
`fsafe.srm.frac_timeouts` `fsafe.srm.all.frac_timeouts`	Fraction of monitoring events that have timed out before declaring that resource as failed, for all resources configured to be monitored on this node, since the time the resources have last been available.
`fsafe.srm.recent.frac_timeouts`	Fraction of monitoring events that have timed out, before declaring that resource as failed, for all resources configured to be monitored on this node, since a data collection reset (via `fsafe.control.reset_srm`).
`fsafe.srm.frac_cumm_timeouts` `fsafe.srm.all.frac_cumm_timeouts`	Fraction of cumulative number of monitoring events that have timed out, for all resources configured to be monitored on this node, since the time `ha_smrd`(1M) has started.
`fsafe.srm.recent.frac_cumm_timeouts`	Fraction of cumulative number of monitoring events that have timed out, for all resources configured to be monitored on this node, since a data collection reset (via `fsafe.control.reset_srm`).
`fsafe.srm.recent.timestamp`	The time when a new collection of statistics was started for the `fsafe.srm.recent.*` metrics, after issuing a store to the metric `fsafe.control.reset_srm`.
`fsafe.config.clustername`	The name of this cluster.
`fsafe.config.hostname`	The name of all hosts in the cluster specified by `fsafe.config.clustername`.
`fsafe.config.nnodes`	Number of nodes in the cluster specified by `fsafe.config.clustername`.
`fsafe.config.cms.interval`	The cluster heartbeat event interval, in milliseconds.

Metric	Description
`fsafe.config.cms.timeout`	The heartbeat event timeout for all nodes in the cluster, in milliseconds.
`fsafe.config.cms.nbuckets`	The number of heartbeat event response intervals per node, where each interval covers a time equal to the heartbeat event interval (`fsafe.config.cms.interval`) for segments of time until the heartbeat event timeout (`fsafe.config.cms.timeout`).
`fsafe.control.debug`	Debugging flags for the `fsafe` PMDA when a decimal integer value is stored to this metric. It ultimately affects what information is put into the `fsafe` PMDA's log (normally at `/var/adm/pcplog/fsafe.log`).
	Reading this metric will return the currently assigned debugging flags as a decimal integer.
`fsafe.control.reset_cms`	Resets data collection statistics for all metrics gathered from `ha_cmsd`(1M). When this metric is stored to, the data provided is ignored; it is the act of storing to this metric which causes the reset.
	Reading this metric will return zero (0).
`fsafe.control.reset_srm`	Resets data collection statistics for all metrics gathered from `ha_srmd`(1M). When this metric is stored, the data provided is ignored; it is the act of storing to this metric which causes the reset.
	Reading this metric will return zero (0).
`fsafe.control.retry`	Sets the number of retries permitted when contacting `ha_cmsd`(1M) or `ha_srmd`(1M), and when the daemons indicate that they are busy.

Metric	Description
	Depending on which metrics are being read, and which daemon is required to obtain values for the required metrics, values for some metrics may not be available, possibly producing the message "`Try again. Information not currently available.`" This metric can be adjusted in order to increase the number of retries permitted when collecting metrics, before giving up and displaying this message. A retry is performed once every 100 ms (approximately). Note that setting this metric does not alter how the fsafe PMDA handles more serious errors from `ha_cmsd`(1M) or `ha_srmd`(1M). Reading this metric will return the current retry count.
`fsafe.cms.expected`	The number of heartbeat events expected to have been received for each node in the cluster (excluding the collector host), since the time `ha_cmsd`(1M) has started.
`fsafe.cms.recent.expected`	The number of heartbeat events expected to have been received for each node in the cluster (excluding the collector host), since a data collection reset (via `fsafe.control.reset_cms`).
`fsafe.cms.received`	The number of heartbeat events actually received for each node in the cluster (excluding the collector host), since the time `ha_cmsd`(1M) has started.
`fsafe.cms.recent.received`	The number of heartbeat events actually received for each node in the cluster (excluding the collector host), since a data collection reset (via `fsafe.control.reset_cms`).
`fsafe.cms.missed`	The number of heartbeat events determined not to have been received for each node in the cluster (excluding the collector host), since the time `ha_cmsd`(1M) has started.

Metric	Description
`fsafe.cms.recent.missed`	The number of heartbeat events determined not to have been received for each node in the cluster (excluding the collector host), since a data collection reset (via `fsafe.control.reset_cms`).
`fsafe.cms.histo`	Histogram of heartbeat event response times for events that have occurred within discrete heartbeat response intervals for each node in the cluster (excluding the collector host), since the time `ha_cmsd`(1M) has started. The heartbeat response intervals are defined to be equal to the configured heartbeat event interval (`fsafe.config.cms.interval`), for a number of intervals up to the configured heartbeat event timeout (`fsafe.config.cms.timeout`).
`fsafe.cms.recent.histo`	Histogram of heartbeat event response times for events that have occurred within discrete heartbeat response intervals for each node in the cluster (excluding the collector host), since a data collection reset (via `fsafe.control.reset_cms`). The heartbeat response intervals are defined to be equal to the configured heartbeat event interval (`fsafe.config.cms.interval`), for a number of intervals up to the configured heartbeat event timeout (`fsafe.config.cms.timeout`).
`fsafe.cms.frac_received`	Fraction of heartbeat events received over all expected events for each node in the cluster, since the time `ha_cmsd`(1M) has started.
`fsafe.cms.recent.frac_received`	Fraction of heartbeat events received over all expected events for each node in the cluster, since a data collection reset (via `fsafe.control.reset_cms`).
`fsafe.cms.frac_missed`	Fraction of heartbeat events determined not to have been received over all expected events for each node in the cluster, since the time `ha_cmsd`(1M) has started.

Metric	Description
`fsafe.cms.recent.frac_missed`	Fraction of heartbeat events determined not to have been received over all expected events for each node in the cluster, since a data collection reset (via `fsafe.control.reset_cms`).
`fsafe.cms.recent.timestamp`	The time when a new collection of statistics was started for the `fsafe.cms.recent.*` metrics, after issuing a store to the metric `fsafe.control.reset_cms`.
`fsafe.cms.pernode.expected`	The number of heartbeat events expected to have been received for a particular node in the cluster, since the time `ha_cmsd`(1M) has started.
`fsafe.cms.recent.pernode.expected`	The number of heartbeat events expected to have been received for a particular node in the cluster, since a data collection reset (via `fsafe.control.reset_cms`).
`fsafe.cms.pernode.received`	The number of heartbeat events actually received for a particular node in the cluster, since the time `ha_cmsd`(1M) has started.
`fsafe.cms.recent.pernode.received`	The number of heartbeat events actually received for a particular node in the cluster, since a data collection reset (via `fsafe.control.reset_cms`).
`fsafe.cms.pernode.missed`	The number of heartbeat events determined not to have been received for a particular node in the cluster, since the time `ha_cmsd`(1M) has started.
`fsafe.cms.recent.pernode.missed`	The number of heartbeat events determined not to have been received for a particular node in the cluster, since a data collection reset (via `fsafe.control.reset_cms`).
`fsafe.cms.pernode.histo`	Histogram of heartbeat event response times for events that have occurred within discrete heartbeat response intervals for a particular node in the cluster, since the time `ha_cmsd`(1M) has started.

Metric	Description
	The heartbeat response intervals are defined to be equal to the configured heartbeat event interval (`fsafe.config.cms.interval`), for a number of intervals up to the configured heartbeat event timeout (`fsafe.config.cms.timeout`).
`fsafe.cms.recent.pernode.histo`	Histogram of heartbeat event response times for events that have occurred within discrete heartbeat response intervals for a particular node in the cluster, since a data collection reset (via `fsafe.control.reset_cms`). The heartbeat response intervals are defined to be equal to the configured heartbeat event interval (`fsafe.config.cms.interval`), for a number of intervals up to the configured heartbeat event timeout (`fsafe.config.cms.timeout`).
`fsafe.cms.pernode.frac_received`	Fraction of heartbeat events received over all expected events for a particular node in the cluster, since the time `ha_cmsd`(1M) has started.
`fsafe.cms.recent.pernode.frac_received`	Fraction of heartbeat events received over all expected events for a particular node in the cluster, since a data collection reset (via `fsafe.control.reset_cms`).
`fsafe.cms.pernode.frac_missed`	Fraction of heartbeat events determined not to have been received over all expected events for a particular node in the cluster, since the time `ha_cmsd`(1M) has started.
`fsafe.cms.recent.pernode.frac_missed`	Fraction of heartbeat events determined not to have been received over all expected events for a particular node in the cluster, since a data collection reset (via `fsafe.control.reset_cms`).

Index