IRIS FailSafe™ Administrator's Guide

Document Number 007-3109-002

CONTRIBUTORS

Written by Susan Ellis
Illustrated by Dany Galgani
Production by Ruth Christian
Engineering contributions by Gilberto Arnaiz, Ashwinee Dinkar, Raghu Mallena, Michael Nishimoto, and Paddy Sreenivasan
Cover design and illustration by Rob Aguilar, Rikk Carey, Dean Hodgkinson, Erik Lindholm, and Kay Maitz

© 1996, Silicon Graphics, Inc.— All Rights Reserved
The contents of this document may not be copied or duplicated in any form, in whole or in part, without the prior written permission of Silicon Graphics, Inc.

RESTRICTED RIGHTS LEGEND
Use, duplication, or disclosure of the technical data contained in this document by the Government is subject to restrictions as set forth in subdivision (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS 52.227-7013 and / or in similar or successor clauses in the FAR, or in the DOD or NASA FAR Supplement. Unpublished rights reserved under the Copyright Laws of the United States. Contractor / manufacturer is Silicon Graphics, Inc., 2011 N. Shoreline Blvd., Mountain View, CA 94043-1389.

Silicon Graphics, CHALLENGE, and IRIS are registered trademarks and IRIX, XFS, WebFORCE, Indy, IRISconsole, and IRIS FailSafe are trademarks of Silicon Graphics, Inc. NFS (Network File System) is a trademark of Sun Microsystems, Inc. Oracle is a registered trademark and Oracle Parallel Server and OPS are trademarks of Oracle Corporation. Netsite and Netscape Communications Server are trademarks of Netscape Communications. Sybase is a registered trademark and SQL Server is a trademark of Sybase, Inc. INFORMIX is a registered trademark of Informix Software, Inc. FLEX*lm* is a trademark of Globetrotter Software, Inc. NetLS is a trademark of Apollo Computer, Inc., a subsidiary of Hewlett-Packard Company.

IRIS FailSafe™ Administrator's Guide
Document Number 007-3109-002

Contents

List of Examples

List of Figures

List of Tables

About This Guide

The configuration and administration of the IRIS FailSafe system software and its IRIS FailSafe NFS and IRIS FailSafe Webserver options are described in this guide, the *IRIS FailSafe Administrator's Guide*. The configuration of other IRIS FailSafe options, such IRIS FailSafe Sybase®, IRIS FailSafe INFORMIX®, and IRIS FailSafe Oracle®, is described in separate guides. They are listed in the section "Related Documentation" below.

This guide was prepared in conjunction with Release 1.1 of the IRIS FailSafe product.

Audience

The *IRIS FailSafe Administrator's Guide* is written for the person who administers the IRIS FailSafe system. The IRIS FailSafe administrator must be familiar with the operation of CHALLENGE® servers, as well as optional Vault storage systems or the CHALLENGE RAID storage system, whichever is used in the IRIS FailSafe configuration. Good knowledge of XLV and XFS is also required.

Structure of This Guide

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

- Chapter 1, "Overview of the IRIS FailSafe System," introduces the components of the IRIS FailSafe system and explains its hardware and software architecture. This chapter also describes IRIS FailSafe node states and how failover works. It gives overviews of the procedures for configuring a new IRIS FailSafe system and for upgrading an IRIS FailSafe system from release 1.0 to release 1.1.
- Chapter 2, "Planning IRIS FailSafe Configuration," describes how to plan the configuration of an IRIS FailSafe cluster.

- Chapter 3, “Configuring Nodes for IRIS FailSafe,” describes several procedures that must be performed on nodes in an IRIS FailSafe cluster to prepare them for IRIS FailSafe.
- Chapter 4, “Creating the IRIS FailSafe Configuration File,” explains how to edit configuration file templates to create or modify an IRIS FailSafe configuration file.
- Chapter 5, “Testing IRIS FailSafe Configuration,” describes how to test the configured IRIS FailSafe system.
- Chapter 6, “Administering IRIS FailSafe,” explains how to perform basic IRIS FailSafe system administration tasks, such as starting and shutting down the IRIS FailSafe system, getting current status information, and using administrative commands to change node state.
- Chapter 7, “Upgrading an IRIS FailSafe Cluster,” explains the procedures for upgrading the hardware and software on IRIS FailSafe nodes, including installing new IRIX releases, IRIS FailSafe software, and patches, adding new highly available services, and installing a new version of the configuration file, */var/ha/ha.conf*.
- Chapter 8, “Performing Simple Hardware Maintenance,” explains how to perform simple cable replacement procedures and how to replace batteries in the remote power control unit that is included in IRIS FailSafe clusters with CHALLENGE S systems.
- Appendix A, “Messages About Configuration File Errors,” lists error messages that can appear as output of the *ha_cfgverify* command and gives an explanation of each one.
- Appendix B, “System Troubleshooting,” explains how to troubleshoot IRIS FailSafe system problems.

A glossary completes this guide.

Related Documentation

Besides this guide, other documentation for the IRIS FailSafe system includes

- *IRIS FailSafe Programmer’s Guide* (007-3298-xxx)
- *IRIS FailSafe INFORMIX Administrator’s Guide* (007-3268-xxx, IRIS / FailSafe INFORMIX option)

- *IRIS FailSafe Oracle Administrator's Guide* (007-3269-xxx, IRIS FailSafe Oracle option)
- *IRIS FailSafe Sybase Administrator's Guide* (007-3306-xxx, IRIS FailSafe Sybase option)

Other documentation that provides information critical to the use of IRIS FailSafe is:

- *Software Installation Administrator's Guide* (007-1364-050, for IRIX 5.3 only)
- *IRIX Admin: Software Installation and Licensing* (007-1364-060, for IRIX 6.2 only)
- *Getting Started with XFS Filesystems* (007-2549-001, for IRIX 5.3 only)
- *IRIX Admin: Disks and Filesystems* (007-2825-xxx)
- *Netscape Commerce and Communications Servers Administrator's Guide* (007-2909-xxx)
- *ONC3/NFS Administrator's Guide* (007-0850-xxx)
- *CHALLENGE RAID Owner's Guide* (007-2532-xxx)

The IRIS FailSafe reference pages are as follows:

- ha_admin(1M)
- ha_appmon(1M)
- ha_cfgchksum(1M)
- ha_cfginfo(1M)
- ha_cfgverify(1M)
- ha_exec(1M)
- ha_hbeat(1M)
- ha_ifa(1M)
- ha_ifmx(1M) (IRIS FailSafe INFORMIX option)
- ha_killd(1M)
- ha_nc(1M)
- ha_orcl(1M) (IRIS FailSafe Oracle option)
- ha_spng(1M)
- ha_sybs(1M) (IRIS FailSafe Sybase option)
- http_ping(1M) (IRIS FailSafe Web option)
- macconfig(1M)

- ha.conf(4)
- failsafe(7M)

Release notes are included with each IRIS FailSafe product. The names of the release notes are as follows:

ha_base	release notes for IRIS FailSafe
ha_nfs	release notes for IRIS FailSafe NFS
ha_www	release notes for IRIS FailSafe Web
ha_ orcl	release notes for IRIS FailSafe Oracle
ha_ ifmx	release notes IRIS FailSafe INFORMIX
ha_ sybs	release notes IRIS FailSafe Sybase

Conventions

These type conventions and symbols are used in this guide:

Bold	Literal command-line arguments and literal parameter values
Italics	Command names, filenames, new terms, the names of *inst* subsystems, manual / book titles, variable command-line arguments, and variables to be supplied by the user in examples, code, and syntax statements
`Fixed-width type`	Examples of command output that is displayed in windows on your monitor and of the contents of files
`Bold fixed-width type`	Commands and text that you are to type literally in response to shell and command prompts
#	IRIX shell prompt for the superuser (*root*)

Chapter 1

Overview of the IRIS FailSafe System

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

If your IRIS FailSafe system is running the 1.0 release of IRIS FailSafe and you plan to upgrade it to the 1.1 Release, you can skip to the last major section of this chapter, "Overview of IRIS FailSafe 1.1 for IRIS FailSafe 1.0 Users," for information about the new and changed features in 1.1 and how to upgrade the system to 1.1.

What Is High Availability?

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

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

SGI's high availability solution, IRIS FailSafe, is based on a two node cluster. This provides redundancy of processors and I/O controllers. The redundancy of storage is obtained through the use of dual-hosted RAID devices and plexed (mirrored) disks.

If one of the nodes in the cluster or one of the nodes' components fails, the second node restarts the highly available services of the failed node. In the client/server paradigm, the client does not care which of the two nodes in the cluster is providing the service. The clients see only a brief interruption of the service.

Highly available services are monitored by the IRIS FailSafe software. During normal operation, if a failure is detected on any of these components, a *failover* process is initiated on the surviving node. This process consists of isolating the failed node (to ensure data consistency), doing any recovery required by the failed over services, and quickly restarting the services on the surviving node.

In a high availability system, each node serves as backup for the other node. Unlike the backup resources in a fault-tolerant system, which serves purely as redundant hardware for backup in case of failure, the resources of each node in a high availability system can be used during normal operation.

What Is IRIS FailSafe?

The Silicon Graphics® IRIS FailSafe product provides a general facility for providing highly available services. These services fall into two groups: highly available resources and highly available applications. Highly available resources are network interfaces, XLV logical volumes, and XFS™ filesystems that have been configured for IRIS FailSafe. Optional IRIS FailSafe products are available for these applications: NFS™, the Netscape™ Communications Server™, the Netscape Commerce Server, Sybase, INFORMIX, and Oracle.

The Silicon Graphics IRIS FailSafe system consists of two CHALLENGE, POWER CHALLENGE, or Onyx servers that provide highly available services. The servers need not be the same model, for example an IRIS FailSafe cluster can consist of a CHALLENGE L server and a CHALLENGE S server. Disks are shared by physically attaching them to both the nodes in the system. The two servers (called *nodes* throughout this guide) and disks, along with IRIS FailSafe software, form an IRIS FailSafe *cluster*.

While running highly available services, the nodes can run other applications that are not highly available. All highly available services are owned and accessed by one node at a time.

The IRIS FailSafe system supports fast *failover*: if a highly available service—interface, disk, application, or the node itself—fails, the service quickly (although not instantaneously) resumes because the second node in the system shuts down the failed node and takes over all of its services. To clients, the services on the second node are indistinguishable from the original services before failure occurred. It appears as if the original node has crashed and rebooted quickly. The clients notice a brief interruption in the highly available service.

Two configurations are possible:

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

The base software for the IRIS FailSafe system consists of IRIX™ 5.3 with XFS or IRIX 6.2, IRIX patches, IRIS FailSafe software, FDDI software (if FDDI networking is used), and software for the optional CHALLENGE RAID storage system.

There are IRIS FailSafe software options for some highly available applications. Optional software includes:

- IRIS FailSafe NFS
- IRIS FailSafe Web (for Netscape servers)
- IRIS FailSafe INFORMIX
- IRIS FailSafe Sybase
- IRIS FailSafe Oracle

IRIS FailSafe enhances the Silicon Graphics Oracle Parallel Server™ (OPS™) by providing IP failover in an OPS hardware configuration. However, the two products are not merged administratively, so different tools are required to maintain a combined system.

IRIS FailSafe provides a framework for making applications into highly available resources. If you want to add highly available applications on an IRIS FailSafe cluster, you must write scripts to handle monitoring and failover functions. In addition, you must add the new highly available applications to the IRIS FailSafe configuration file */va/ha/ha.conf* to register the application and scripts with the IRIS FailSafe software. Developing these scripts and making additions to the configuration file is described in the *IRIS FailSafe Programmer's Guide*.

Hardware Components of an IRIS FailSafe Cluster

Figure 1-1 shows the IRIS FailSafe hardware components.

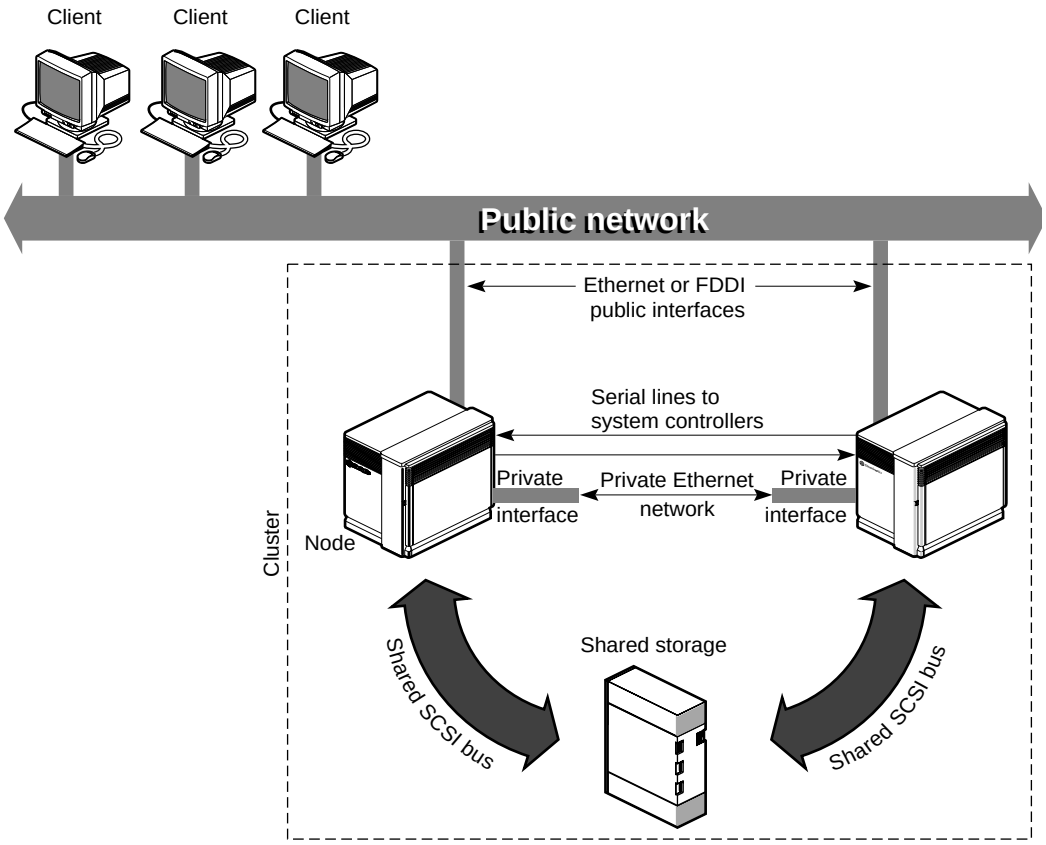


Figure 1-1 IRIS FailSafe System Components

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

- two CHALLENGE nodes—S, DM, L, XL, POWER CHALLENGE, or Onyx—in any combination
- one or more interfaces on each node to one or more public networks (Ethernet or FDDI)

  These public interfaces attached to each node connect the node to one or more public networks, which link the cluster to clients. Each public interface has an IP address called a *fixed IP address* that doesn't move to the other node in the cluster during failover. Each public interface can have additional IP addresses, called *high availability IP addresses*, that are transferred to an interface on the surviving node in case of failover.
- a serial line from a serial port on each node to the other's Remote System Control port (to the Silicon Graphics remote power control unit on CHALLENGE S nodes)

  A surviving node uses this line to reboot the failed node during takeover. This procedure ensures that the failed node is not using the shared disks when the surviving node takes them over.
- one interface on each node for the private network (Ethernet or FDDI)

  One Ethernet or FDDI interface on each node is required for the private *heartbeat* connection, by which each node monitors the state of the other node. The IRIS FailSafe software also uses this connection to pass control messages between nodes. These interfaces, called private interfaces, have distinct IP addresses that are kept private for security reasons.
- disk storage and SCSI bus shared by the nodes in the cluster

  The nodes in the IRIS FailSafe system share dual-hosted disk storage over a shared fast and wide SCSI bus. The bus is shared so that either node can take over the disks in case of failure. The hardware required for the disk storage is either:

  – a CHALLENGE Vault peripheral enclosure with SCSI disks

  – CHALLENGE RAID deskside or rackmount storage system; each chassis assembly has two storage-control processors (SPs) and at least five disk modules with caching enabled

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

IRIS FailSafe Software Architecture

IRIS FailSafe software includes a set of processes running on each node and communicating with each other. These processes use scripts failover and recovery operations and for monitoring the highly available services on each node. These processes and scripts read IRIS FailSafe configuration information from a configuration file, */var/ha/ha.conf*.

The IRIS FailSafe daemons and scripts are shown in Figure 1-2. For each daemon or set of scripts, the diagram shows other daemons and scripts it communicates with and the communication path it uses.

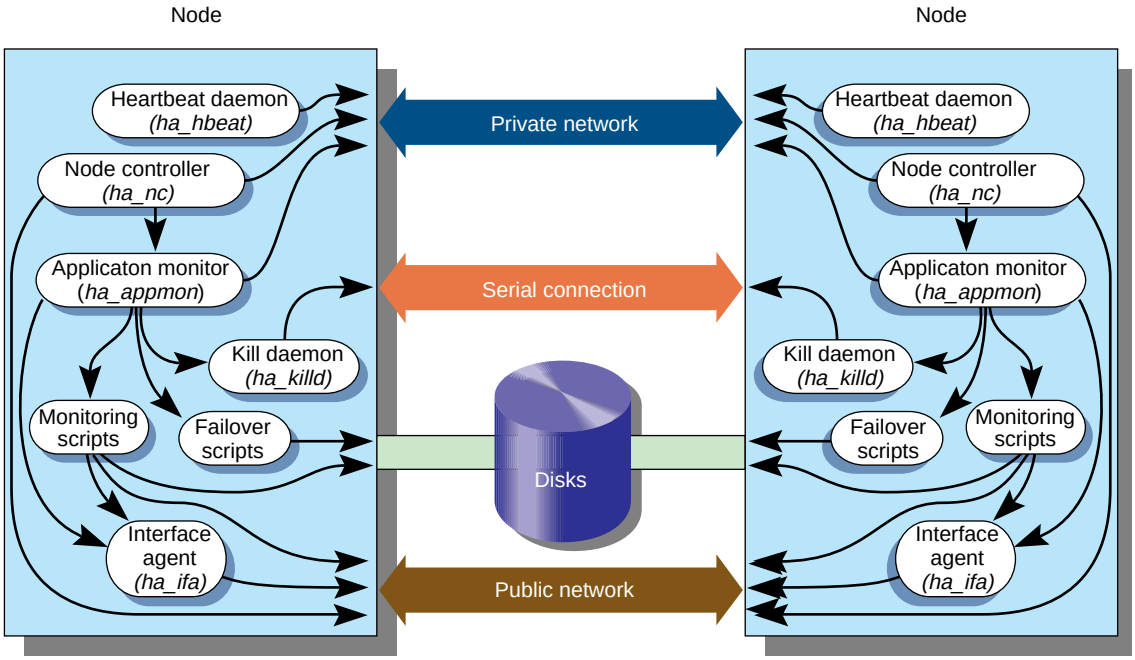


Figure 1-2 IRIS FailSafe Software Architecture

These IRIS FailSafe daemons and scripts are described in the following subsections.

Heartbeat Daemon

The heartbeat daemon *ha_hbeat* runs in each node and is the first IRIS FailSafe process to start. It is polled by the application monitor on the other node. These heartbeat messages enable each node to determine the liveliness of the other node. Heartbeat messages are passed on the private network. If there is a failure of the private network, heartbeat messages can be passed on the public network.

Node Controller

The node controller process *ha_nc* determines each node's current state. The node states are described in the section "Node States" in this chapter.

The node controllers in the cluster pass messages to each other over the private network. If there is a failure of the private network, the node controllers don't use the public network.

Application Monitor

On each node, the application monitor process *ha_appmon* monitors all services on both nodes and reports any failures to the node controller.

The application monitor polls the heartbeat daemon on the other node to determine its liveliness. It also executes the failover scripts during state transitions.

Because the application monitor *ha_appmon* is a multi-threaded process, you may see several instances of *ha_appmon* running on a node simultaneously when you look at the output of the *ps* command.

Kill Daemon

The kill daemon *ha_killd* on each node monitors the serial connection to the other node and provides the power-cycling capability.

Interface Agent

The interface agent monitors all local interfaces to determine if they are still functioning.

The interface agent uses the number of input packets as the criteria to determine whether a network interface is working or not. The interface agent injects packets into public network if it finds the number of input packets in an interface is not increasing. This prevents false failovers in networks that do not have any I/O activity.

Highly Available Resources

This section discusses the highly available resources that are provided on an IRIS FailSafe system.

Nodes

If a node crashes or hangs (for example, due to a parity error or bus error), it will not respond to the heartbeat message sent by the application monitor on the other node. The other (good) node takes over the failed node's services after resetting the failed node.

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

Network Interfaces and IP Addresses

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

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

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

IRIS FailSafe requires that each network interface in a cluster have an IP address that does not failover. These IP addresses, called *fixed IP addresses*, are used to monitor network interfaces. Each fixed IP address must be configured to a network interface at system boot up time. All other IP addresses in the cluster are configured as *high availability IP addresses*. They are included in the IRIS FailSafe configuration file */var/ha/ha.conf*.

High availability IP addresses are configured on a network interface and moved to another network interface in the other node by IRIS FailSafe during a failover or recovery process. IRIS FailSafe uses the *ifconfig* command to configure an IP address on a network interface and to move IP addresses from one interface to another.

In some networking implementations, it is not sufficient to move IP addresses from one interface to another using the *ifconfig* command. IRIS FailSafe uses *MAC address impersonation* (also called *re-mac*'ing) to support these networking implementations. MAC address impersonation moves the physical network address of a network interface to another interface. If an interface is configured for MAC address impersonation in the IRIS FailSafe configuration file, IRIS FailSafe moves the IP address using the *ifconfig* command and moves the physical address using the *macconfig* command.

Since the physical address of an interface is moved in MAC address impersonation, each network interface has to be backed up by a dedicated backup interface (each backup interface backs up only one network interface) if MAC address impersonation is required. CISCO routers and PC / NFS clients require MAC address impersonation.

See the section "Planning Network Interface and IP Address Configuration" in Chapter 2 for more information about determining if MAC address impersonation is required.

Disks

The IRIS FailSafe system includes shared SCSI-based storage in the form of one or more CHALLENGE RAID storage systems or CHALLENGE Vaults with plexed disks. All data for highly available applications must be stored in XLV logical volumes on shared disks. If highly available applications use filesystems, XFS filesystems must be used.

For CHALLENGE RAID storage systems, if a disk or disk controller fails, the RAID storage system is equipped to keep services available through its own capabilities.

With plexed XLV logical volumes on the disks in a CHALLENGE Vault, the XLV system provides redundancy. No participation of the IRIS FailSafe system software is required for a disk failure. If a disk controller fails, the IRIS FailSafe system software initiates the failover process.

Figure 1-3 shows disk storage takeover. The surviving node takes over the shared disks and recovers the logical volumes and filesystems on the disks. This process is expedited by the XFS filesystem, which supports fast recovery because it uses journaling technology that does not require the use of the *fsck* command for filesystem consistency checking.

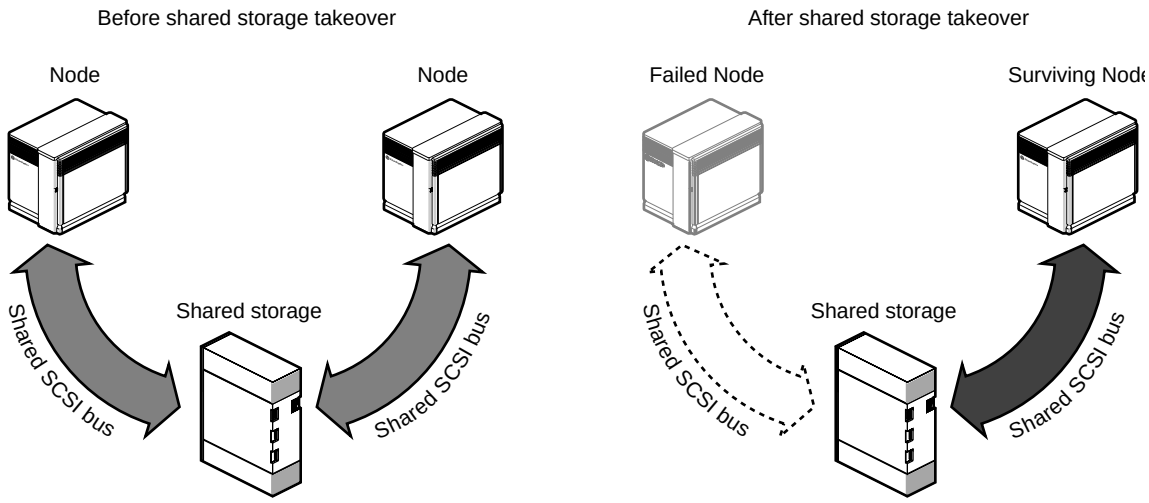


Figure 1-3 Disk Storage Failover

Highly Available Applications

Each application has a primary node and backup node. The primary node is the node on which the application runs when FailSafe is in *normal state*. When a failure of any highly available resources or highly available application is detected by IRIS FailSafe software, all highly available resources on the failed node are failed over to the other node and the highly available applications on the failed node are stopped. When these operations are complete, the highly available applications are started on the backup node.

All information about highly available applications, including the primary node and backup node for the application and monitoring scripts, is specified in the IRIS FailSafe configuration file. Monitoring scripts detect the failure of a highly available application.

IRIS FailSafe option products provide monitoring scripts and failover scripts that make NFS, Web, Oracle, INFORMIX, and Sybase applications highly available.

The IRIS FailSafe software provides a framework for making applications highly available. By writing scripts and modifying the IRIS FailSafe configuration file, you can turn client/server applications into highly available applications. For information, see the *IRIS FailSafe Programmer's Guide*.

The Failover and Recovery Processes

When a failure is detected on one node (the node has crashed, hung, or been shut down, or a highly available service is no longer operating), the other node performs a failover of the highly available services that are being provided on the node with the failure (called the *failed node*). Failover makes all of the highly available services, previously provided by both nodes in a cluster, available on the surviving node in the cluster. This is called *degraded state*. (Node states are more fully described in the next section, "Node States.")

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

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

- Stops all highly available applications running on the node
- Moves all highly available resources (IP addresses and shared disks) to the other node
- Sends a message to the other node (surviving node) to start providing all highly available resources and applications previously provided by the failed node
- Moves to a state called *standby state*

When it receives the message, the surviving node performs these operations:

- Transfers ownership of all the highly available resources from the failed node to itself
- Starts offering the highly available resources of the failed node and the applications that were running on the failed node
- Moves to degraded state

If the failure is detected by FailSafe software running on the other node, the node detecting the failure (the surviving node) performs these operations:

- Using the serial connection between the nodes, reboots the failed node to prevent corruption of data
- Transfers ownership of all the highly available resources from the failed node to itself
- Starts offering the highly available resources of the failed node and the applications that were running on the failed node
- Moves to degraded state

When a failed node is coming back up (called a *recovering node*), it determines if the other node is running. There are three possible scenarios:

- If IRIS FailSafe is not running on the other node or if the private interfaces or private network are not functioning, the recovering node does not begin providing highly available services. It goes into standby state.
- If IRIS FailSafe is running on the other node and controlled failback is configured on for the node, the recovering node doesn't begin proving highly available services; it goes to a state called *controlled failback state*.

- If IRIS FailSafe is running on the other node and controlled failback is configured on for the node, the surviving node shuts down the highly available services for which it is the backup node, and the recovering node begins providing the highly available services for which it is the primary node.

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

Node States

Each node that is running IRIS FailSafe software is in one of the six states described in Table 1-1.

Table 1-1 Node States

Node State	Definition
joining	The node is coming up and joining the cluster. The node should never remain in this state for more than two or three minutes.
normal	The node is actively providing its own highly available services.
degraded	The node is providing all highly available services for the cluster; the other node is unavailable.
standby	This node has stopped monitoring the other node in the cluster and is no longer providing highly available services because a local failure has been detected or an administrative command has moved the node to this state. Also, if a node cannot move to normal state during the joining phase, it moves to this state.
controlled failback	This node is no longer providing highly available services, but it is monitoring the other node in the cluster and the services it is providing.
error	An unrecoverable failure has occurred.

Figure 1-4 diagrams the node states and the events that govern them.

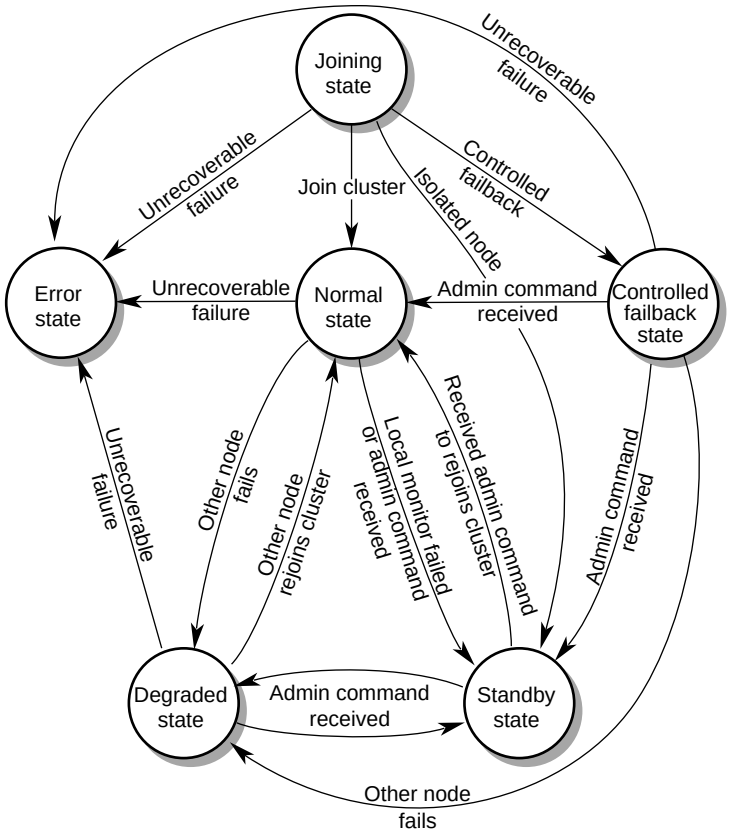


Figure 1-4 IRIS FailSafe Node States and Transitions

Table 1-2 shows the possible combinations of states for two nodes. When the state is listed as (none), it means that IRIS FailSafe software is not running or the node is shut down.

Table 1-2 Possible Combinations of Node States

State of One Node	State of the Other Node	Situation
joining joining joining joining joining	joining normal degraded standby (none)	These are transient state combinations that occur immediately after one or both nodes have been rebooted.
normal	normal	Both nodes are operating normally, providing the highly available services for which they are the primary node.
degraded	standby	The node in degraded state is providing all highly available services. The node in standby state is not providing any highly available services and is not performing monitoring.
degraded	controlled failback	The node in degraded state is providing all highly available services. The node in controlled failback state is not providing any highly available services, but is performing monitoring.
degraded	(none)	The node in degraded state is providing all highly available services while the other node isn't running IRIS FailSafe software or is shut down.
standby	(none)	The node in standby state is running IRIS FailSafe software, but is not providing any highly available services. The other node isn't running IRIS FailSafe software or is shut down.

Overview of Configuring and Testing a New IRIS FailSafe Cluster

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

1. Become familiar with IRIS FailSafe terms by reviewing this chapter and the "Glossary" at the end of this guide.
2. Plan the configuration of highly available applications and services on the cluster using Chapter 2, "Planning IRIS FailSafe Configuration."

3. Perform various administrative tasks, including the installation of prerequisite software, that are required by IRIS FailSafe. The instructions are in Chapter 3, “Configuring Nodes for IRIS FailSafe.”
4. Prepare the IRIS FailSafe configuration file as explained in Chapter 4, “Creating the IRIS FailSafe Configuration File.”
5. Test the IRIS FailSafe system in three phases: test individual components prior to starting IRIS FailSafe software, test normal operation of the IRIS FailSafe system, and simulate failures to test the operation of the system after a failure occurs. The instructions are in Chapter 5, “Testing IRIS FailSafe Configuration.”

Overview of IRIS FailSafe 1.1 for IRIS FailSafe 1.0 Users

The subsections below explain the differences between Release 1.0 of IRIS FailSafe and Release 1.1 of IRIS FailSafe. The final subsection describes the procedure for upgrading an IRIS FailSafe system from 1.0 to 1.1.

New IP Address Failover Model

IRIS FailSafe 1.1 uses a substantially different model for IP address failover that is changed in several ways and provides new features:

- IRIS FailSafe 1.1 supports the failover of multiple network interfaces in a node, not just a designated primary interface.
- There are no designated primary and secondary network interfaces or primary and secondary IP addresses in a node.
- Any network interface in a node can act as primary interface and as a secondary interface for an IP address (if MAC address impersonation is not required).
- A dual-active cluster can be configured even if each node has just one public network interface. (If MAC address impersonation is required, two public interfaces are required on each node.)
- IRIS FailSafe 1.1 can failover multiple IP addresses configured to a single network interface using the IP aliasing mechanism. These IP addresses are called *high availability IP addresses*.
- The IP addresses configured to several network interfaces on one node can be failed over to a single network interface on the other node.

- Each network interface in a cluster has an IP address that is configured at the boot time and is not failed over. These IP addresses are called *fixed IP addresses*.
- The IRIS FailSafe 1.0 restriction that the IP name used for the private network between the nodes be the hostname of the node has been removed.

New Controlled Failback Feature

IRIS FailSafe 1.1 has an optional new feature called *controlled failback* that gives administrators more control over the behavior of an IRIS FailSafe cluster after a failover.

In IRIS FailSafe 1.0, when a remote monitor failure or a heartbeat failure is detected, the node with the failure is rebooted and the other node takes over all the highly available services. The failed node boots up and takes back its services. Thus, the services provided by the node with the failure have moved twice, and the clients using the services see two interruptions in the service.

In IRIS FailSafe 1.1, if the controlled failback feature is enabled for the node, the second interruption in service can be prevented. The rebooted node comes up and does not take back its services. It is now in a state called controlled failback. It monitors the highly available services running in the other node, providing a backup for the highly available services. Using an administrative command, the node in the controlled failback state can be moved to normal state and thus resume providing the highly available services for which it is the primary node.

New Interface Agent Replaces Remote Monitoring Scripts

A new process called an interface agent (*/usr/etc/ha_ifa*) has been added to IRIS FailSafe 1.1. It monitors all the network interfaces that have high availability IP addresses in the cluster. The interface agent makes the monitoring of the highly available services on the other node in the cluster unnecessary. So, remote monitoring scripts are no longer used in general. (An exception is IRIS FailSafe 1.1 systems using IRIS FailSafe 1.0 configuration files and the IRIS FailSafe NFS or IRIS FailSafe Web options. They continue to use remote monitoring scripts.)

Support for Oracle, INFORMIX, and Sybase Database Failover Added

New IRIS FailSafe option products support highly available Oracle, INFORMIX and Sybase database servers. All the database servers are monitored using database agents (*/usr/etc/ha_sybs* for Sybase databases, */usr/etc/ha_ifmx* for INFORMIX databases, and */usr/etc/ha_orcl* for Oracle databases).

New Heartbeat Daemon

IRIS FailSafe 1.1 uses the new process */usr/etc/ha_hbeat* to receive heartbeat messages sent by the application monitor on the other node. In IRIS FailSafe 1.0, heartbeat messages were sent and received by the application monitor (*/usr/etc/ha_appmon*).

In IRIS FailSafe 1.1, all configurations can use the public network to send and receive heartbeat messages in case the private network fails. This feature was available only in dual-active configurations in IRIS FailSafe 1.0.

New Configuration File Format

The format of the IRIS FailSafe configuration file, the file that contains configuration and monitoring frequency information, has changed in IRIS FailSafe 1.1. The blocks required in the configuration file and their contents are different.

IRIS FailSafe 1.1 can work with IRIS FailSafe 1.0 configuration files. However, using IRIS FailSafe 1.1 with a 1.0 configuration file provides only bug fixes and an improvement in heartbeat failure detection time. It does not provide the new features of IRIS FailSafe 1.1. To use the new features of IRIS FailSafe 1.1, you must replace the 1.0 configuration file with one in the IRIS FailSafe 1.1 format. You cannot simply add the new 1.1 configuration file blocks to a 1.0 configuration file. Upgrading 1.0 configuration files to the IRIS FailSafe 1.1 format is recommended. Chapter 4, “Creating the IRIS FailSafe Configuration File,” explains how to create a 1.1 configuration file.

Overview of Upgrading an IRIS FailSafe Cluster From Release 1.0 to Release 1.1

Follow this general procedure when you are upgrading an IRIS FailSafe cluster from Release 1.0 to Release 1.1:

1. Review the IRIS FailSafe concepts and the information about differences between IRIS FailSafe 1.0 and 1.1 presented in this chapter.
2. Review Chapter 2, “Planning IRIS FailSafe Configuration,” and perform any planning activities required, such as interface configuration, because of the changes from IRIS FailSafe 1.0 to 1.1.
3. Review Chapter 3, “Configuring Nodes for IRIS FailSafe,” and perform any necessary tasks, such as installing the IRIS FailSafe 1.1 software.
4. If you decide to switch to a Release 1.1 configuration file, prepare one as described in Chapter 4, “Creating the IRIS FailSafe Configuration File.”
5. Perform the upgrade procedure in Chapter 7, “Upgrading an IRIS FailSafe Cluster,” to start IRIS FailSafe 1.1 on both nodes in the cluster.

Chapter 2

Planning IRIS FailSafe Configuration

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

Introduction to Configuration Planning

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

- What do you plan to use the two nodes for?

  Your answers might include uses such as home directories for users, running particular applications, World Wide Web browsing, file server, and Netscape server.

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

  The IRIS FailSafe NFS option enables you to provide exported NFS filesystems as highly available services. Similarly, the IRIS FailSafe Web option is used for Netscape Commerce and Communications servers, the IRIS FailSafe INFORMIX option is used for Informix databases, the IRIS FailSafe Oracle option is used for Oracle databases, and the IRIS FailSafe Sybase option is used for Sybase databases.

To offer other applications as highly available services, you must develop a set of shell scripts that provide switch over and switch back functionality. Developing these scripts is described in the *IRIS FailSafe Programmer's Guide*.

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

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

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

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

- Are the shared disks going to be part of a CHALLENGE RAID storage system or are they going to be disks in CHALLENGE Vaults that have plexed XLV logical volumes on them?

  Shared disks must be part of a CHALLENGE RAID storage system or in CHALLENGE Vaults with plexed XLV logical volumes on them.

- Will the shared disks be used as raw XLV logical volumes or XLV logical volumes with XFS filesystems on them?

  XLV logical volumes are required by IRIS FailSafe; filesystems must be XFS filesystems. The choice of volumes or filesystems depends on the application that is going to use the disk space.

- Which interfaces on primary nodes will be used by clients of highly available services?

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

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

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

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

  Each public interface that is used by clients of highly available services must have at least two IP addresses, one fixed IP address and one high availability address. The fixed IP address is not available to clients after a failover.

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

  Every high availability IP address used by a highly available service must be mapped to a pair of interfaces, one on the primary node and one on the backup node. The high availability IP addresses are failed over from the interface in the primary node to the interface in the backup node.

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

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

Two public interfaces are available on each node, ec0 and ec2. Both are used, ec0 by clients of the NFS filesystems and ec2 by clients of the Netscape servers. Each interface has two IP addresses, one fixed and one high availability. The high availability IP address is publicized for use by clients. In the event of failover, the high availability IP address on ec0 fails over to the ec0 interface on the other node, and the IP address on ec2 fails over to the other ec2 interface.

The following sections help you answer the questions above, make additional configuration decisions required by IRIS FailSafe, and collect the information you need to perform the configuration tasks described in Chapter 3, “Configuring Nodes for IRIS FailSafe,” and Chapter 4, “Creating the IRIS FailSafe Configuration File.”

NFS Filesystem Configuration

The first subsection below describes NFS filesystem issues that must be considered when planning an IRIS FailSafe system. The second subsection gives an example of an NFS filesystem configuration on an IRIS FailSafe system. The third subsection explains the aspects of the configuration that must be specified as parameters in the IRIS FailSafe configuration file.

Planning NFS Filesystems

The IRIS FailSafe NFS option enables IRIS FailSafe to provide failover protection for

- filesystems that have been configured to be exported by the *exportfs* command
- NFS file-locking information, as supported by *rpc.lockd/rpc.statd*

In an IRIS FailSafe cluster, one or both nodes can export NFS filesystems. If a node that exports NFS filesystems fails, the other node provides backup service.

When the NFS server on one node fails, the surviving node must take over NFS file-locking information from the failed node. IRIS FailSafe does this by storing the NFS locking state for each exported filesystem. The information for each node is stored on that node in a single directory that is in one of the filesystems on the shared disk. This directory is called the statmon directory and its name must be *statmon*.

Do not place NFS filesystems on shared disks in the */etc/exports* file. IRIS FailSafe exports these filesystems only after making sure that the other node does not have these filesystems exported.

Note: When clients are actively writing to a FailSafe NFS filesystem during filesystem failover, data corruption can occur unless filesystems are exported with the mode *wsync*. This mode requires that local mounts of the XFS filesystems use the *wsync* mount mode as well. Using *wsync* affects performance considerably. See the section “Wsync Filesystem Options” in Chapter 4 for more information.

Example NFS Filesystem Configuration

For example, if the node named stocks exports the NFS filesystem */shared1* and the node named bonds exports the NFS filesystems */shared2* and */shared3*, the two directories */shared1/statmon* and */shared2/statmon* should be created to hold NFS file-locking information. The directory */shared1/statmon* stores NFS lock information that allows bonds to recover NFS locks for the */shared1* filesystem if stocks fails. The directory */shared2/statmon* stores NFS lock information that allows stocks to recover NFS locks for */shared2* and */shared3* if bonds fails.

Configuration Parameters for NFS FIlesystems

Table 2-1 lists the name of the parameter that specifies the statmon directory.

Table 2-1 NFS Configuration Parameters

Parameter	Value for stocks	Value for bonds	Comment
statmon-dir	/shared1/statmon	/shared2/statmon	The value is a directory called *statmon* on any NFS filesystem on a shared disk. It is used to store NFS locks for each node.

The procedure for configuring NFS filesystems for IRIS FailSafe is described in the section “Configuring NFS Filesystems” in Chapter 3.

Netscape Server Configuration

Configuration of one or more highly available Netscape servers on an IRIS FailSafe cluster requires the use of the IRIS FailSafe Web option. One subsection below discusses several choices you must make when configuring Netscape servers. The remaining subsections describe two example Netscape configurations and the configuration parameters for these examples.

Note: The IRIS FailSafe Web software supports the Netscape Communications Server and the Netscape Commerce Server. For other Web servers, you must write a monitoring script that is similar to */var/ha/actions/ha_web_lmon* and a failover script that is similar to */var/ha/resources/webserver*. For information on creating these scripts see the *IRIS FailSafe Programmer’s Guide*.

Planning Netscape Server Configuration

In configuring one or more Netscape servers on an IRIS FailSafe cluster, you need to consider the locations of these components:

- Netscape server software, which is usually installed in the directory */usr/netscape*
- Netscape configuration information, which is installed in a directory called the server root
- Log files for Netscape usage accounting

  Most Netscape sites run Netscape usage accounting. One common way of accounting for Netscape usage is to parse the access log files. These files are normally held within the *httpd* configuration directory tree (for example, */usr/ns-home/httpd-80/logs*), which is within the server root.
- Documents (HTML pages), which are stored in a directory called the document root

Failover of a Netscape server has these requirements:

- The Netscape server software is available on each node acting as a Netscape server.
- Each Netscape server's configuration information is available on each node.
- Log files must fail over.
- A document root must be available to each Netscape server on each node.
- The IP addresses associated with each Netscape server must fail over.
- If the log files, the document root, the server configuration, or the server software are kept on a shared disk, they must also fail over.
- The primary and backup nodes of the Netscape server, of the server's high availability IP address, and of the shared disks used by the server must be the same.

Note: The accesses make by the IRIS FailSafe monitoring scripts are recorded in the log file records. Therefore, you must subtract these accesses from the total number of hits to get an accurate count. An almost-accurate way to do this is to eliminate all accesses made from both nodes in the cluster. Doing so also eliminates accesses made by any users on the nodes in the cluster, but because these nodes are servers, removing these accesses should present no serious problems.

The locations of Netscape server software, server configuration information, logs, and document root present a lots of possibilities. For the simple case of one or more Netscape servers on one node in an IRIS FailSafe cluster (an active/backup configuration), the easiest way to configure the cluster is to put each of these components on one or more shared disks.

When configuring a Netscape server on each node in an IRIS FailSafe cluster (a dual-active configuration), the Netscape server software and configuration information must be duplicated on each node. The log files must be on shared disks, with each node being the primary node for a shared disk that contains log files. For documents, there are two possibilities:

- The servers on both nodes serve the same documents.

  You must make two copies of the document root on non-shared disks because a filesystem cannot be mounted simultaneously on both nodes. Since each node serves the same documents, it is unnecessary to distinguish requests for the different servers. Thus, you do not need to specify a server's IP address during Netscape server configuration. A single server can handle all Netscape requests, even in case of failover.

  If you use this configuration, be certain that the contents of the document root do not change. For example, if the Netscape server is used to collect input from clients that is stored in the document root, the document roots on the two nodes are no longer identical.

- A different set of documents is served from each node.

  When different documents are being served from each node, for example, stocks is serving pages about the stock market and bonds is serving pages about the bond market, each set of documents must be on a shared disk, so they can be failed over. If stocks fails, bonds serves both stock and bond market pages.

  In this configuration, the Netscape servers must each have a unique IP alias in order to distinguish requests to the two Netscape servers. In normal operation, only one server is active on each node. However, you must configure all servers to cover the case in which a node fails, and both Netscape servers run on the surviving node.

  When installing a Netscape server, you specify the root directory in which this information is stored. The default is */usr/ns-home*. Each server has a subdirectory under this directory, corresponding to its IP address; for example, the server root directory for the bonds Netscape server is */usr/ns-home/httpd-80.190.0.2.4*.

In a degraded configuration with both Netscape servers on the surviving node, this arrangement ensures that all requests to stocks are sent to its server, even though it is running on the same node as bonds. The two Netscape servers on each node must have different document roots. Each Netscape server's document root is on a separate filesystem on a shared disk. For example, stocks normally uses */shared1*, and bonds normally uses */shared2*.

If the Netscape server fails on one node, the surviving node first takes over the IP alias and the shared disk containing the failed node's Netscape server root and document root, then runs the appropriate */etc/init.d/ns_httpd* startup script. This script starts all Netscape servers, including those that are already running (such as the Netscape server on the surviving node). Ignore the warnings that these Netscape servers are already up.

In some cases, multiple Netscape servers are configured to provide the same content. The client requests are distributed between the multiple servers using a round robin DNS mechanism. In these cases, the nodes serving Web pages have identical Netscape server configurations—the Netscape server software, Netscape configuration information, netscape software, log files and document root are the same and are kept in the same location on node. These Netscape servers have duplicates of all files on local disks (no shared disks). IRIS FailSafe can be used by these Netscape servers for IP address failover and for monitoring the Netscape server processes on each node.

In this configuration, when there is failure in a node, the backup node takes over the IP address. The backup node does not restart the Netscape server because it already has Netscape server processes running. When the failed node returns to normal state, the Netscape server is restarted on the failed node. This special case of Netscape server configuration is enabled by setting the httpd-restart parameter to **true** in the webserver block of the IRIS FailSafe configuration file.

Active/Backup Netscape Server Example

The first subsection below describes an active/backup Netscape server configuration—one node is used as a Netscape server and the other node serves as a backup in case the first node fails. The second subsection explains the aspects of the configuration that must be specified as parameters in the IRIS FailSafe configuration file.

Example Active/Backup Netscape Server Configuration

This example is an active/backup configuration with two Netscape Commerce servers on the primary node xfs-ha1. The backup node is called xfs-ha2. Two IP aliases for a network interface on xfs-ha1 are 190.0.2.3 and 190.0.2.5. One Netscape server uses port number 80 and has its server root in */shared/httpd-80*. The other Netscape server uses port number 90 and has its server root in */shared/httpd-90*.190.0.2.5. The script used to start and stop the Netscape servers is */etc/init.d/ns_commerce* (rather than the default */etc/init.d/ns_httpd*). The configuration file */etc/config/ns_commerce.options* (instead of the default */etc/config/ns_httpd.options*) is used to start the Netscape servers.

Note: Although this example has two Netscape servers on a single node, listening on ports 80 and 90, a typical configuration is a single Netscape server listening to a single port. This example is included to illustrate how to specify two Netscape servers on a single node in the IRIS FailSafe configuration file.

Configuration Parameters for Active/Backup Example

Table 2-2 shows the various labels, section names, and parameters used in the configuration file for the active/backup example in the previous section.

Table 2-2 Netscape Configuration Parameters (Active/Backup Configuration)

Label, Section, or Parameter	Value	Comments
label	webxfs-ha1	A unique name.
server-node	xfs-ha1	The primary node.
backup-node	xfs-ha2	The backup node.
httpd-script	/etc/init.d/ns_commerce	The startup and shutdown script for the server.
httpd-options-file	ns_commerce.options	The configuration file for the Netscape server.
webserver-num	2	The number of Netscape servers on the primary node.
section name for the first server	web-config1	This value is web-config followed by a digit.

Table 2-2 (continued) Netscape Configuration Parameters (Active/Backup Configuration)

Label, Section, or Parameter	Value	Comments
ip-address (for the first server)	190.0.2.3	Any IP alias for the primary node in *X.X.X.X* form.
port-num (for the first server)	80	The port number used by this server.
httpd-dir (for the first server)	/shared/httpd-80	Server root for the first server.
section name for the second server	web-config2	This value is web-config followed by a digit that is different from the other server.
ip-address (for the second server)	190.0.2.3	Any IP alias for the primary node in *X.X.X.X* form.
port-num (for the second server)	90	The port number used by this server.
httpd-dir (for the second server)	/shared/httpd-90.190.0.2.5	Server root for the second server.

The section "Configuring a Netscape Server" in Chapter 3 describes the procedure for configuring Netscape.

Dual-Active Netscape Server Example

The first subsection below describes a dual-active Netscape server configuration—each node serves as a Netscape server during normal operation. If one node fails, the other node takes over the server duties of the failed node. The second subsection explains the aspects of the configuration that must be specified as parameters in the IRIS FailSafe configuration file.

Example Dual-Active Netscape Server Configuration

This example is a dual-active configuration with one Netscape server on each of the nodes xfs-ha1 and xfs-ha2. The default startup and shutdown script and configuration file are used. xfs-ha1 has the IP alias 190.0.2.3, and xfs-ha2 has the IP alias 190.0.2.4. Each server uses port 80. The server root on xfs-ha1 is */shared.stocks/httpd-80*. The server root on xfs-ha2 is */shared.bonds/httpd-80*.

Configuration Parameters for Dual-Active Example

Table 2-3 shows the various labels, section names, and parameters used in the configuration file for the dual-active Netscape server example in the previous section.

Table 2-3 Netscape Configuration Parameters (Dual-Active Configuration)

Label, Section, or Parameter	Value for xfs-ha1	Value for xfs-ha2	Comments
label	webxfs-ha1	webxfs-ha2	A unique name.
server-node	xfs-ha1	xfs-ha2	The primary nodes.
backup-node	xfs-ha2	xfs-ha1	The backup nodes.
webserver-num	1	1	The number of Netscape servers on the primary node.
section name for the first server	web-config1	web-config1	This value is web-config followed by a digit.
ip-address	190.0.2.3	190.0.2.4	An IP alias for the primary node.
port-num	80	80	The port number.
httpd-dir	/shared.stocks/ httpd-80	/shared.bonds/ httpd-80	The server root.

The section “Configuring a Netscape Server” in Chapter 3 describes the procedure for configuring Netscape.

Disk Configuration

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

Planning Disk Configuration

For each disk in an IRIS FailSafe cluster, you must choose whether to make it a shared disk, which enables it to be failed over, or a non-shared disk. Non-shared disks are not failed over. Choosing to make a disk shared or non-shared depends on the needs of the highly available services that use the disk.

Both nodes in an IRIS FailSafe cluster must follow these requirements:

- The system disk must be a non-shared disk.
- The IRIS FailSafe software, in particular the directory */var/ha*, must be on a non-shared disk.

Each highly available service has requirements about the location of data associated with the service:

- Some data must be placed on non-shared disks
- Some data must not be placed on shared disks
- Some data can be on shared or non-shared disks

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

- A non-shared disk on each node
- A shared disk in an active/backup cluster
- A shared disk in a dual-active cluster

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

An IRIS cluster can contain a combination of the basic disk configurations listed above.

Figure 2-1 shows two nodes in an IRIS FailSafe cluster, each of which has a non-shared disk. When non-shared disks are used by highly available applications, the data required by those applications must be duplicated on non-shared disks on both nodes. When a failover occurs, IP aliases fail over. The data that was originally available on the failed node is still available from the surviving node by using the IP alias to access it.

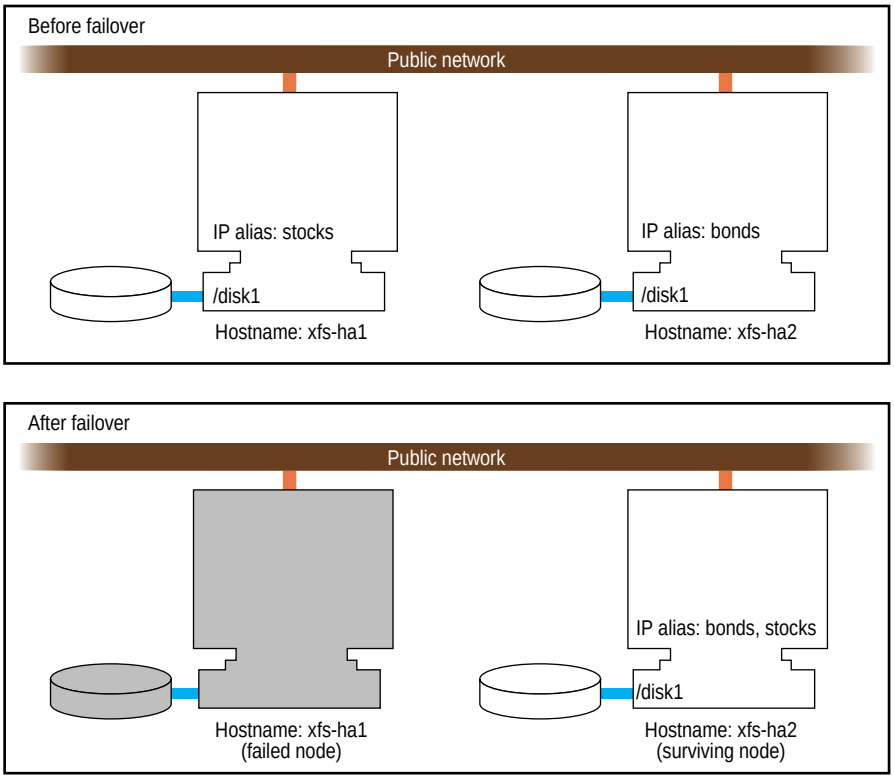


Figure 2-1 Non-Shared Disk Configuration and Failover

Figure 2-2 shows a shared disk in an active/backup configuration. In this configuration, the disk has a *primary node*, which is the node that accesses the disk prior to a failover. It is shown by a solid line connection. The backup node, which accesses the disk after a failover, is shown by a dotted line. Thus, the disk is shared between the nodes. In an active/backup cluster, all shared disks have the same primary node. The backup node doesn't run any highly available applications until a failover occurs.

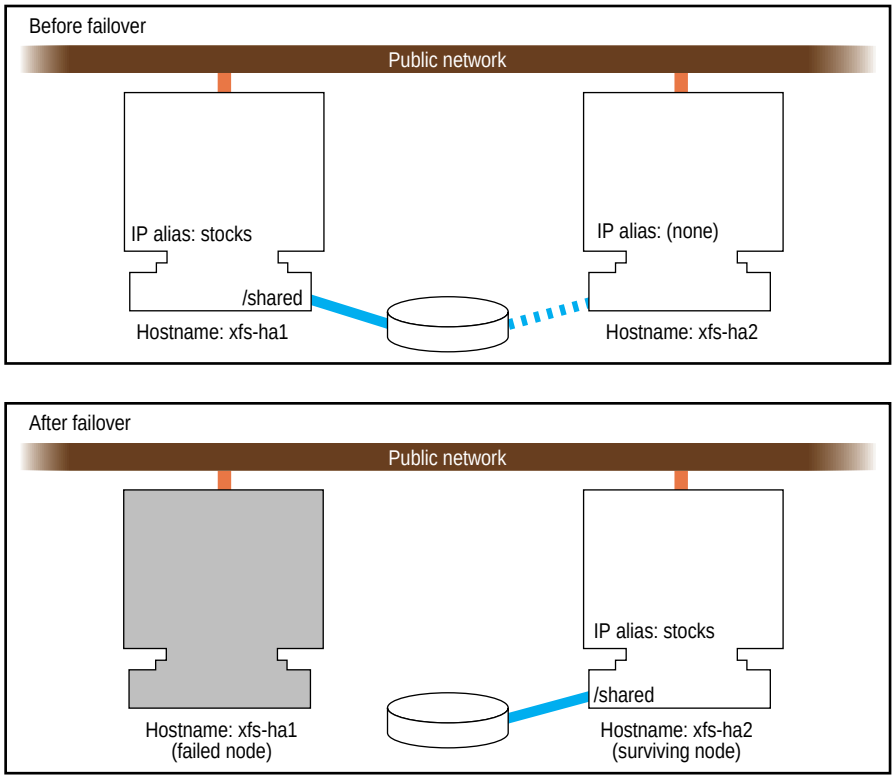


Figure 2-2 Shared Disk Configuration for Active/Backup Use

Figure 2-3 shows two shared disks in a dual-active cluster. In this case, each node serves as a primary node for one shared disk. The solid line connections show the connection to the primary node prior to failover. The dotted lines show the connections to the backup nodes. After a failover, the surviving node accesses all shared disks.

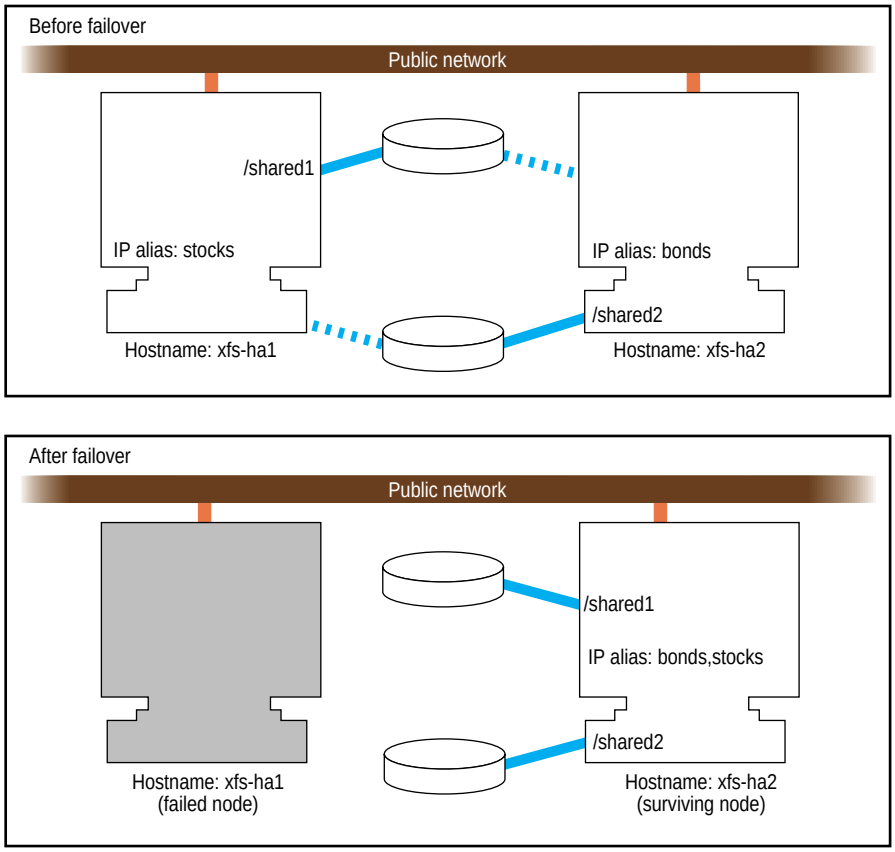


Figure 2-3 Shared Disk Configuration For Dual-Active Use

Other sections in this chapter and similar sections in the *IRIS FailSafe Oracle Administrator's Guide*, *IRIS FailSafe INFORMIX Administrator's Guide*, Sybase, and *IRIS FailSafe Sybase Administrator's Guide* provide more specific information about choosing between shared and non-shared disks for various types of data associated with each highly available service.

The choice of an active/backup cluster or a dual-active cluster is based on your preference: in an active/backup cluster, one node serves only as a "hot spare" for highly available applications on the other node. In a dual-active cluster, the highly available applications, and therefore the load, are distributed between the nodes.

Configuration Parameters for Disks

There are no configuration parameters associated with non-shared disks. They are not specified in the IRIS FailSafe configuration file. Only shared disks (actually, the XLV logical volumes on shared disks) are specified in the configuration file. See the section "Configuration Parameters for Logical Volumes" in this chapter for details.

Logical Volume Configuration

The first subsection below describes logical volume issues that must be considered when planning an IRIS FailSafe system. The second subsection gives an example of an XLV logical volume configuration on an IRIS FailSafe system. The third subsection explains the aspects of the configuration that must be specified as parameters in the IRIS FailSafe configuration file.

Planning Logical Volumes

All shared disks must have XLV logical volumes on them. You can work with XLV logical volumes on shared disks as you would work with other disks. However, for correct operation of the IRIS FailSafe configuration, you must follow these rules:

- All data that is used by highly available applications on shared disks must be stored in XLV logical volumes.
- XLV allows multiple volumes to be created on the same physical disk. In an IRIS FailSafe environment, if you create more than one volume on a single disk, they must all be owned by the same node. For example, if a disk has two partitions that are part of two XLV volumes, both XLV volumes must be owned by the same node. (See the sections "Creating XLV Logical Volumes and XFS Filesystems" in Chapter 3 and "Trouble With Volumes" in Appendix B for more information about XLV volume ownership.)

- Each disk in a vault or RAID LUN must be owned by one of the IRIS FailSafe nodes. (A disk is owned by the node that is its primary node.) Therefore, you must divide the vault disks and RAID LUNs into two sets, one for each node. If you create multiple volumes on a vault disk or RAID LUN, all those volumes must be owned by one node.
- Do not access a shared XLV volume from both nodes simultaneously. Doing so causes data corruption.

The IRIS FailSafe software relies on the XLV naming scheme to operate correctly. A fully qualified XLV volume name is */pathname/volname* or */pathname/nodename.volname*. The components are:

- */pathname*, which is /dev/dsk/xlv or /dev/rdsk/xlv
- *nodename*, which by default is the same as the hostname of the node the volume was created on
- *volname*, a name specified when the volume was created; this component is commonly used when a volume is to be operated on by any of the XLV tools

For example, if volume *vol1* is created on node ha1 using disk partitions located on a shared disk, the raw character device name for the assembled volume is */dev/rdsk/xlv/vol1*. On the peer ha2, however, the same raw character volume appears as */dev/rdsk/xlv/ha1.vol1*, where *ha1* is the nodename component, and *vol1* is the volname component. As can be seen from this example, when the *nodename* component is the same as the local hostname, it does not appear as part of the device node name.

One *nodename* is stored in each disk or LUN volume header. This is why all volumes with volume elements on any single disk must have the same *nodename* component. If this rule is not followed, the IRIS FailSafe software does not operate correctly.

The IRIS FailSafe software modifies the *nodename* component of the volume header as volumes are transferred between nodes during failover and recovery operations. This is important because *xlv_assemble* assembles only those volumes whose *nodename* matches the local hostname. Some of the other XLV utilities allow you to see (and modify) all volumes, regardless of which node owns them.

If you use XLV logical volumes as raw volumes (no filesystem) for storing database data, the database system may require that the device names (in */dev/rdsk/xlv* and */dev/dsk/xlv*) have specific owners, groups, and modes. See the documentation provided by the database vendor to determine if the XLV logical volume device names must have owners, groups, and modes that are different from the default values (the default owner, group, and mode for XLV logical volumes are root, sys, and 0600).

Example Logical Volume Configuration

As an example of XLV logical volume configuration, say that you have these logical volumes on four disks that we will call disk 1 through disk 5:

- A logical volume called */dev/dsk/xlv/volA* (volume A) that is comprised of disk 1 and a portion of disk 2.
- A logical volume called */dev/dsk/xlv/volB* (volume B) that is comprised of the remainder of disk 2 and disk 3.
- A logical volume called */dev/dsk/xlv/volC* (volume C) that is comprised of disks 4 and 5.

Volumes A and B must have the same primary node (stocks, for example), because they share a disk. Volume C could have either node as its primary node. For this example, say that bonds is the primary node, which makes this a dual-active configuration.

Configuration Parameters for Logical Volumes

Configuration parameters for XLV logical volumes list

- The primary node and backup node for each volume.
- The nodes that act as primary nodes (one node for active / backup clusters and two nodes for dual-active clusters).
- The owner, group, and mode of the device filename for the volume.

Table 2-4 lists a label and parameters for individual logical volumes.

Table 2-4 XLV Logical Volume Configuration Parameters

Label or Parameter	Value for Volume A	Value for Volume B	Value for Volume C	Comments
volume label	volA	volB	volC	Each volume is given a label to identify it in the configuration file.
server-node	stocks	stocks	bonds	The primary node.
backup-node	bonds	bonds	stocks	The backup node.
devname	/dev/dsk/xlv/volA	/dev/dsk/xlv/volB	/dev/dsk/xlv/volB	The device name of the XLV logical volume.
devname-owner	root	root	root	The owner of the device name.
devname-group	sys	sys	root	The group of the device name.
devname-mode	0600	0600	0600	The mode of the device name.

The server-node parameter is used in another context as well. Each node specified as a server-node in Table 2-4 is also specified as a server-node in a block of the configuration file called “application-class volumes.” This block lists each node that is a primary node (one node for active/backup clusters and two nodes for dual-active clusters).

See the section “Creating XLV Logical Volumes and XFS Filesystems” in Chapter 3 for information about creating XLV logical volumes.

Filesystem Configuration

The first subsection below describes filesystem issues that must be considered when planning an IRIS FailSafe system. The second subsection gives an example of an XFS filesystem configuration on an IRIS FailSafe system. The third subsection explains the aspects of the configuration that must be specified as parameters in the IRIS FailSafe configuration file.

Planning Filesystems

The IRIS FailSafe software supports the automatic failover of XFS filesystems on shared disks. Shared disks must be in CHALLENGE Vaults or CHALLENGE RAID storage systems that is shared between two CHALLENGE nodes.

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

- All filesystems to be failed over must be XFS filesystems.
- All filesystems to be failed over must be created on XLV logical volumes on shared disks.
- For availability, filesystems to be failed over in an IRIS FailSafe cluster must be created on either mirrored disks (using the XLV plexing software) or on the CHALLENGE RAID storage system.
- Make filesystems for which this node is primary only on those volumes owned by this node. Create the mount points for these filesystems on both nodes.
- When you set up the various IRIS FailSafe filesystems on each node, make sure that each filesystem uses a different mount point. For example, do not use the mount point */shared* on each node. In the degraded state, in which all filesystems are owned by the surviving node, that node must be able to concurrently service requests for all filesystems. If the mount point directories are different, such as */shared1* and */shared2*, the surviving node can ensure that all NFS requests sent to the failed node are handled properly, even though they are being handled by the surviving node.
- Do not simultaneously mount filesystems on shared disks on both nodes. Doing so causes data corruption. Normally, IRIS FailSafe performs all mounts of filesystems on shared disks. If you manually mount a filesystem on a shared disk, make sure that it is not being used by the other node.
- Do not place filesystems on shared disks in the */etc/fstab* file. IRIS FailSafe mounts these filesystems only after making sure that the other node does not have these filesystems mounted.

Note: When clients are actively writing to a FailSafe NFS filesystem during failover of filesystems, data corruption can occur unless filesystems are exported with the mode *wsync*. This mode requires that local mounts of the XFS filesystems use the *wsync* mount mode as well. Using *wsync* affects performance considerably. See the section “Wsync Filesystem Options” in Chapter 4 for more information.

Example Filesystem Configuration

Continuing with the example configuration from the section "Example Logical Volume Configuration" in this chapter, say that volumes A and B have XFS filesystems on them:

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

Configuration Parameters for Filesystems

Table 2-5 lists a label and configuration parameters for each filesystem.

Table 2-5 Filesystem Configuration Parameters

Label or Parameter	Value for Filesystem A	Value for Filesystem B	Comments
filesystem label	filesysA	filesysB	Each filesystem is given a label to identify it in the configuration file.
mount-point	/sharedA	/sharedB	The filesystem mount point (on the primary and backup nodes).
volume-name	volA	volB	The label of the logical volume on which the filesystem was created.
mode	rw,noauto	rw,noauto,wsync	The modes of the filesystem (identical to the modes specified in */etc/fstab*).

See the section "Creating XLV Logical Volumes and XFS Filesystems" in Chapter 3 for information about creating XFS filesystems.

Network Interface and IP Address Configuration

The first subsection below describes network interface and IP address issues that must be considered when planning an IRIS FailSafe system. The second subsection gives an example of the configuration of network interfaces and IP addresses on an IRIS FailSafe system. The third subsection explains the aspects of the configuration that must be specified as parameters in the IRIS FailSafe configuration file.

Planning Network Interface and IP Address Configuration

Follow these guidelines when planning the configuration of the interfaces to the private network between nodes in a cluster:

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

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

- Each interface has one fixed IP address (fixed IP addresses don’t fail over).
- Each fixed IP address has an IP name, which can be specified in */etc/hosts*.
- On each node, one IP name for a fixed IP address must be the same as the hostname of the node. In other words, the hostname cannot be a high availability IP address.
- Each interface used by clients to access highly available services must have at least one IP alias.
- Making good choices for IP aliases is important; these are the “hostnames” that will be used by users of the highly available services, not the true hostnames of the nodes.
- Make a plan for publicizing the IP aliases to the user community, since users of highly available services must use IP aliases (high availability IP addresses) instead of the output of the *hostname* command. See the section “Educating the User Community About IRIS FailSafe” in Chapter 6 for more information.

Follow this procedure to determine whether MAC address impersonation is required (see the section "Network Interfaces and IP Addresses" in Chapter 1 for information about MAC address impersonation):

1. Configure an IP address on one of the interfaces of a node (*node1*), for example:

   # **ifconfig ec0 inet 190.0.2.1 netmask 0xffffff00 up**

   In this example, 190.0.2.1 is an IP address for *node1*.

2. From a node not in the cluster, *ping* the IP address used in step 1:

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

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

3. Enter this command on node1 to shut down the interface you configured in step 1:

   # **ifconfig ec0 down**

4. On the other node (*node2*), enter this command to move the IP address to *node2*:

   # **ifconfig ec0 inet 190.0.2.1 netmask 0xffffff00 up**

5. From a node not in the cluster, *ping* the IP address from a client:

   # **ping -c 2 190.0.2.1**

   If the *ping* command fails, MAC address impersonation (re-mac'ing) is needed to fail over the IP address.

If MAC address failover is required, all of the IP aliases configured to a single interface must have the same backup interface. MAC address failover requires a dedicated backup interface (an interface that does not have a high availability IP address).

Example Interface and IP Address Configuration

Figure 2-4 shows an example of an IRIS FailSafe cluster configured with one public interface on each node.

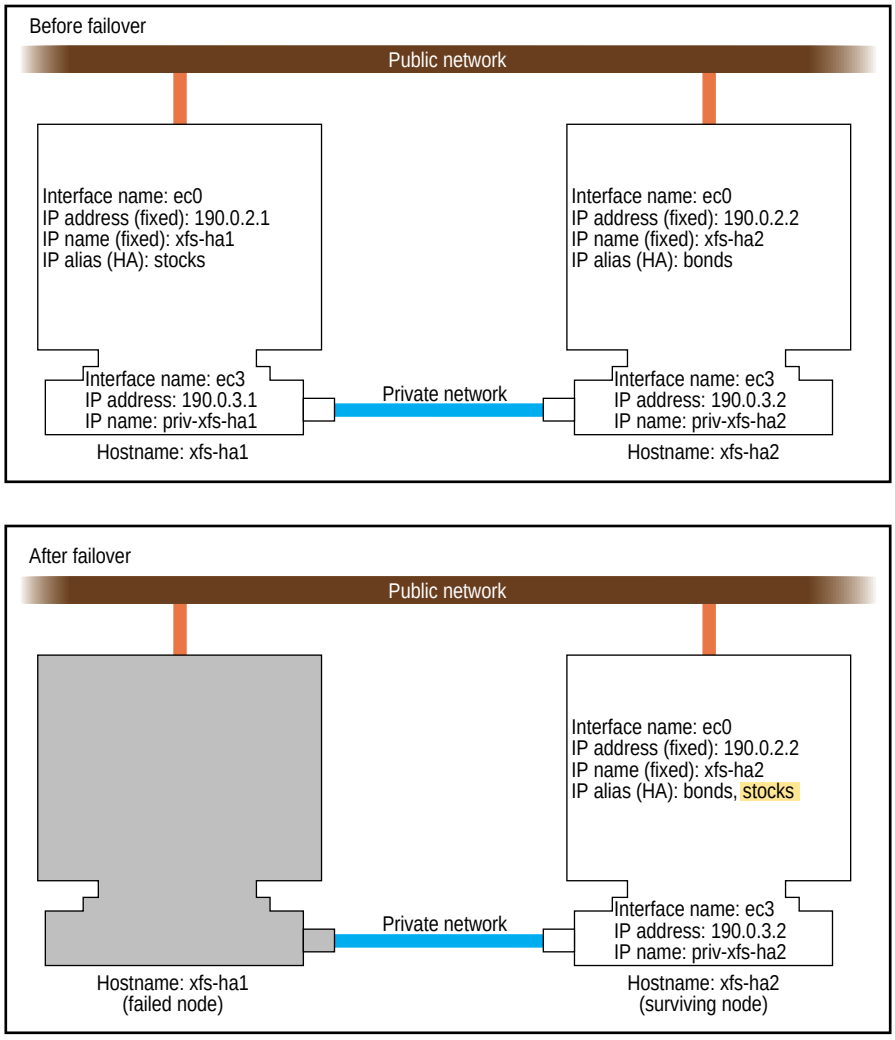


Figure 2-4 Example Interface Configuration

The components of this diagram are as follows:

public network
: One or more Ethernet or FDDI networks.

ec0, ec3
: An interface name, as reported by the Name column in the output of the command *netstat -i*. Examples of interface names are ec0, fxp0, and xpi0.

xfs-ha1, xfs-ha2
: These names are hostnames for the nodes, as reported by the command *hostname*. They are also IP addresses for the ec0 interfaces, as reported by *netstat -i*. These IP addresses are known as *fixed IP addresses*. They do not failover to the other node if their node fails.

stocks, bonds
: These names are IP aliases. They are IP addresses in internet notation, *X.X.X.X*, or IP names. IP aliases are known as *high availability IP addresses*. If their node fails, these IP aliases failover—they become IP addresses for the backup (secondary) interface associated with this interface.

190.0.2.1, 190.0.2.2
: Fixed IP addresses for the interfaces in internet address notation. Each interface has one fixed IP address.

priv-xfs-ha1, priv-xfs-ha2
: These names are IP addresses for the private interfaces as specified in */etc/config/netif.options*. They are fixed IP addresses (they do not failover to the other node if their node fails).

private Ethernet
: The private Ethernet is installed as part of IRIS FailSafe cluster installation. It carries control messages and heartbeat messages between the nodes.

In configuring the interfaces on your IRIS FailSafe cluster, it is very helpful to draw a diagram similar to Figure 2-4 for your configuration.

Configuration Parameters for Interfaces and IP Addresses

The interface names, IP addresses, IP aliases, and so on in Figure 2-4 are used in the IRIS FailSafe configuration file that you develop as you configure an IRIS FailSafe system. Table 2-6 lists the per-node interface-related configuration file parameters and the values they would have for the example shown in Figure 2-4.

Table 2-6 Per-Node Interface Configuration Parameters

Parameter	Value or Label for Node xfs-ha1	Value or Label for Node xfs-ha2	Comments
interface	xfs-ha1-ec0	xfs-ha2-ec0	A unique name for the public interface. A suggestion is a combination of the IP address (name form) and the interface name.
name	ec0	ec0	The public interface name.
ip-address	xfs-ha1	xfs-ha2	The fixed IP address of the public interface.
hb-private-ipname	priv-xfs-ha1	priv-xfs-ha2	The IP address of the private interface.
hb-public-ipname	xfs-ha1	xfs-ha2	The IP address of a public interface that will provide an alternative interface for heartbeat messages if the private network fails.

Table 2-7 lists the information about the high availability IP addresses in the cluster. High availability IP address information is organized in terms of network interface pairs. Each pair of network interfaces consists of an interface called the primary interface and an interface called the secondary interface. The primary interface owns a set (one or more) of high availability IP addresses in normal state. The high availability IP addresses are moved to the secondary interface when there is a failover.

Table 2-7 Per-Interface Configuration Parameters

Parameter	Value or Label for Interface xfs-ha1-ec0	Value or Label for Interface xfs-ha2-ec0	Comments
interface-pair	one	two	Any label.
primary-interface	xfs-ha1-ec0	xfs-ha2-ec0	These match the interface labels from the first line of Table 2-6.
secondary-interface	xfs-ha2-ec0	xfs-ha1-ec0	These match interface labels from the first line of Table 2-6 and are the backup interface for the primary interface of this interface pair.
ip-aliases	stocks	bonds	The IP alias for the primary interface of this interface pair.

If MAC address impersonation is necessary, the parameter re-mac is used. Table 2-8 shows examples of the parameter for the MAC address for each node.

Table 2-8 Interface MAC Address Configuration Parameters

Parameter	Value for Node xfs-ha1	Value for Node xfs-ha2	Comment
re-mac	false	false	Possible values are **true** and **false**. Set to **true** if MAC address impersonation is necessary.
MAC-address	8:0:69:9:31:f8	8:0:69:8:d:1	Use this parameter only if re-mac is set to **true**. The value is the MAC address (physical address) exactly as it is output by the *macconfig* command (it may be missing a leading 0).

The procedure for configuring interfaces and IP addresses for IRIS FailSafe is described in the section “Configuring Interfaces” in Chapter 3.

Serial Port Configuration

A serial cable on each node is connected to the remote power control unit or to the system controller port of the other node. This serial cable is used to reset the other node when necessary. The device name of its serial tty port must be specified in the configuration file. Its device name is /dev/ttyd*port*, where *port* is the serial port number. For example, say that port 2 is being used for the reset serial cable on both nodes. The device name is /dev/ttyd2. More information about serial ports is available in the serial(7) reference page. Table 2-9 shows the configuration parameter that specifies the device name.

Table 2-9 Serial Cable Configuration Parameter

Parameter	Value for Node xfs-ha1	Value for Node xfs-ha2	Comment
reset-tty	/dev/ttyd2	/dev/ttyd2	

The serial tty port used by the serial cable must be configured to have its *getty* process turned off (for more information about *getty*, see the getty(1M) reference page). For the procedure to configure the port, see the section “Configuring the Serial Port” in Chapter 3.

Chapter 3

Configuring Nodes for IRIS FailSafe

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

The major sections in this chapter are as follows:

Overview of Configuring Nodes for IRIS FailSafe

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

1. Install required software as described in the section “Installing Required Software.”
2. Check the setting of two important NVRAM variables as described in the section “Setting NVRAM Variables.”

3. Create the XLV logical volumes and XFS filesystems required by the highly available applications you plan to run on the cluster. See the section “Creating XLV Logical Volumes and XFS Filesystems.”
4. Configure the network interfaces on both nodes using the procedure in the section “Configuring Interfaces.”
5. Configure the serial ports used on each node for the serial connection to the other node by following the procedure in the section “Configuring the Serial Port.”
6. If the IRIS FailSafe NFS option is used, perform necessary NFS configuration as described in the section “Configuring NFS Filesystems.”
7. If the IRIS FailSafe Web option is used, configure Netscape servers as described in the section “Configuring a Netscape Server.”
8. When you are ready configure the nodes so that IRIS FailSafe software starts up when they are rebooted, follow the instructions in the section “Configuring IRIS FailSafe On.”

To complete the configuration of nodes for IRIS FailSafe, you must create and install the IRIS FailSafe configuration file */var/ha/ha.conf* as described in Chapter 4, “Creating the IRIS FailSafe Configuration File.”

Installing Required Software

This section explains software installation requirements and procedures for new IRIS FailSafe clusters. To install a new release of IRIX or IRIS FailSafe software on an IRIS FailSafe cluster already in use, see Chapter 7, “Upgrading an IRIS FailSafe Cluster.”

Several categories of software must be installed on each node in a cluster:

Base system software
: IRIX 5.3 with XFS or IRIX 6.2 is required. NFS software is required if the IRIS FailSafe NFS option is used. Netscape server software is required if the IRIS FailSafe Web option is used. RAID software is required if RAID is being used.

Patches
: IRIS FailSafe requires several base system software patches, which are listed in the IRIS FailSafe base software release notes.

IRIS FailSafe software
: The base IRIS FailSafe option product is required. The IRIS FailSafe NFS option is required for failing over filesystems. The IRIS FailSafe Web option is required for failing over Netscape servers.

Disk plexing licenses
: On clusters that use plexed XLV logical volumes on shared disks, a NetLS license is required on nodes running IRIX 5.3 and a FLEX*lm* license is required on nodes running IRIX 6.2.

The two nodes in a cluster need not be running the same version of IRIX. However, they must run the same version of IRIS FailSafe software.

Follow this procedure for installing software on the nodes in a new cluster:

1. On one node, follow normal software installation procedures and use *inst* to install required subsystems that aren't already installed. The required subsystems are listed in Table 3-1. (See "Related Documentation" for guides about *inst*).

Table 3-1 Required Software Subsystems

Product	Base System Software Subsystems Required by IRIS FailSafe	IRIS FailSafe Subsystems
IRIS FailSafe	For IRIX 5.3: eoe2.sw.xfs, eoe2.sw.xlv, eoe2.sw.xlvplex For IRIX 6.2: eoe.sw.xfs, eoe.sw.xlv, eoe.sw.xlvplex For FDDI (if used): FDDIXPress.sw.FDDIXPress For RAID (if used): raid5.sw.cli	ha.sw.base or ha_IP22.sw.base
IRIS FailSafe NFS	nfs.sw.nfs	ha_nfs.sw.base
IRIS FailSafe Web	ns_httpd.sw.server (Communications server) ns_commerce.sw.server (Commerce server)	ha_www.sw.base

2. On the same node, install required patches. See the release notes for IRIS FailSafe (called ha_base, ha_nfs, and ha_www) for lists of the required patches.
3. Install software and patches on the other node by repeating steps 1 and 2 on the other node.
4. If you are using plexed XLV logical volumes, install a disk plexing license on each node.

   For IRIX 5.3, the license is a NetLS license and it is installed in a non-standard location, */etc/nodelock*. For more information, see the Disk Plexing release notes (called plexing).

   For IRIX 6.2, the license is a FLEX*lm* license and installed in */etc/flexlm/license.dat*. For more information see the *FLEXlm End User Manual*.
5. If you are using plexed XLV logical volumes, verify that the license has been successfully installed by entering these commands on each node:

   ```
   # xlv_mgr
   xlv_mgr> show config
   ...
   Plexing license: present
   ...
   xlv_mgr> quit
   ```

   The line shown about the plexing license indicates that the plexing license has been successfully installed.

Setting NVRAM Variables

During the hardware installation of IRIS FailSafe nodes, two NVRAM variables must be set:

- The boot parameter AutoLoad must be set to **yes**. The IRIS Failsafe software requires the nodes to be automatically booted when they are reset or when the node is powered on.
- The SCSI IDs of the nodes in an IRIS FailSafe cluster, specified by the scsihostid variable, must be different.

You can check the setting of these variables with these commands:

```
# nvram AutoLoad
Y
# nvram scsihostid
0
```

To set these variables, use these commands:

```
# nvram AutoLoad yes
# nvram scsihostid number
```

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

Creating XLV Logical Volumes and XFS Filesystems

In Chapter 2 you planned the XLV logical volumes and XFS filesystems to be used by highly available applications on the cluster. You can create them by following the instructions in one these guides:

- *Getting Started with XFS Filesystems* (IRIX 5.3)
- *IRIX Admin: Disks and Filesystems* (IRIX 6.2)

When you create the XLV logical volumes and XFS filesystems you need, some important points to remember are:

- If the shared disks are not in a CHALLENGE RAID storage system, plexed XLV logical volumes should be created.
- Each XLV logical volume must be owned by the same node that is the primary node for the highly available applications that use the logical volume (see "Planning Logical Volumes" in Chapter 2). To simplify the management of the *nodename*s (owners) of volumes on shared disks, follow these recommendations:
  - Work with the volumes on a shared disk from only one node in the cluster.
  - After you initially create all the volumes on one node, you can then selectively change the *nodename* to the other node afterwards using *xlv_mgr*.
- If the XLV logical volumes you create are used as raw volumes (no filesystem) for storing database data, the database system may require that the device names (in */dev/rdsk/xlv* and */dev/dsk/xlv*) have specific owners, groups, and modes. If this is the case (see the documentation provided by the database vendor), use the *chown* and *chmod* commands (see the chown(1) and chmod(1) reference pages) to set the owner, group, and mode as required.
- No filesystem entries are made in */etc/fstab* for XFS filesystems on shared disks; IRIS FailSafe software mounts the filesystems on shared disks. However, to simplify system administration, consider adding comments to */etc/fstab* that list the XFS filesystems configured for IRIS FailSafe. Thus, a system administrator who sees mounted IRIS FailSafe filesystems in the output of the *df* command and looks for the filesystems in the */etc/fstab* file will learn that they are filesystems managed by IRIS FailSafe.
- Be sure to create the mount point directory for each filesystem on both nodes.

Configuring Interfaces

The procedure in this section describes how to configure the public and private interfaces on the nodes in an IRIS FailSafe cluster. The example shown in Figure 3-1 is used in the procedure.

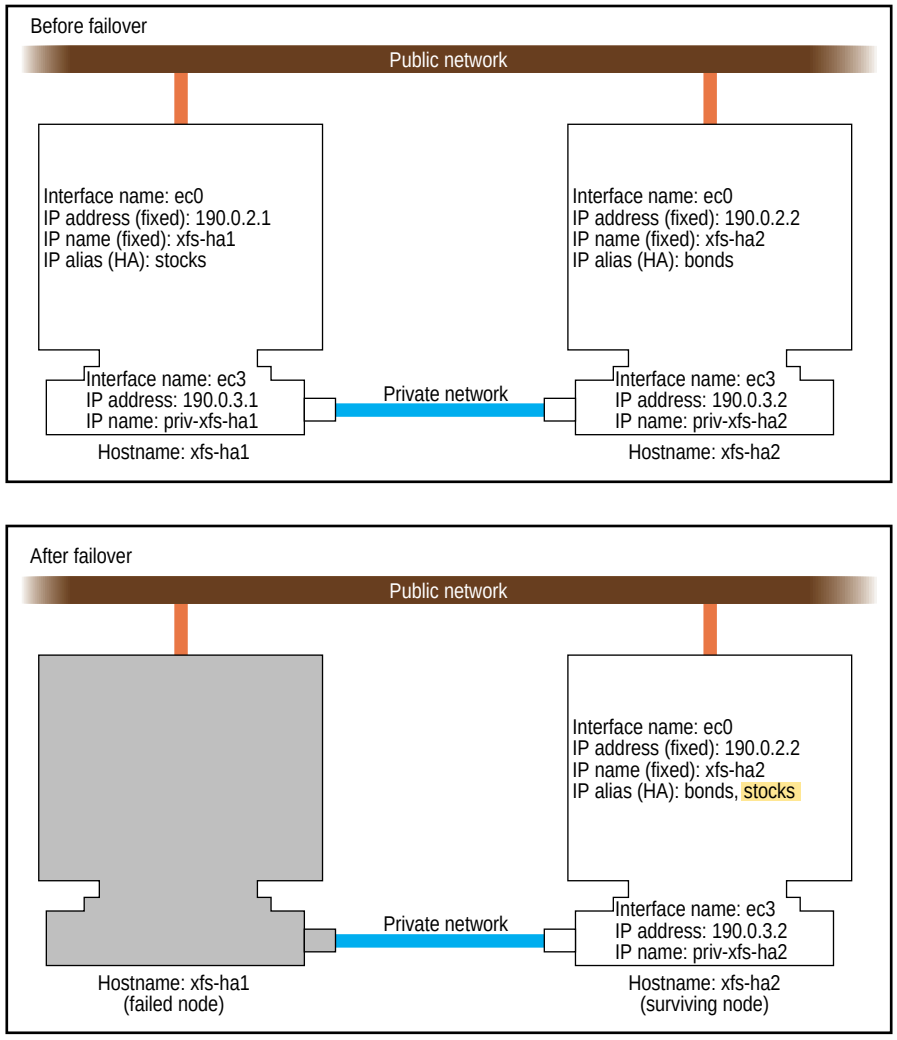


Figure 3-1 Example Interface Configuration

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

   For example:

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

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

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

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

   address is an IP address, IP name, or IP alias. If *ypmatch* reports that *address* doesn't match, it must be added to the NIS database.

4. On one node, add that node's public and private interfaces and their fixed IP addresses to the file */etc/config/netif.options* (high availability IP addresses are not added to the *netif.options* file).

   For the example in Figure 3-1, the public interface name and IP address lines are:

   ```
   if1name=ec0
   if1addr=$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:

   ```
   if2name=
   if2addr=
   ```

   In the example, the private interface name and IP address are:

   ```
   if3name=ec3
   if3addr=priv-$HOSTNAME
   ```

   The private interface IP address in this example, priv-$HOSTNAME, is an alias for an IP address that appears in */etc/hosts*.

5. If there are more than eight interfaces on the node, change the value of `if_num` to the number of interfaces. For less than eight interfaces (as in the example in Figure 3-1) the line looks like this:

   ```
   if_num=8
   ```

6. Repeat steps 4 and 5 on the other node.

7. Edit the file */etc/config/routed.options* on each node so that the routes are not shown over the private network (routing is turned off). The content of */etc/config/routed.options* should be:

   ```
   -h -q
   ```

   These options are required for IRIS FailSafe to function correctly.

8. Verify that IRIS FailSafe is *chkconfig*'d **off** on each node:

   ```
   # chkconfig
           Flag                State
           ====                =====
           ...
           failsafe            off
           ...
   ```

   If *failsafe* is **on** on either node, enter this command on that node:

   ```
   # chkconfig failsafe off
   ```

9. Reboot both nodes to put the network configuration into effect.

10. To configure your e-mail interface so that you can receive notification of cluster transitions, set up an e-mail alias on each node that includes at least one user outside the IRIS FailSafe cluster and an alias on the other node that can direct notification to an outside user. For example, in */usr/lib/aliases* on each node, add

    ```
    fsafe_admin:outside,outside@xfs-ha1.company.com
    outside:operations@console.company.com
    ```

11. If the nodes use NIS (*yp* is *chkconfig*'ed **on**) or the BIND domain name server (DNS), switching to local name resolution is recommended. Create or modify the file */etc/resolv.conf* so that `local` is listed first for the hostresorder keyword (the order of `nis` and `bind` is up to you):

    ```
    hostresorder local nis bind
    ```

12. If FDDI is being used, finish configuring the new FDDI station, as explained in Chapter 2 of the *FDDIXpress Administration Guide*. If you are familiar with the procedures, follow the section "Quick and Easy Configuration Instructions" in that chapter. Verify the FDDI connection as explained in Chapter 2 of the *FDDIXpress Administration Guide*.

Configuring the Serial Port

The *getty* process for the tty ports to which the reset serial cables are connected must be turned off. Perform these steps on each node:

1. Determine which port is used for the reset serial line (see the section "Serial Port Configuration" in Chapter 2).
2. Open the file */etc/inittab* for editing.
3. Find the line for the port by looking at the comments on the right for the port number from step 1.
4. Change the third field of this line to **off**, for example:

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

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

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

Configuring NFS Filesystems

Follow the procedure below to perform the NFS configuration for the filesystems to be failed over and for the status monitor *statd*. Note that no entries in */etc/exports* are required for these filesystems; IRIS FailSafe software exports NFS filesystems.

1. Create or identify the filesystems to be failed over. They must follow the guidelines in the section "Planning Filesystems" in Chapter 2.
2. On each node that exports filesystems, create the statmon directory on any one of the exported filesystems on a shared disk. The name of the directory must be *statmon*.

3. For nodes that are running IRIX 5.3, skip the remaining steps in this procedure; they apply only to nodes that are running IRIX 6.2.
4. On one node, open the file */etc/config/statd.options* for editing (it may be new).
5. Put **-h** on the first line of the file:

   ```
   -h
   ```

6. Close the file.
7. Set the owner, group, and mode of the file, if you just created it:

   ```
   # chown root.sys /etc/config/statd.options
   # chmod 644 /etc/config/statd.options
   ```

8. Repeat steps 4 through 7 on the other node.

Configuring a Netscape Server

To configure a Netscape Communications or Commerce Server, follow these steps:

1. Because Netscape server installation requires that the interface be accessible from a Netscape browser, *ifconfig* the interface to the public network up by entering this command:

   # **/usr/etc/ifconfig** *interface* **alias** *ip-address* **netmask** *netmask*

   interface is the interface to be used access the server, *ip-address* is a high availability IP address for the interface, and *netmask* is the netmask of the IP address.

2. To install the first Web server (*stocks*), enter

   ```
   # cd /usr/netscape/httpd/install
   # ./ns-setup
   ```

3. Start the Netscape browser on a workstation and open the Netscape server's configuration page (see the output of the *ns-setup* command for the URL).
4. Click the *Server Config* button and enter information according to the prompts. An example of the server information is as follows:
   - name of the server, for example *stocks.company.com*
   - IP address of the server, for example 190.0.2.3, if this is the only IP address that this server responds to (if it responds to any request, don't specify the IP address)

- accessible on port *80*, the default port for the Communications server, or port 443, the default port for the Commerce server

  Check */etc/services* to make sure the port you want is not already in use. If you choose the default port, the URL to your home page will be *http://servername.domainname*. If you choose a different port, the URL to your home page will be *http://servername.domainname:portnumber*.

  To configure multiple Web servers, use the same IP address and different port numbers or different IP addresses and different port numbers.

- installed in the directory */usr/ns-home*
- errors recorded to a file in the server root

5. Click the *Document Config* button and enter information according to the prompts. An example of the information is as follows:
   - server looks for documents in */usr/ns-home/stocks_root* (or use a symbolic link to a directory on a filesystem on a shared disk)
   - server looks for the documents *index.html* and *home.html* in directories
6. Click the *Admin Config* button and enter information according to the prompts. Exit the page.
7. Copy the contents of the server's root (*/usr/ns-home/httpd-80.190.2.3*) and document root (*/usr/ns-home/stocks_root*) to a filesystem on a shared disk.
8. Replace the original directory with symbolic links to the directories on the shared disk.

Configuring IRIS FailSafe On

To configure IRIS FailSafe so that it starts up on a node each time it is rebooted, enter this command on the node:

```
# chkconfig failsafe on
```

Note: During the testing described in Chapter 5, "Testing IRIS FailSafe Configuration," IRIS FailSafe should be *chkconfig*'d **off**.

Chapter 4

Creating the IRIS FailSafe Configuration File

The IRIS FailSafe system uses the configuration file */var/ha/ha.conf* to determine system resources, such as the node names, network interface names and addresses, and shared storage parameters. This chapter explains how to create and customize a configuration file for your IRIS FailSafe system.

The major sections of this chapter are as follows:

A Note About Configuration File Formats and Their Versions

The version number of the format of *ha.conf* is *version-major*.*version-minor*, where *version-major* is the value of the version-major configuration parameter and *version-minor* is the value of the version-minor configuration parameter (see the section "Internal Block" in this chapter). IRIS FailSafe 1.0 uses version 1.0 *ha.conf* files. IRIS FailSafe 1.1 uses version 1.0 or 1.1 *ha.conf* files.

This chapter describes 1.1 *ha.conf* files. A 1.0 *ha.conf* file developed for IRIS FailSafe Release 1.0 can be used as is with IRIS FailSafe Release 1.1. However, features new to IRIS FailSafe in the 1.1 release, such as IP aliasing, multiple IP addresses, and database failover, cannot be added to a 1.0 *ha.conf* file. To use any of the new 1.1 features you must develop a 1.1 *ha.conf* file. For information about the 1.1 features, see the section "Overview of IRIS FailSafe 1.1 for IRIS FailSafe 1.0 Users" in Chapter 1.

Creating a Configuration File

Follow these steps to create a new configuration file for an IRIS FailSafe cluster:

1. Determine your IRIS FailSafe configuration by studying the examples in Chapter 2, “Planning IRIS FailSafe Configuration,” and developing your own diagrams of your configuration and lists of parameters as suggested in that chapter.
2. Begin creating your configuration file by concatenating configuration file templates from the directory */var/ha/templates* in the order below and saving the result in a temporary file in a convenient location:
   - one copy of *ha.conf.system*
   - one copy of *ha.conf.interfaces*
   - one copy of *ha.conf.volumes* if there are shared disks in your configuration
   - one copy of *ha.conf.filesystems* if XFS filesystems are used on shared disks in your configuration
   - one copy of *ha.conf.nfs*, if you are using the optional IRIS FailSafe NFS product
   - one copy of *ha.conf.web*, if you are using the optional IRIS FailSafe Web product
3. Using the information you prepared with Chapter 2 and the information in the section “The Blocks in the Configuration File” in this chapter, edit your configuration file from step 2.
4. On one of the nodes in the IRIS FailSafe cluster, verify the format and contents of your configuration file by running the *ha_cfgverify* command:

   # **/usr/etc/ha_cfgverify** *yourfile*

   The messages output by this command are described in Appendix A, “Messages About Configuration File Errors.”
5. For warnings reported by *ha_cfgverify*, check manually that there are no errors in the configuration file.
6. Resolve errors reported by *ha_cfgverify* and re-run the command until it reports that the file has passed.

7. Copy the configuration file to any location on the other node in the cluster and repeat steps 4 through 6.
8. Copy the final configuration file from the second node back to the original file so that the copies of the configuration file on each node are identical.
9. If IRIS FailSafe is not running on the nodes in the cluster, install the configuration file by entering these commands on each node:

   ```
   # cp yourfile /var/ha/ha.conf
   # chown root.sys /var/ha/ha.conf
   # chmod 500 /var/ha/ha.conf
   # /usr/etc/ha_cfgchksum
   ```

   The checksums output by *ha_cfgchksum* on each node must be identical.
10. If IRIS FailSafe is running on the cluster, follow the instructions in the section "Upgrade Procedure C" in Chapter 7 to install the configuration file on each node.

The Blocks in the Configuration File

An IRIS FailSafe configuration file includes a series of blocks that define the configuration of an IRIS FailSafe system and the *application classes* (categories of resources and applications) that are to be failed over on the system. The blocks contain a hierarchical description of parameters and their values. These parameters provide the IRIS FailSafe software with information about the highly available services on the IRIS FailSafe cluster.

Table 4-1 lists the blocks in a configuration file and summarizes their contents.

Table 4-1 Major Blocks in *ha.conf*

Name	Description
action *service*	Describes the scripts that are to be executed for the application class *service*. An action block exists for each application class. For all application classes the action block specifies the local monitor script. For the main application class, it also specifies the *giveaway*, *giveback*, *takeback*, *takeover*, and *kill* scripts.
action-timer *service*	Describes the various timers that are used by the application monitor to decide when to execute and time out a script. The values can be adjusted based upon expected response time.
application-class	Describes one application classes that is failed over by this IRIS FailSafe cluster. These blocks identify the nodes that provide the application class service in normal state.
filesystem *label*	Each filesystem block describes a single shared filesystem. For each filesystem, it specifies the primary and backup node and mount information. There should be one filesystem block for each filesystem on a shared disk in the cluster.
interface-pair *label*	Contains IP addresses to be failed over and the primary interface and secondary interface for the IP addresses.
internal	Describes the various timeout values that are used by FailSafe daemons. The values in this block, except for long-timeout, must not be changed.
nfs *label*	nfs blocks are present if NFS is failed over in this cluster. Each nfs block describes the NFS export information associated with a single exported filesystem.
node *label*	Each node block describes the network interface configuration of a node in the cluster.
system-configuration	Describes global variables for the IRIS FailSafe cluster as a whole.
volume *label*	Each volume block describes the ownership and location of one XLV volume.
webserver	webserver blocks are present if any of the nodes in this cluster are Netscape servers. Each webserver block describes the webserver configurations of one node.

The syntax of the configuration file is defined by the following grammar:

<config> ::= EOF
| <blocks> EOF

<blocks> ::= <block>
| <blocks> <block>

<block> ::= STRING '=' VALUE
| STRING <compound>
| STRING STRING <compound>
| STRING '=' '(' <items> ')'

<items> ::= <item>
| <items> <item>

<item> ::= VALUE

<compound> ::= '{' <blocks> '}'

Every equal sign, =, must be separated from the characters before and after it by whitespace (space characters, tabs, or newline). Comments begin with a pound sign, #, and terminate at the end of the line on which they appear.

The subsections below show examples of each type of block in the configuration file and describe the parameters in each block.

System-Configuration Block

Example 4-1 shows an example system-configuration block.

Example 4-1 System-Configuration Block

```
system-configuration
{
        mail-dest-addr = root@localhost
        reset-host = irisconsole
        pwrfail = true
}
```

The parameters in this block are:

mail-dest-addr
: Mail is sent to this address when private network failure has been detected, the local node controller process appears to be hung or dead, the cluster is transitioning to degraded state, the cluster is transitioning to standby state, killing of a node fails, the *ha_killd* daemon has died, the *ha_killd* daemon could not be started, or the reset device monitor has failed. Do not set if mail is not configured on this node.

reset-host
: This parameter applies only to systems running the Oracle Parallel Server (OPS). If an IRISconsole is used to provide reset functionality, the value is the hostname of the Indy running OPS software. reset-host is ignored if reset-tty is set. Do not set reset-host if you are not running OPS in the IRIS FailSafe cluster.

pwrfail
: This parameter does not apply to CHALLENGE S nodes. When set to **true** (the default), it allows the surviving node to attempt to go to degraded state after it detects a power failure on the other node (or the private network, public network, and serial connections are broken). If pwrfail is set to **false** or is not set, the node goes to standby state after it detects a power failure on the other node.

Node Blocks

A node block describes each of the public interfaces for that node that have to be monitored. It also has information about heartbeat messages and the private network.

Example 4-2 shows an example of the node block for one node. Each configuration file must include a second, similar node block for the other node.

Example 4-2 Node Block

```
node xfs-ha1
{
        interface xfs-ha1-ec0
        {
                name = ec0
                ip-address = xfs-ha1
                netmask = 0xffffff00
                broadcast-addr = 190.0.2.255
                # MAC-address = 8:0:69:2:65:fd
        }
        heartbeat
        {
                hb-private-ipname = priv-xfs-ha1
                hb-public-ipname = xfs-ha1
                hb-probe-time = 5
                hb-timeout = 5
                hb-lost-count = 3
        }
        reset-tty = /dev/ttyd2
        controlled-failback = false
}
```

The node label (xfs-ha1 in this example) must match the return value of the *hostname* command on the node.

The sections in a node block are:

interface — An interface section describes a public interface for the node. The interface label is created from the ip-address and name parameters and must be unique in the configuration file. There is one interface section for each public interface in the node that is part of IRIS FailSafe. Not all public interfaces need to be part of IRIS FailSafe.

heartbeat — This section specifies heartbeat monitoring parameters.

The configuration parameters used in this block are:

name
: A network interface. Each node has several network interfaces. For example, a CHALLENGE S node has the network interfaces ec0, ec2, and ec3.

ip-address
: The IP address for this interface. It can be a name (*string*) or an address (*X.X.X.X*). The IP address must be configured in the file */etc/config/netif.options*. It is not failed over and is known as a *fixed* IP address.

netmask
: The netmask used to identify the subnet of this node.

broadcast-addr
: The broadcast address for the subnet.

MAC-address
: MAC addresses (*X:X:X:X:X:X*) are required only if the network interfaces have to use MAC address impersonation (see the section "Network Interfaces and IP Addresses" in Chapter 1). To get the MAC address for an interface, enter this command:

```
# /usr/etc/macconfig
Name      Physical Address    IP Address
ec0       8:0:69:8:fe:2b      190.0.2.1
ec3       8:0:69:2:65:fd      190.0.3.1
```

The Physical Address column lists the interface's MAC address (interfaces must be configured up to be displayed, for more information see the macconfig(1M) reference page).

hb-private-ipname
: This node's IP address in string form or *X.X.X.X* notation for the private network used by heartbeat and control messages.

hb-public-ipname
: This node's IP address in string form or *X.X.X.X* notation for the public network that is used for heartbeat messages if the private network fails. This IP address is a fixed IP address, which means that it is not failed over when this node fails. Fixed IP addresses are configured in the file */etc/config/netif.options*.

hb-probe-time
: Heartbeat messages begin this many seconds after the node controller tells the application monitor to start monitoring. Also, this value specifies how long to wait (in seconds) after completion of the last heartbeat message to begin the next heartbeat message. The recommended value for hb-probe-time is 5.

hb-timeout
: Specifies how long (in seconds) to wait for a heartbeat response before declaring a failure. The recommended value for hb-timeout is 5.

hb-lost-count
: Specifies how many heartbeat probe failures must occur to declare a heartbeat failure. The worst case length of time that it takes for one node to declare that the other node is down is

  (hb-probe-time + hb-timeout) * hb-lost-count

  The recommended value for hb-lost-count is 3.

reset-tty
: The device filename of the serial port used by the serial cable connected to the remote power control unit or system controller port.

controlled-failback
: Controls whether this node automatically moves to normal state after a failure. If controlled-failback is set to **true**, the node doesn't move to normal state after failure; it moves to controlled failback state. If set to **false** or not set, the node moves to normal state.

Interface-Pair Blocks

IRIS FailSafe software fails over high availability IP addresses and MAC addresses that are configured in interface-pair blocks. (Fixed IP addresses are not failed over and are specified in node blocks.)

Example 4-3 shows two interface-pair blocks. They are based on the node block for xfs-ha1 shown in Example 4-2 and a similar node block for xfs-ha2. The configuration is shown in Figure 2-4. In the example interface-pair blocks shown in Example 4-3, the interface-pair block labeled one specifies that if there is a failure on node xfs-ha1, the IP alias stocks (configured on the primary interface ec0 on xfs-ha1) moves to interface ec0 in the node xfs-ha2 (the secondary interface). After failover, both of the high availability IP addresses stocks and bonds are automatically configured on the ec0 interface of xfs-ha2 by IRIS FailSafe using IP aliasing. Similarly, the interface-pair block labeled two specifies that the IP alias bonds (configured on the primary interface ec0 on xfs-ha2) moves to interface ec0 in the node xfs-ha1 (the secondary interface) during failover.

Example 4-3 Interface-Pair Blocks

```
interface-pair one {
        primary-interface = xfs-ha1-ec0
        secondary-interface = xfs-ha2-ec0
        re-mac = false
        netmask = 0xffffff00
        broadcast-addr = 190.0.2.255
        ip-aliases = ( stocks )
}
interface-pair two {
        primary-interface = xfs-ha2-ec0
        secondary-interface = xfs-ha1-ec0
        re-mac = false
        netmask = 0xffffff00
        broadcast-addr = 190.0.2.255
        ip-aliases = ( bonds )
}
```

The interface-pair labels (one and two in this example) are arbitrary and are not used elsewhere in the configuration file.

The parameters used in interface-pair blocks are:

primary-interface
: A name for an interface on which the IP aliases are configured in normal state, typically created by combining the hostname and interface name. The value must match an interface section label in a node block.

secondary-interface
: A name for an interface on the other node which replaces the primary-interface on failover. It is typically created by concatenating the hostname and interface name. The value must match an interface section label in a node block.

re-mac
: If the IP address and the physical address of the primary interface are to be transferred to the secondary interface when a failover occurs, set re-mac to **true**. Otherwise set it to **false** or leave it undefined.

netmask
: The netmask for the IP aliases in *X.X.X.X* notation.

broadcast-addr
: The subnet broadcast IP address for the IP aliases in *X.X.X.X* notation.

ip-aliases
: The list of IP addresses to be failed over using IP aliasing. The format of the value is a left parenthesis, IP addresses separated by blanks, and a right parenthesis.

Interface-Agent Block

The interface-agent block describes the monitors and timeouts associated with the interface agent.

Example 4-4 shows an example interface-agent block.

Example 4-4 Interface-Agent Block

```
interface-agent {
        start-monitor-time = 60
        interface-probe-interval = 30
        interface-probe-timeout = 20
        remote-send-probe-interval = 25
        remote-send-timeout = 10
}
```

The configuration parameters used in the interface-agent block are:

start-monitor-time
: The length of time (in seconds) that the interface agent waits after it is told by the application monitor to start monitoring the local interfaces before beginning to monitor. This value must be greater than or equal to the value of long-timeout.

interface-probe-interval
: The length of time (in seconds) between the completion of one probe of the local interfaces and the beginning of the next probe. The value is rounded to the nearest five second increment.

interface-probe-timeout
: The length of time (in seconds) that the interface agent waits after probing the local interfaces without a response before declaring a failure.

remote-send-probe-interval
: The length of time (in seconds) between the completion of one probe of the other node's highly available interfaces and the beginning of the next probe. This parameter must be less than the value of interface-probe-interval, but should be only slightly less. The value is rounded to the nearest five seconds.

remote-send-timeout
: The length of time (in seconds) that the interface agent waits before declaring that a one probe of the other node's highly available interfaces has failed. The minimum value is 10.

Standard Application-Class Blocks

The application classes are IRIS FailSafe highly available services. An application class is a category of service; there may be many instances of the service. For example there may be many NFS filesystems; their application class is nfs. The main, interfaces, volumes, and filesystems application classes are provided as part of the IRIS FailSafe product. NFS, Netscape, INFORMIX, Oracle, and Sybase application classes are supported by optional IRIS FailSafe products. Each application class is provided by at least one node, which is called the primary node and is designated as server-node in the configuration file.

Example 4-5 shows the application-class blocks for main, interfaces, volumes, and filesystems (without comments).

Example 4-5 Standard Application-Class Blocks

```
application-class main
{
        server-node = xfs-ha1
        server-node = xfs-ha2
}

application-class interfaces
{
        server-node = xfs-ha1
        server-node = xfs-ha2
        agent = /usr/etc/ha_ifa
}

application-class volumes
{
        server-node = xfs-ha1
        server-node = xfs-ha2
}

application-class filesystems
{
        server-node = xfs-ha1
        server-node = xfs-ha2
}
```

The parameters used in these application-class blocks are:

server-node — The primary node for instances of the application class. Server-node is listed twice if each of the nodes in a cluster serves as the primary node for some instances. The values assigned to server-node must match the labels for node blocks.

agent — The pathname of the agent for the application class.

Volume Blocks

The IRIS FailSafe software supports failover for XLV logical volumes, allowing the backup node to take over volumes if the primary node fails. These XLV logical volumes must be created on either plexed (mirrored) disks or on CHALLENGE RAID disks. The volume block describes the primary and backup node for one volume and its device name.

Example 4-6 shows an example of a volume block.

Example 4-6 Volume Block

```
volume shared1_vol
{
        server-node = xfs-ha1
        backup-node = xfs-ha2
        devname = /dev/dsk/xlv/shared1_vol
        devname-owner = root
        devname-group = sys
        devname-mode = 0600
}
```

The parameters used in volume blocks are:

server-node
: The primary node for this XLV logical volume. The value assigned to server-node must match the label for a node block.

backup-node
: The secondary node for this XLV logical volume. The value assigned to backup-node must match the label for a node block.

devname
: The device name for the XLV logical volume.

devname-owner
: The user ID of the device name for the XLV logical volume (reported by *ls -l*). The default value is **root**.

devname-group
: The group ID of the device name for the XLV logical volume (reported by *ls -l*). The default value is **sys**.

devname-mode
: The access mode of the device name for the XLV logical volume (reported by *ls -l*). The default value is **0600**.

Filesystem Blocks

The IRIS FailSafe software supports failover for filesystems on shared XLV logical volumes, allowing the backup node to take over filesystems if the primary node fails. The filesystem block describes the mount options and arguments passed to the *mount* command.

Example 4-7 shows a filesystem block.

Example 4-7 Filesystem Block

```
filesystem shared1
{
        mount-point = /shared1
        mount-info {
                fs-type = xfs
                volume-name = shared1_vol
                mode = rw,noauto,wsync
        }
}
```

The parameters used in filesystem blocks are:

mount-point	The pathname of a filesystem mount point. Both nodes use the same mount point.
fs-type	The filesystem type. Only xfs filesystems are supported.
volume-name	The value assigned to volume-name must match a volume block label.
mode	The filesystem mount options (see the fstab(4) reference page). See the section “Wsync Filesystem Options” in this chapter for information about the wsync option.

Action Blocks

Action blocks are defined for each application class. The action block specifies the pathnames of the monitoring scripts.

Example 4-8 shows the action blocks for the main, interfaces, volumes, and filesystems application classes.

Example 4-8 Standard Action Blocks

```
action main
{
        giveaway = /var/ha/actions/giveaway
        giveback = /var/ha/actions/giveback
        takeback = /var/ha/actions/takeback
        takeover = /var/ha/actions/takeover
        kill = /usr/etc/ha_kill
}

action interfaces
{
        local-monitor = /var/ha/actions/ha_ifa_lmon
}

action volumes
{
        local-monitor = /var/ha/actions/ha_vol_lmon
}

action filesystems
{
        local-monitor = /var/ha/actions/ha_filesys_lmon
}
```

The parameters used in the action block for the main application class are:

giveaway, giveback, takeback, takeover
: The pathnames of the *giveaway*, *giveback*, *takeback*, and *takeover* scripts in the */var/ha/actions* directory.

kill
: The pathname of the *ha_kill* command.

The parameter used in action blocks for other application classes is:

local-monitor
: The pathname of the local monitoring script for this application class.

Action-Timer Blocks

Action-timer blocks are defined for each application class that is monitored. They specify various monitoring frequency parameters. Information about determining values for monitoring frequency parameters is provided in the section “Choosing Monitoring Frequency Values” later in this chapter. An action-timer block is required for each application class except the main application class.

Example 4-9 shows the action-timer blocks for the interfaces, volumes, and filesystems application classes. There is no action-timer block for the main application class.

Example 4-9 Standard Action-Timer Blocks

```
action-timer interfaces
{
        start-monitor-time = 60
        lmon-probe-time = 60
        lmon-timeout = 60
        retry-count = 5
}

action-timer volumes
{
        start-monitor-time = 300
        lmon-probe-time = 300
        lmon-timeout = 60
        retry-count = 1
}

action-timer filesystems
{
        start-monitor-time = 60
        lmon-probe-time = 120
        lmon-timeout = 60
        retry-count = 1
}
```

The parameters used in action-timer blocks vary with the application class. The parameters used for application classes are:

start-monitor-time
: The length of time (in seconds) that the monitoring script waits after it is told by the application monitor to start monitoring before it begins monitoring. The value must be greater than or equal to the value of long-timeout and is required.

lmon-probe-time
: The probe interval (in seconds) for local monitoring of this application class.

lmon-timeout
: Local monitoring of this application class times out in this many seconds if no response is received.

retry-count
: The local monitoring script does retries of certain commands. The value of retry-count determines the number of retries. The application monitor declares a local monitor failure after lmon-timeout seconds independent of the retry-count value.

Internal Block

The internal block contains timeout and version parameters. With the exception of long-timeout, these parameter values should not be changed. Example 4-10 shows an example of an internal block.

Example 4-10 Internal Block

```
internal
{
        short-timeout = 5
        long-timeout = 60
        version-major = 1
        version-minor = 1
}
```

The parameters in the internal block are:

short-timeout — The maximum length of time (in seconds) for certain communications tasks to complete.

long-timeout — This value is the maximum time taken by the *takeover*, *takeback*, *giveaway*, and *giveback* scripts. If these scripts cannot be executed in this length of time, the value should be increased. This value is also used as the maximum time taken by several types of internal IRIS FailSafe communications.

version-major — A version number that, along with version-minor, specifies the configuration file format used in this file.

version-minor — A version number that, along with version-major, specifies the configuration file format used in this file.

NFS Blocks

Example 4-11 shows the application-class block in a dual-active NFS configuration. In this configuration, each node (xfs-ha1 and xfs-ha2) is the primary node for at least one filesystem. For an active / backup configuration, only one server-node would be included in the application-class block for nfs.

Example 4-11 NFS Application-Class Block

```
application-class nfs
{
        server-node xfs-ha1 {
                statmon-dir = /shared1/statmon
                ip-aliases = ( stocks )
        }
        server-node xfs-ha2 {
                statmon-dir = /shared2/statmon
                ip-aliases = ( bonds )
        }
}
```

There is a server-node section for each node that is a primary node for NFS services. Its label matches the label of one of the node blocks. The parameters in the server-node sections are:

statmon-dir — Specifies the pathname of a directory on a shared filesystem belonging to its server-node where NFS locking information is stored. This directory should be used exclusively for NFS locking information. The basename of this directory must be *statmon*.

ip-aliases — Specifies a list of IP aliases. This parameter, with at least one IP alias, is required if NFS file locking is used (listing at least one IP alias is recommended). Each IP alias that is used by NFS clients to mount NFS filesystems should be listed.

The nfs blocks specify exported NFS filesystems including the options and arguments used by the *exportfs* command. Example 4-12 shows an example of an nfs block. You must create an nfs block for every filesystem to be exported and failed over.

Example 4-12 NFS Block

```
nfs shared1
{
        filesystem = shared1
        export-point = /shared1
        export-info = rw,wsync
        ip-address = 190.0.2.3
}
```

The label for the nfs block must match the label of a filesystem block. The parameters used in nfs blocks are:

filesystem — A filesystem to be exported. The value of this parameter must match the label of a filesystem block and the label of the block.

export-point — The pathname of an exported filesystem.

export-info — Filesystem export options (see the exports(4) reference page). See the section “Wsync Filesystem Options” in this chapter for information about the wsync option.

ip-address — One (any one) of the IP addresses or IP aliases on the primary node for this filesystem, preferably in the form *X.X.X.X*. A good choice is the IP address or IP alias used by clients to mount the filesystem.

Example 4-13 shows action and action-timer blocks (without comments) for NFS.

Example 4-13 NFS Action and Action-Timer Blocks

```
action nfs
{
        local-monitor = /var/ha/actions/ha_nfs_lmon
}

action-timer nfs
{
        start-monitor-time = 60
        lmon-probe-time = 20
        lmon-timeout = 30
        retry-count = 1
}
```

The parameters in these blocks are explained in the sections “Action Blocks” and “Action-Timer Blocks” in this chapter.

Webserver Blocks

Example 4-14 shows the webserver application-class block for an active/backup Netscape server configuration.

Example 4-14 Webserver Application-Class Block (Active/Backup Configuration)

```
application-class webserver
{
        server-node = xfs-ha1
}
```

Example 4-15 shows the webserver application-class block for a dual-active Netscape server configuration.

Example 4-15 Webserver Application-Class Block (Dual-Active Configuration)

```
application-class webserver
{
        server-node = xfs-ha1
        server-node = xfs-ha2
}
```

The server-node parameters are the nodes that serve as primary nodes for Netscape servers. Their values must match a node label used in a node block.

The webserver block in a Netscape server configuration file specifies the location and ports for the Netscape servers. Example 4-16 shows this block for an active/backup configuration. In the example in Example 4-16, the primary node is configured with two Netscape Commerce servers: one listens to the default port, 443, and the other one listens to port 453.

Example 4-16 Webserver Block (Active/Backup Configuration)

```
webserver webxfs-ha1 {
        server-node = xfs-ha1
        backup-node = xfs-ha2
        httpd-script = /etc/init.d/ns_commerce
        httpd-options-file = ns_commerce.options
        httpd-restart = false
        webserver-num = 2

        web-config1 {
                ip-address = 190.0.2.3
                port-num = 443
                httpd-dir = /shared/httpd-443
        }
        web-config2 {
                ip-address = 190.0.2.5
                port-num = 453
                httpd-dir = /shared/httpd-453.190.0.2.5
        }
}
```

Example 4-17 shows the webserver blocks for a dual-active configuration. In this example, each node is the primary node for one Netscape Communications server.

Example 4-17 Webserver Blocks (Dual-Active Configuration)

```
webserver webxfs-ha1 {
        server-node = xfs-ha1
        backup-node = xfs-ha2
        httpd-restart = false
        webserver-num = 1
        web-config1 {
                ip-address = 190.0.2.3
                port-num = 80
                httpd-dir = /shared.stocks/httpd-80
        }
}

webserver webxfs-ha2 {
        server-node = xfs-ha2
        backup-node = xfs-ha1
        httpd-restart = false
        webserver-num = 1
        web-config1 {
                ip-address = 190.0.2.4
                port-num = 80
                httpd-dir = /shared.bonds/httpd-80
        }
}
```

The webserver labels (webxfs-ha1 and webxfs-ha2) are arbitrary. The parameters used in webserver blocks are:

server-node — The primary node for this Netscape server. The value must match the label for a node block.

backup-node — The secondary node for the Netscape server. The value must match the label for a node block.

httpd-options-file — The Netscape configuration file that starts multiple Netscape servers. The value is not a full pathname; it is a file in the directory */etc/config*. The default value is ns_httpd.options, which is the configuration file for the Netscape Communications server.

httpd-script	The full pathname of the script used to start and stop the Netscape server. The default value is /etc/init.d/ns_httpd, which is the configuration file for the Netscape Communications server.
httpd-restart	If the two nodes have identical Netscape server configurations (same configuration information, log locations, and document root) then Netscape server doesn't need to be restarted after failover (because an identical server is already running) and httpd-restart should be set to **true**. Otherwise, the Netscape server needs to be started on the backup node after failover and httpd-restart should be set to **false**.
webserver-num	The number of Netscape servers configured on this node (server-node). This is the number of web-config sections for this Netscape server.
ip-address	The high availability IP address used by clients to access the Netscape server. An IP address of the form *X.X.X.X* is recommended.
port-num	The Netscape server port number.
httpd-dir	The Netscape server root location.

There is one webconfig*n* section (webconfig1, webconfig2, and so on) for each Netscape server on a node.

Example 4-18 shows webserver action and action-timer blocks (without comments).

Example 4-18 Webserver Action and Action-Timer Blocks

```
action webserver
{
        local-monitor = /var/ha/actions/ha_web_lmon
}

action-timer webserver
{
        start-monitor-time = 60
        lmon-probe-time = 20
        lmon-timeout = 30
        retry-count = 1
}
```

The parameters in these blocks are explained in the sections "Action Blocks" and "Action-Timer Blocks" in this chapter.

Wsync Filesystem Options

NFS clients writing to mounted filesystems typically receive confirmation from the node that a write has completed once data has been successfully received at the node, but not necessarily written to disk. This delayed-write (also known as asynchronous write) feature can greatly improve performance because it allows the node's disk buffer cache to be utilized. In a FailSafe configuration, however, the potential for data corruption due to this feature of NFS is greatly increased because of the automatic failover mechanism (power to the failed node is cut).

You can reconfigure NFS exports and XFS local mounts on the node to perform writes synchronously to disk (when received) by adding the parameter *wsync* to the mode parameter in all filesystem blocks and to the export-info parameter in all nfs blocks. This means that a reply to an NFS write is not returned until the NFS write data gets written to the server's disk. However, performance is greatly affected by adding the *wsync* option in the filesystem blocks and the *wsync* option in nfs blocks. You must balance the risk of NFS data corruption during a node failure against the performance gain from using asynchronous NFS writes. Some applications perform their own error checking of data or perform writes in such a way that data corruption does not occur.

For more information on the performance impact when *wsync* is used, see the *ONC3/NFS™ Administration Guide* (007-0850-xxx).

Monitoring Frequency Parameters

The subsections below discuss the monitoring frequency parameters for the application classes and provide information to help you choose values for these parameters that are appropriate for your IRIS FailSafe cluster.

Monitoring Frequency Parameters and the Communication Path They Control

Figure 4-1 shows the communication paths between the IRIS daemons and scripts. Each of the communications paths that has monitoring frequency parameters associated with it is labeled with those parameters. You can use this diagram to help you understand which communication path and monitoring activity each of the parameters controls.

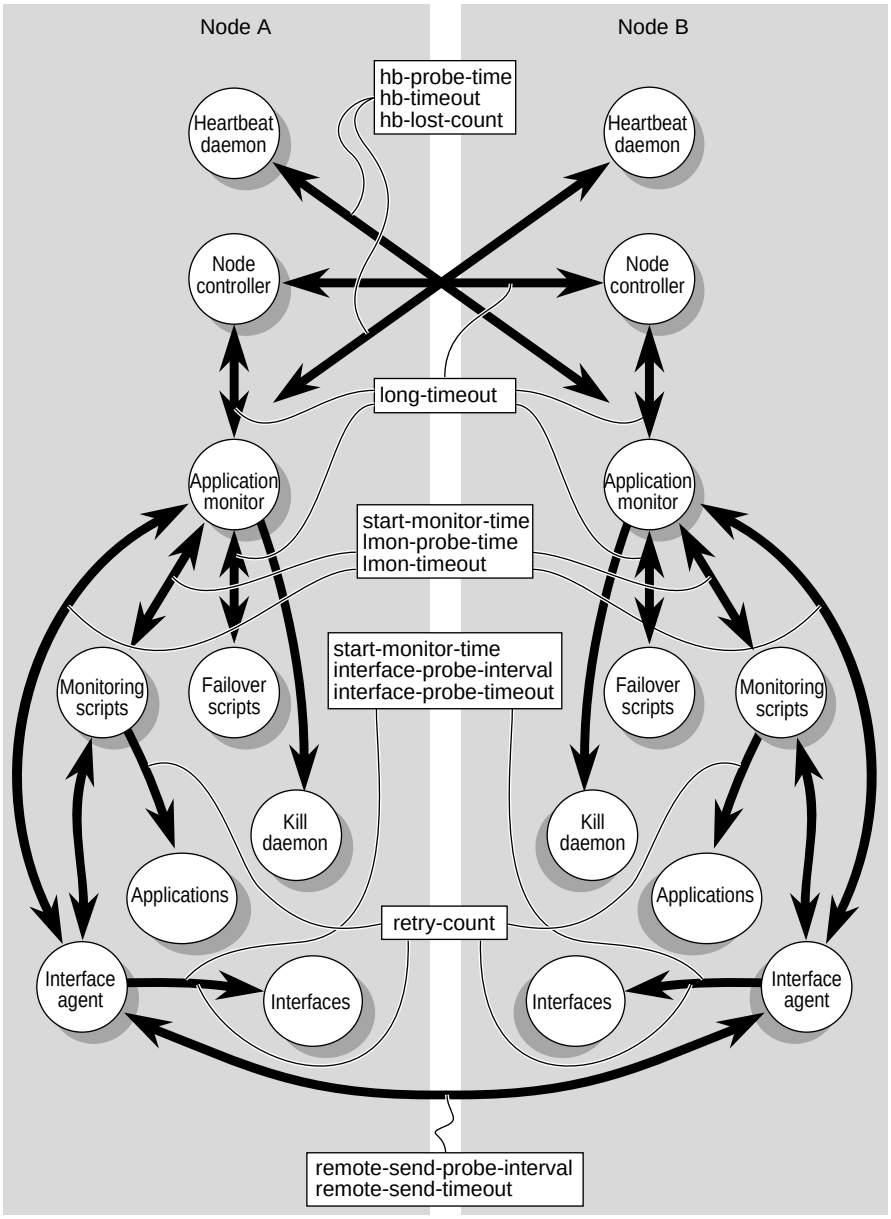


Figure 4-1 Monitoring Frequency Parameters

What Each Monitoring Frequency Parameter Means

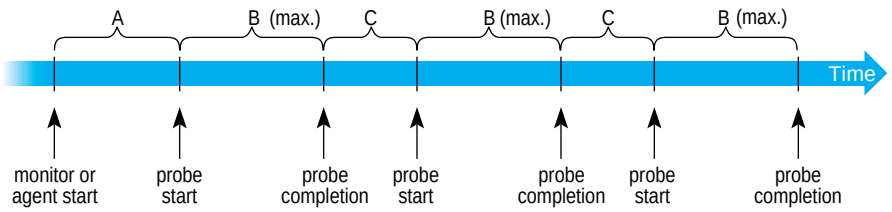
The start-monitor-time parameter is the length of time in seconds that the application monitor or agent waits after the application instances have been started up by IRIS FailSafe before performing its first probe. This time is provided to allow the resource or application to start up.

The hb-timeout, lmon-timeout, interface-probe-timeout, and remote-send-timeout parameters are the maximum lengths of time that the application monitor or interface agent waits for a response to a probe. If a response is received, the probe is completed and has been successful. If a response isn't received within the hb-timeout, lmon-timeout, interface-probe-timeout, or remote-send-timeout length of time, the probe is also completed, but it has failed.

The hb-probe-time, lmon-probe-time, interface-probe-interval, and remote-send-probe-interval parameters are the length of time in seconds that the application monitor or interface agent waits from the completion (successful completion or timeout) of the previous probe before performing the next probe.

Figure 4-2 has a time line that shows the relationship between the parameters described above. It illustrates that:

- The minimum period between probes is hb-probe-time or lmon-probe-time or interface-probe-interval (depending upon the process being monitored).
- The maximum period between probes is a probe-time or probe-interval parameter plus a timeout parameter.



A = hb-probe-time or start-monitor-time
B = hb-timeout or lmon-timeout or interface-probe-timeout
C = hb-probe-time or lmon-probe-time or interface-probe-time

Figure 4-2 Monitoring Timing

For applications monitored by scripts (NFS filesystems and Netscape servers), retry-count is the number of times that some of the commands internal to the scripts are retried. The retry-count value does not affect the application failure detection time. The worst case application failure detection time is:

xxx-probe-time + *xxx*-timeout

For applications monitored by agents (the interface agent or a database agent), retry-count is the number of times that the probes are retried before declaring an application failure. The worst case application failure detection time is:

(*xxx*-probe-time + *xxx*-timeout) * retry-count

The lost-count parameter is used for heartbeat messages between the nodes in the cluster. The worst case node failure detection time using heartbeats is:

(hb-probe-time + hb-timeout) * hb-lost-count

Choosing Monitoring Frequency Values

In choosing time and retry count values for IRIS FailSafe monitoring, keep these important general considerations in mind:

- Timeout values must be long enough to avoid treating slow responses as failures.
- The failure detection time affects the time that highly available services may be unavailable.
- IRIS FailSafe monitoring can affect the performance of a node, so it is important to consider both the cost of each type of monitoring and the overall cost of all monitoring.
- All the monitoring frequencies and timeouts specified in the IRIS FailSafe configuration file are rounded up to the nearest multiple of the smaller of the values of the short-timeout and hb-probe-time parameters. Small changes to the parameter values may have no effect.
- The suggested values for each of the monitoring frequency, timeout, and retry count parameters that are given in the configuration file template files and in the examples in Chapter 4 are good starting points for selecting values. However, each cluster is different, so some tuning for your cluster will be required.

- The start-monitor-time value for all the applications should be greater than the long-timeout value.
- The long-timeout value is the maximum time that is needed for executing the failover scripts (*giveaway, takeback, takeover,* and *giveback*) for all of the applications and resources.

Preventing False Failovers

In choosing monitoring frequency values, you have two somewhat conflicting objectives:

- Choose monitoring frequency values sufficiently short that failures are detected quickly.
- Choose monitoring frequency values sufficiently long and forgiving that IRIS FailSafe doesn't mistakenly conclude that a failure has occurred and perform an unnecessary failover.

Follow these guidelines to avoid false failovers and tune the configuration file to prevent future false failovers:

- Choose higher retry-count values so that some of the commands internal to the probes are retried.
- Consider the impact of the time it takes IRIS FailSafe software to resolve IP addresses given as names in the IRIS FailSafe configuration file into IP addresses in internet notation (*X.X.X.X*). The file */etc/resolv.conf* specifies how names are resolved (see step 11 in section "Configuring Interfaces" in Chapter 3). If false failovers are occurring because the name server that resolves names fails to respond within timeout values, you may need to switch the name resolution method or increase the appropriate timeout values.
- When choosing monitoring frequencies, keep in mind the load on the public networks. The public network load is high when there is lots of activity in the network. If the timeout values of the FailSafe monitoring scripts and interface agent are short, there can be false failovers due to heavy network loads.

Chapter 5

Testing IRIS FailSafe Configuration

This chapter explains how to test the IRIS FailSafe system configuration. The tests in each section in this chapter, except the last section, are performed when IRIS FailSafe software is not running. The last section describes how to test the running IRIS FailSafe software.

The sections in this chapter are as follows:

Testing the Serial Connections

To test the serial connections between the IRIS FailSafe nodes, follow these steps:

1. If a remote power control unit is used, confirm that it is powered on by checking that the display light on the front of the box is lit green. (The section "Replacing Batteries in the Remote Power Control Unit" in Chapter 8 explains how to change the batteries.)
2. Enter this command on one node:

   # **`/usr/etc/ha_spng -i 10 -f`** *reset-tty*

   reset-tty is the value of the reset-tty parameter in the configuration file */var/ha/ha.conf*.

3. Check the return value of the command by entering the first command if you are using *csh* and the second command if you are using *sh*:

   ```
   # echo $status
   ```

   ```
   # echo $?
   ```

   If the return value is 0, the connection is good.

4. If the return value is 1**,** verify the cable connections of the serial cable from each node's serial port to the remote power control unit or the other node's system controller port.

5. Repeat steps 2 through 4 on the second node.

Testing the Private Network

To test the private (heartbeat) network, follow these steps:

1. Enter this command on one node:

   ```
   # /usr/etc/ping -r -c 3 priv-xfs-ha1
   PING priv-xfs-ha1.eng.sgi.com (190.0.3.1): 56 data bytes
   64 bytes from 190.0.3.1: icmp_seq=0 ttl=254 time=3 ms
   64 bytes from 190.0.3.1: icmp_seq=1 ttl=254 time=2 ms
   64 bytes from 190.0.3.1: icmp_seq=2 ttl=254 time=2 ms
   ```

   priv-xfs-ha1 is the private IP address of the other node. Typical *ping* output, such as that shown, should appear.

2. If the *ping* command fails, verify that the private network interface has been configured up using the *ifconfig* command, for example:

   ```
   # /usr/etc/ifconfig ec3
   ec3: flags=c63<UP,BROADCAST,NOTRAILERS,RUNNING,FILTMULTI,MULTICAST>
           inet 190.0.3.1 netmask 0xffffff00 broadcast 190.0.3.255
   ```

   The UP in the first line of output indicates that the interface is configured up.

3. If the *ping* command fails and the private network interface has been configured up, verify tha the private network cables are connected properly.

4. Repeat steps 1 through 3 on the other node.

Testing the Public Network Interfaces

The procedure below describes how to test the public interfaces on each node. It uses this interface as an example:

```
node xfs-ha1
        interface xfs-ha1-ec0
        {
                name = ec0
                ip-address = xfs-ha1
                netmask = 0xffffff00
                broadcast-addr = 190.0.2.255
        }
        ...
}
node xfs-ha2
...
interface-pair one {
        primary-interface = xfs-ha1-ec0
        secondary-interface = xfs-ha2-ec0
        re-mac = false
        netmask = 0xffffff00
        broadcast-addr = 190.0.2.255
        ip-aliases ( stocks )
}
```

Follow these steps:

1. To test the public network interfaces on the first node (xfs-ha1), enter the following command from a client:

   ```
   # /usr/etc/ping -c 3 xfs-ha1
   PING xfs-ha1.engr.sgi.com (190.0.2.1): 56 data bytes
   64 bytes from 190.0.2.1: icmp_seq=0 ttl=254 time=3 ms
   64 bytes from 190.0.2.1: icmp_seq=1 ttl=254 time=2 ms
   64 bytes from 190.0.2.1: icmp_seq=2 ttl=254 time=2 ms
   ```

   xfs-ha1 is an IP address for an interface on the node xfs-ha1.

2. Repeat step 1 for the remaining public network interfaces on xfs-ha1.
3. Repeat step 1 for all public interfaces of the other node in the cluster.

Testing Volumes

Follow the procedure below to verify that XLV logical volumes have been configured properly. It uses this portion of a configuration file as an example:

```
volume sharedsybase_vol
{
        server-node = xfs-ha1
        backup-node = xfs-ha2
        devname = /dev/dsk/xlv/shared_sybase
        devname-owner = sybase
        devname-group = sybase
        devname-mode = 0664
}
```

1. On a node that is a primary node for volumes (xfs-ha1 in this example), enter this command to stop the RAID agent if the cluster uses a CHALLENGE RAID storage system:

   ```
   # /etc/init.d/raid5 stop
   ```

2. On the same node, enter the following commands to assemble the XLV logical volume sharedsybase_vol:

   ```
   # xlv_mgr -c "change nodename xfs-ha1 shared_sybase"
   set node name xfs-ha1 for object shared_sybase done

   # xlv_assemble -l -s shared_sybase
   VOL shared_sybase       flags=0x1, [complete]         (node=xfs-ha1)
   DATA    flags=0x0()     open_flag=0x0() device=(192, 4)
   PLEX 0  flags=0x0
   VE 0    [active]
           start=0, end=3583999, (cat)grp_size=1
           /dev/dsk/dks5d1s0 (3584000 blks)
   ```

3. Repeat step 2 for each of the other volumes with the same primary node.

4. If you stopped the RAID agent in step 1, restart the RAID agent by entering this command:

   ```
   # /etc/init.d/raid5 start
   ```

5. On the same node, list all of the XLV logical volumes on the node:

   ```
   # ls -l /dev/dsk/xlv
   total 0
   brw-rw-r--    1 sybase   sybase   192,  4 May 22 11:18 shared_sybase
   ...
   ```

   You should see all volumes that have this node listed as their server-node in the configuration file.

6. Enter this command to read ten blocks from one of the XLV logical volumes (for example */dev/dsk/xlv/shared_sybase*) and discard them:

   ```
   # dd if=/dev/dsk/xlv/shared_sybase of=/dev/null count=10
   10+0 records in
   10+0 records out
   ```

   The output should match the output shown.

7. Repeat step 6 for every volume in the configuration file for which this node is the primary node.

8. If the other node serves as the primary node for any XLV logical volumes, repeat steps 1 through 7.

Testing Filesystems

The procedure below tests filesystems configured for IRIS FailSafe by executing the *mount* commands that the IRIS FailSafe software would execute. These filesystem and volume sections of a configuration file are used as an example:

```
filesystem shared1
{
           mount-point = /shared1
           mount-info
           {
                   fs-type = xfs
                   volume-name = shared1_vol
                   mode = rw,noauto
           }
}
```

```
volume shared1_vol
{
        server-node = xfs-ha1
        backup-node = xfs-ha2
        devname = /dev/dsk/xlv/shared1_vol
        devname-owner = root
        devname-group = sys
        devname-mode = 0600
}
```

For each filesystem listed in the configuration file, follow this procedure:

1. Identify the primary node for the filesystem by looking up the primary node (the server-node) of the XLV logical volume used by this filesystem. In the example above, volume-name is shared1_vol; look for the volume block with the label shared1_vol. Its server-node (primary node) is xfs-ha1.
2. On the primary node, check to see if the XLV logical volume device name exists:

   ```
   # ls /dev/dsk/xlv/shared1_vol
   ```

3. If the device name doesn't exist, enter the following commands to assemble the XLV logical volume shared1_vol:

   ```
   # xlv_mgr -c "change nodename xfs-ha1 shared1_vol"
   set node name xfs-ha1 for object shared1_vol done

   # xlv_assemble -l -s shared1_vol
   VOL shared1_vol         flags=0x1, [complete]         (node=xfs-ha1)
   DATA    flags=0x0()     open_flag=0x0() device=(192, 4)
   PLEX 0  flags=0x0
   VE 0    [active]
           start=0, end=3583999, (cat)grp_size=1
           /dev/dsk/dks5d1s0 (3584000 blks)
   ```

4. On the primary node, mount the filesystem using a *mount* command that mimics the *mount* command given by IRIS FailSafe:

   ```
   # mount -txfs -o rw,noauto /dev/dsk/xlv/shared1_vol /shared1
   ```

   The mount should be successful.

5. Unmount the filesystem:

   ```
   # umount /shared1
   ```

6. On the secondary node, check to see if the XLV logical volume device name exists:

   # **ls /dev/dsk/xlv/shared1_vol**

7. If the device name doesn't exist, enter the following commands on the secondary node to assemble the XLV logical volume shared1_vol:

```
# xlv_mgr -c "change nodename xfs-ha2 shared1_vol"
set node name xfs-ha2 for object shared1_vol done

# xlv_assemble -l -s shared1_vol
VOL shared1_vol         flags=0x1, [complete]         (node=xfs-ha2)
DATA    flags=0x0()     open_flag=0x0() device=(192, 4)
PLEX 0  flags=0x0
VE 0    [active]
        start=0, end=3583999, (cat)grp_size=1
        /dev/dsk/dks5d1s0 (3584000 blks)
```

8. Mount the filesystem on the secondary node by entering the command from step 4 on the secondary node:

   # **mount -txfs -o rw,noauto /dev/dsk/xlv/shared1_vol /shared1**

9. Unmount the filesystem:

   # **umount /shared1**

Testing NFS Configuration

The procedure below tests NFS configuration by exporting filesystems manually and determining if a client can access them.

It uses this NFS entry in *ha.conf* as an example:

```
nfs shared1
{
        filesystem = shared1
        export-point = /shared1
        export-info = rw
        ip-address = 190.0.2.3
}
```

For each NFS block in *ha.conf*, follow these steps:

1. Mount */shared1* on either node as described in the section "Testing Filesystems."

2. Make sure the IP address is configured by entering this command on the node where */shared1* was mounted:

```
# /usr/etc/ping -c 3 190.0.2.3
PING 190.0.2.3 (190.0.2.3): 56 data bytes
64 bytes from 190.0.2.3: icmp_seq=0 ttl=254 time=3 ms
64 bytes from 190.0.2.3: icmp_seq=1 ttl=254 time=2 ms
64 bytes from 190.0.2.3: icmp_seq=2 ttl=254 time=2 ms
```

3. From the node on which it is mounted, export the filesystem:

```
# exportfs -i -o rw /shared1
```

4. Make sure the filesystem was exported:

```
# exportfs
/shared1 -rw
```

5. Verify that you can mount the exported filesystem on a client by entering these commands from a client:

```
# mkdir /tempmount
# mount 190.0.2.3:/shared1 /tempmount
```

6. On the client, unmount the filesystem and remove the temporary directory:

```
# umount /shared1
# rmdir /tempmount
```

7. From the node on which the filesystem is mounted, unexport it and unmount it in preparation for running this test from the other node:

```
# exportfs -u /shared1
# umount /shared1
```

8. Repeat steps 1 through 7 on the other node. Make sure you do not mount the filesystem simultaneously from both nodes.

Testing Netscape Server Configuration

To test whether the Netscape servers are correctly configured, follow these steps:

1. Start the Netscape Communications and Commerce servers:

```
# /etc/init.d/ns_httpd start
# /etc/init.d/ns_commerce start
```

2. Run a Web browser, such as Netscape, on a client and try to access some Web pages served by the server.

3. Stop the Netscape Communications and Commerce servers:

```
# /etc/init.d/ns_httpd stop
# /etc/init.d/ns_commerce stop
```

Testing System Behavior with IRIS FailSafe Running

Testing system behavior with IRIS FailSafe running is broken into four phases in the following subsections. The phases are: preparing for testing, checking normal operation, checking failover, and cleaning up after testing.

Preparing for Testing

1. Edit the file */etc/init.d/failsafe* on each node and change the value of MIN_UPTIME from the default (300 seconds) to 0. This enables you to allow multiple failovers without the FailSafe software disabling itself due to frequent failovers.
2. Bring up IRIS FailSafe software by entering these two commands on each node:

```
# chkconfig failsafe on
# /etc/init.d/failsafe start
```

Checking Normal Operation

Follow this procedure to verify that the IRIS FailSafe cluster is operating normally:

1. Verify that the nodes are in normal state by entering this command on each node:

```
# /usr/etc/ha_admin -i
ha_admin: Node controller state normal
```

   If either node has not reached normal state, wait a few minutes and try the command again. If normal state isn't reached, check the */var/adm/SYSLOG* file on both nodes for errors. See Appendix B, "System Troubleshooting," for troubleshooting information.
2. Verify that NFS filesystems exported by the cluster by mounting them from a client.
3. Verify that Netscape servers on the cluster are working by running a browser on a client and viewing Web pages served by the Netscape servers.
4. Check any other highly available applications running on the cluster.

Checking Failover

After you have confirmed that the cluster operates correctly when both nodes are active, confirm that the cluster functions correctly in the face of failures by performing the tests below. Each test is an independent test and should be performed on an IRIS FailSafe cluster that is operating normally.

1. Power off one node in the cluster. The other node in the cluster should detect the failure and take over the services. If you have an active/backup configuration, power off the active node.
2. Disconnect the private network. If you have enabled heartbeat messages to be sent over the public network, the cluster should continue to function as before. Otherwise, one node takes over the services of the other node. The node whose services got taken over gets rebooted.

   If heartbeat messages are sent over the public network (hb-public-ipname is set to a fixed IP address), enter this command after reconnecting the private network to switch heartbeat messages back to the private network:

   ```
   # /usr/etc/ha_admin -x
   ```
3. Disconnect the public network from one of the active nodes in the cluster. The other node should take over the services.
4. Forcibly unmount a filesystem on an active node to make it unavailable. The other node should take over the services. The failed node enters standby state.
5. Kill the application daemons (for example, *ns_httpd*) on a node to make the service unavailable. The other node should take over the service.
6. Disconnect the serial line to the system controller port (if you are using CHALLENGE XL/L/DM) or the remote power control unit (if you are using CHALLENGE S). If you have configured the IRIS FailSafe software to send mail, it notifies the administrator of the failure and otherwise continues to function.

   When the cluster is in this state, neither node can take over if another failure occurs. After you have reconnected the serial line, you can resume monitoring of the serial line by executing */usr/etc/ha_admin -m start <node_name>*.

Cleaning Up After Testing

Follow this procedure to return the nodes to normal state after testing:

1. Edit the file */etc/init.d/failsafe* on each node and return the value of MIN_UPTIME to its initial suggested value, 300.
2. Restart IRIS FailSafe on both nodes by entering these commands on each node:

```
# /etc/init.d/failsafe stop
# /etc/init.d/failsafe start
```

Chapter 6

Administering IRIS FailSafe

This chapter describes administrative commands and procedures for IRIS FailSafe. The major sections in this chapter are as follows:

Starting IRIS FailSafe

Follow this procedure to start the IRIS FailSafe system:

1. On one node, enter this command to *chkconfig* IRIS FailSafe on, in case it is not already on:

   ```
   # chkconfig failsafe on
   ```

2. Reboot the node or enter this command:

   # **/etc/init.d/failsafe start**

3. Repeat steps 1 and 2 for the second node.

Shutting Down IRIS FailSafe

To stop IRIS FailSafe software on one node or both nodes in the cluster, follow the steps:

1. On one node, shut down IRIS FailSafe by entering this command:

   # **/etc/init.d/failsafe stop**

   This command causes all highly available services to fail over from the node where you gave the command to the other node. All FailSafe processes (node controller, application monitor, and so on) on the node where you gave the command are terminated. The other node moves to degraded state.

2. Wait for the command to finish.

3. If you want to shut down IRIS FailSafe on the second node, enter this command on that node:

   # **/etc/init.d/failsafe stop**

   All highly available services are shut down and the FailSafe processes are terminated.

Tips for Administering IRIS FailSafe Nodes

The subsections below provide information that system administrators need to know to successfully administer IRIS FailSafe nodes.

Messages from IRIS FailSafe

All messages from scripts and the IRIS FailSafe daemons go into the */var/adm/SYSLOG* file. Check this file to get information about node state changes and errors. See the section "Errors Logged to /var/adm/SYSLOG" in Appendix B for descriptions of some of the *SYSLOG* messages.

ha_admin and State Changes

When a node is in the process of changing from one state to another and you enter an *ha_admin* command, the *ha_admin* command can time out. Wait for a minute or two for the state change to complete and retry the command.

Administration Procedures to Avoid

When the IRIS FailSafe software is running, some administrative procedures should not be used because they prevent proper operation of the IRIS FailSafe system. Follow these guidelines to avoid problems:

- Do not reset the network (resetting the network is done by this sequence of commands: */etc/init.d/network stop; /etc/init.d/network start*.
- Do not set the XLV_ASSEMBLE_ARGS environment variable to change the default behavior of the *xlv_assemble* command (see the xlv_assemble(1M) reference page for information about XLV_ASSEMBLE_ARGS).
- Do not stop highly available services without first stopping IRIS FailSafe. Because IRIS FailSafe monitors the highly available services, a stopped service can appear to be a failed service and cause IRIS FailSafe to perform an undesired failover.

Exported Filesystems and Automounting

When deciding which filesystems on the non-shared disks of a node in an IRIS FailSafe cluster to export, you need to follow this rule: don't export a filesystem on a non-shared disk if it is the *parent* of a *highly available NFS filesystem*. A parent filesystem contains the mount point of another filesystem. A highly available NFS filesystem is a filesystem on a shared disk that is configured as an NFS filesystem in the IRIS FailSafe configuration file. This rule prevents problems with automounting of the highly available NFS filesystem by clients. A simple example of this rule is that /, the root directory on a node, should not be exported.

CHALLENGE RAID and xlv_assemble

If you are using a CHALLENGE RAID storage system, you must stop the RAID agent before running the *xlv_assemble* command and restart it after running *xlv_assemble*. (The */dev/scsi* device nodes can be opened by only one process at a time. As a consequence, *xlv_assemble* does not correctly assemble the failover paths to a RAID device if the RAID agent is active.)

To stop the RAID agent, enter this command:

```
# /etc/init.d/raid5 stop
```

To restart the RAID agent enter this command:

```
# /etc/init.d/raid5 start
```

Updating Replicated Files

If your cluster contains replicated files, for example, the same Netscape document root on each node, care must be taken when updating the files, for example adding new Web documents or installing a new software release. The important points are:

- Identical changes to the files must be made on both nodes.
- Changes should be made on both nodes as close to simultaneously as possible. This reduces the possibility that a failover occurs when the files are not the same and as a result clients see different pages after failover.

Educating the User Community About IRIS FailSafe

When IRIS FailSafe is running on the nodes in a cluster, users need to know these things:

- Which applications on the nodes are highly available applications
- What to do if a highly available application needs to be stopped (for example, stop IRIS FailSafe first or contact a system administrator)
- Which XLV logical volumes and XFS filesystems are on shared disks and which are on non-shared disks

- Which IP address to use for various types of access to the nodes. Because there are two types of IP addresses for nodes in an IRIS FailSafe cluster, fixed IP addresses and high availability IP addresses, users must learn which IP address to use when they access the nodes for different purposes.

Users should use a fixed IP address, such as the hostname, in these situations:

- When using the rcp command to copy files from a filesystem on a non-shared disk
- When putting NFS filesystems on non-shared disks in */etc/fstab*
- When using *automount* to access a filesystem on a non-shared disk
- When browsing a Web page whose document root is on a non-shared disk

Client users should use a high availability IP address for a node instead of its hostname in these situations that require hostnames:

- When using the *rcp* command to copy files from a filesystem on a shared disk
- When putting NFS filesystems on shared disks in */etc/fstab*
- When using *automount* to access a filesystem on a shared disk
- When browsing a Web page whose document root is on a shared disk

Getting Information About Interfaces

The *netstat -i* command tells you which interfaces are *ifconfig*'ed up and the IP addresses they own. For each interface, the first line of output from *netstat -i* for that interface lists the fixed IP address for the interface. The IP addresses on the second and later lines of output for an interface are high availability IP addresses.

Because IP aliases are *ifconfig*'ed up only when a node is in normal or degraded state, the output varies of *netstat -i* varies. The examples below from the configuration shown in Figure 2-4 show *netstat -i* output for both nodes for the three possible situations:

- IRIS FailSafe is not running.

  On xfs-ha1 each interface is listed once; the IP alias stocks isn't listed:

```
# /usr/etc/netstat -i
Name Mtu   Network     Address      Ipkts Ierrs    Opkts Oerrs  Coll
ec0  1500  190.0.2.1   xfs-ha1      71174     0      645     0   210
ec3  1500  190.0.3.1   priv-xfs-ha1  1030     0     1031     0     0
lo0  8304  loopback    localh         857     0      857     0     0
```

  On xfs-ha2 each interface is listed once; the IP alias bonds isn't listed:

```
# /usr/etc/netstat -i
Name Mtu   Network     Addres       Ipkts Ierrs    Opkts Oerrs  Coll
ec0  1500  190.0.2.2   xfs-ha2      71174     0      645     0   200
ec3  1500  190.0.3.2   priv-xfs-ha2  1030     0     1031     0     0
lo0  8304  loopback    localhost      857     0      857     0     0
```

- IRIS FailSafe is running, and both nodes are in normal state.

  On xfs-ha1 the IP alias stocks is listed for ec0, as well as its IP address xfs-ha1:

```
# /usr/etc/netstat -i
Name Mtu   Network     Address      Ipkts Ierrs    Opkts Oerrs  Coll
ec0  1500  190.0.2.1   xfs-ha1      87234     0     1445     0   218
ec0  1500  190.0.2.3   stocks       87234     0     1445     0   218
ec3  1500  190.0.3.1   priv-xfs-ha1  1090     0     1091     0     0
lo0  8304  loopback    localhost      857     0      857     0     0
```

  On xfs-ha2 the IP alias bonds is listed for ec0, as well as its IP address xfs-ha2:

```
# /usr/etc/netstat -i
Name Mtu   Network     Address      Ipkts Ierrs    Opkts Oerrs  Coll
ec0  1500  190.0.2.2   xfs-ha2      78972     0     1645     0   200
ec0  1500  190.0.2.4   bonds        78972     0     1645     0   200
ec3  1500  190.0.3.2   priv-xfs-ha2  1090     0     1091     0     0
lo0  8304  loopback    localhost      857     0      857     0     0
```

- One node (xfs-ha1) is in degraded state, and the other node (xfs-ha2) is in standby state.

  On xfs-ha1 the IP aliases stocks and bonds are both listed because the IP alias bonds has failed over from xfs-ha2:

  ```
  # /usr/etc/netstat -i
  Name Mtu   Network     Address      Ipkts Ierrs    Opkts Oerrs  Coll
  ec0  1500  190.0.2.1   xfs-ha1      87234     0     1445     0   218
  ec0  1500  190.0.2.3   stocks       87234     0     1445     0   218
  ec0  1500  190.0.2.4   bonds        87234     0     1445     0   218
  ec3  1500  190.0.3.1   priv-xfs-ha1  1090     0     1091     0     0
  lo0  8304  loopback    localhost      857     0      857     0     0
  ```

  On xfs-ha2 no IP aliases are listed:

  ```
  # /usr/etc/netstat -i
  Name Mtu   Network     Address      Ipkts Ierrs    Opkts Oerrs  Coll
  ec0  1500  190.0.2.2   xfs-ha2      78972     0     1645     0   200
  ec3  1500  190.0.3.2   priv-xfs-ha2  1030     0     1031     0     0
  lo0  8304  loopback    localhost      857     0      857     0     0
  ```

Getting a Node's State

To display the current state of a node (referred to as the node controller state in the output), enter this command on that node:

```
# /usr/etc/ha_admin -i
```

A possible return might be

```
ha_admin: Node controller state normal
```

Table 1-1 explains the possible states returned from this command.

To get the state of the other node in the cluster, enter this command:

```
# /usr/etc/ha_admin -i hostname
```

hostname is the hostname of the other node.

Getting Cluster Information

The *ha_admin -a* command provides information about the cluster including each node's state and IP aliases. Each volume, filesystem, NFS filesystem, and Web server is listed along with its primary node. For example:

```
# /usr/etc/ha_admin -a
Node controller states
          Node:            xfs-ha1      State: normal
          Node:            xfs-ha2      State: normal

Interface-pairs
Interface-pair:                  one    Owner: xfs-ha1
IP aliases in interface-pair one: stocks
Interface-pair:                  two    Owner: xfs-ha2
IP aliases in interface-pair two: bonds

XLV Volumes
    XLV volume:          shared1_vol    Owner: xfs-ha1
    XLV volume:          shared2_vol    Owner: xfs-ha2

Filesystems
    Filesystem:           shared1_fs    Owner: xfs-ha1
    Filesystem:           shared2_fs    Owner: xfs-ha2

NFS
           NFS:           export1_fs    Owner: xfs-ha1
           NFS:           export2_fs    Owner: xfs-ha2

Webservers
     Webserver:               webha1    Owner: xfs-ha1
```

This output shows an NFS filesystem being exported from each node (the NFS portion shows two NFS filesystems with different owners) and one Netscape server (Webserver). The names in the middle column of the output (xfs-ha1, one, shared1_vol, and so on) are all labels for blocks (node, interface-pair, volume, and so on) in the configuration file.

Moving a Node From Standby State to Normal or Degraded State

When a node is in standby state, it is not providing any highly available services. The **-rf** option and the **-G** option of *ha_admin* are used to tell the node to begin providing highly available services.

If both nodes are running, the *ha_admin -rf* command is used to move a node in standby state to a state that enables it to provide highly available services. The state that it moves to depends upon the state of the other node:

- If the other node is in degraded state (it is providing all highly available services for which it is the primary or backup node), both nodes move to normal state (each node provides the highly available services for which it is the primary node).
- If the other node is in standby state (it is not providing any highly available services), the node on which you enter the command moves to degraded state (it provides all highly available services for which it is the primary or backup node) and the other node remains in standby state.

On the node in standby state that you want to move to normal or degraded state, enter this command:

```
# /usr/etc/ha_admin -rf
```

If the other node is physically disconnected (no Ethernet connection, no FDDI connection, no serial line connection, and no connection to shared disk storage), you must use the *ha_admin -G* command to move the node in standby state to degraded state. In degraded state it provides all highly available services for which it is the primary or backup node. Follow this procedure:

1. Make sure that all the shared resources are not connected to the disconnected node: the disconnected node is not attached to the Ethernet or FDDI, the serial line is disconnected, and shared disk storage is disconnected.

   Note: If you enter the command *ha_admin -G* while the other node is connected, even by just a serial line, the other node is rebooted.

2. Enter the *ha_admin* command:

   ```
   # /usr/etc/ha_admin -G
   CAUTION: The -G option must only be used when the other node has
            been physically removed from the cluster. If this
            condition is not met, data integrity could be compromised.
            Read ha_admin(1m) or IRIS FailSafe release notes for more
            information.

            Continue (y/n)?
   ```

3. Respond to the prompt with **y** if you are ready to move the node to degraded state.

Moving a Node from Normal State to Standby State

The **-s** option of *ha_admin* is used to move a node from normal state to standby state (a state where it provides no highly available services). When a node is in standby state, it is said to be *removed* from the cluster. All highly available services provided by the node are failed over to the other node.

You can remove a node that is in normal state by entering a command from either node:

- Enter this command to remove this node:

  # **/usr/etc/ha_admin -s**

- Enter this command to remove the other node:

  # **/usr/etc/ha_admin -s** *hostname*

  hostname is the hostname of the other node.

Moving a Node from Degraded State to Standby State

The **-sf** option of *ha_admin* is used to move a node from degraded state (providing all highly available services for which it is the primary or backup node) to standby state (not providing any highly available services). Moving the node to standby state *removes* it from the cluster. The other node remains in standby state.

You can remove a node that is in degraded state by entering a command from either node:

- Enter this command to remove this node:

  # **/usr/etc/ha_admin -sf**

- Enter this command to remove the other node:

  # **/usr/etc/ha_admin -sf** *hostname*

  hostname is the hostname of the other node.

Moving a Node From Controlled Failback State to Normal State

The **-r** option of the *ha_admin* command is used to move a node from controlled failback state (the node does not provide highly available services and is actively monitoring the other node) to normal state (providing the highly available services for which it is the primary node). The other node (in degraded state) also moves to normal state.

You can move a node in controlled failback state to normal state by entering a command from either node:

- Enter this command to move this node from controlled failback state to normal state and the other node from degraded state to normal state:

  # **/usr/etc/ha_admin -r**

- Enter this command to move this node from degraded state to normal state and the other node from controlled failback state to normal state:

  # **/usr/etc/ha_admin -r** *hostname*

  hostname is the hostname of the node in controlled failback state.

Moving a Node From Controlled Failback State to Standby State

The **-s** option of the *ha_admin* command is used to move a node from controlled failback state to standby state (the node, which has not been providing highly available services, doesn't begin to provide them, but it does stop monitoring the other node). All the highly available services are provided by the other node (which remains in degraded state).

You can move a node from controlled failback state to standby state by entering a command from either node:

- Enter this command to move this node from controlled failback state to standby state:

  # **/usr/etc/ha_admin -s**

- Enter this command to move the other node from controlled failback state to standby state:

  # **/usr/etc/ha_admin -s** *hostname*

  hostname is the hostname of the node in controlled failback state.

Moving Heartbeat Messages to the Private Network

If there is a failure of the private network between two nodes in a cluster, the heartbeat messages that are normally transmitted on the private network are switched automatically to the public network if the parameter hb-public-ipname is set to a fixed IP address. When the problem with the private network has been resolved, you must manually switch the heartbeat messages from the public network back to the private network.

To switch the heartbeat messages from the public network back to the private network, follow this procedure on either node:

1. Bring up the private interface by entering this command:

   # **/usr/etc/ifconfig** *interface* **inet** *IPaddress1* **up**

   interface is the value of the parameter hb-private-ipname and *IPaddress1* is the value of the parameter hb-private-ipname for this node.

2. Wait 30 seconds and start a *ping* to the private IP address of the other node:

   # **ping** *IPaddress2*
   ```
   PING priv-xfs-ha1 (190.0.3.1): 56 data bytes
   ping: sendto: No route to host
   ping: wrote priv-xfs-ha1 64 chars, ret=-1
   ping: sendto: No route to host
   ping: wrote priv-xfs-ha1 64 chars, ret=-1
   64 bytes from 190.0.3.1: icmp_seq=0 ttl=255 time=1 ms
   64 bytes from 190.0.3.1: icmp_seq=1 ttl=255 time=1 ms
   ```

   (In the example output, the command was entered on xfs-ha2 and *IPaddress2* was priv-xfs-ha1.)

3. When the pings start succeeding, kill the command with EOF (**<Ctrl-d>** by default).

4. Switch the heartbeat messages back to the private network with this command:

   # **/usr/etc/ha_admin -x**

Chapter 7

Upgrading an IRIS FailSafe Cluster

Because of the tightly-coupled nature of the nodes in an IRIS FailSafe cluster, special care must be taken for both routine upgrades, such as installing new operating system releases, patches, or additional software (other than IRIS FailSafe software) or performing hardware upgrades, and IRIS FailSafe upgrades such as a new release of IRIS FailSafe or a changed configuration file. This chapter describes the upgrade procedures for an IRIS FailSafe cluster.

The major sections of this chapter are as follows:

Choosing the Correct Upgrade Procedure

Three upgrade procedures are presented in this chapter. All upgrades are done using one or more of these procedures, called A, B, and C. Table 7-1 lists the upgrades you may need to perform and the procedure you should use for each of these upgrades. If you need to perform more than one procedure, follow these guidelines:

- If you need to perform both procedure A and procedure B, with one exception you can use just procedure A: you can perform all of the A and B upgrades during procedure A.
- The exception to the first bullet is this: if you need to upgrade a node from IRIX 5.3 to IRIX 5.3 with the latest patches (a prerequisite to upgrading one or both nodes to IRIX 6.2), you must perform this upgrade first, using procedure B, before performing any other upgrades.

- You must complete all of the upgrades that require procedure A or procedure B before beginning procedure C.

Table 7-1 Upgrade Procedures

Upgrade Procedure	Upgrade
A	Upgrade IRIS FailSafe 1.0 to IRIS FailSafe 1.1 on both nodes
B	Perform a hardware upgrade on one node
B (perform once on each node)	Perform hardware upgrades on both nodes
B	Upgrade IRIX 5.3 to the latest patch level
B	Upgrade IRIX 5.3 to IRIX 6.2 on one node (IRIS FailSafe 1.1 is a prerequisite and IRIX 5.3 at the latest patch level or IRIX 6.2 on the other node is a prerequisite)
B (perform once on each node, the node that doesn't have the latest 5.3 patches must be upgraded first)	Upgrade IRIX 5.3 to IRIX 6.2 on both nodes (IRIS FailSafe 1.1 is a prerequisite and at least one of the nodes must have the latest patches installed)
B	Install software (other than IRIX and IRIS FailSafe) or IRIX patches that require a reboot on one node
B (perform once on each node)	Install software (other than IRIX and IRIS FailSafe) or IRIX patches that require a reboot on both nodes
B	Install software (other than IRIX and IRIS FailSafe) or IRIX patches that don't require a reboot on one node
B (perform once on each node)	Install software (other than IRIX and IRIS FailSafe) or IRIX patches that don't require a reboot on both nodes
C	Upgrade from a 1.0 *ha.conf* file to a 1.1 *ha.conf* (see "A Note About Configuration File Formats and Their Versions" in Chapter 4)
C	Upgrade from a 1.1 *ha.conf* file to a 1.1 *ha.conf* file with changes or additional highly available services

Upgrade Procedure A

Follow the steps below to perform procedure A. It assumes that both nodes are running IRIS FailSafe and are in normal state. Both nodes are rebooted during this procedure.

Note: If you are using this procedure to upgrade a node to IRIX 6.2, the other node must be running IRIX 5.3 with the latest patches or IRIX 6.2. See the IRIS FailSafe release notes for more information.

1. Enter these commands to shut down IRIS FailSafe on the node you want to upgrade first (*host1*) and prevent it from restarting automatically:

   ```
   # chkconfig failsafe off
   # /etc/init.d/failsafe stop
   ```

   The other node, *host2*, automatically moves into degraded state (providing all highly available services for the cluster).

2. On *host2*, enter this command to verify that it is in degraded state:

   ```
   # /usr/etc/ha_admin -i
   ha_admin: Node controller state degraded
   ```

3. Perform any hardware upgrades on *host1*.

4. Use normal software installation procedures (miniroot or *inst* from IRIX depending upon what you are installing) on *host1* to install IRIX 6.2 if desired, IRIS FailSafe 1.1, and other software such as patches and software options.

5. Reboot *host1*.

6. On *host2*, enter these commands to shut down IRIS FailSafe and prevent it from restarting automatically:

   ```
   # chkconfig failsafe off
   # /etc/init.d/failsafe stop
   ```

   At this point, no highly available services are available from either node.

7. Start IRIS FailSafe on *host1* and put it into standby state (providing no highly available services):

   ```
   # chkconfig failsafe on
   # /etc/init.d/failsafe start
   ```

8. Wait about 90 seconds and confirm that *host1* is in standby state:

   ```
   # /usr/etc/ha_admin -i
   ha_admin: Node controller state standby
   ```

9. Move *host1* to degraded state:

   ```
   # /usr/etc/ha_admin -G host1
   ha_admin: host1 successfully moved to degraded
   ```

   All highly available services are now being provided by *host1*. *host2* is automatically rebooted.

10. Perform any hardware upgrades on *host2*.

11. Use normal software installation procedures (miniroot or *inst* from IRIX depending upon what you are installing) on *host2* to install IRIX 6.2 if desired, IRIS FailSafe 1.1, and other software such as patches and software options.

12. Reboot *host2*.

13. Enter these commands on *host2* to restart IRIS FailSafe:

    ```
    # chkconfig failsafe on
    # /etc/init.d/failsafe start
    ```

    Host1 and *host2* automatically move to normal state (providing the highly available services for which they are the primary node).

14. Confirm that both nodes are in normal state by entering this command on each node:

    ```
    # /usr/etc/ha_admin -i
    ha_admin: Node controller state normal
    ```

Upgrade Procedure B

Follow the steps below to perform procedure B. It assumes that both nodes are running IRIS FailSafe 1.1 and are in normal state. This procedure upgrades a single node. To upgrade both nodes, perform the procedure twice, once on each node.

Note: If you are using this procedure to upgrade a node to IRIX 6.2, the other node must be running IRIX 5.3 with the latest patches or IRIX 6.2. See the IRIS FailSafe release notes for more information about patches.

1. Enter this command on the node you are upgrading (*host*) to put it into standby state (no highly available services running):

   ```
   # /usr/etc/ha_admin -s
   ha_admin: host successfully moved to standby
   ```

   The other node moves into degraded state (providing all highly available services).

2. Perform any hardware upgrades on *host*.
3. Use normal software installation procedures (miniroot or *inst* from IRIX depending upon what you are installing) on *host* to install the software you want to install (any software except IRIS FailSafe).
4. Reboot *host* if the software you installed requires it. (A date of Jan 1 1970 for the file */unix* indicates a reboot is required.)
5. Check the states of both nodes:

   ```
   # /usr/etc/ha_admin -i
   ha_admin: Node controller state normal
   # /usr/etc/ha_admin -i otherhost
   ha_admin: Node controller state normal
   ```

   otherhost is the hostname of the node you are not upgrading.
6. If either of the nodes are not in normal state, move them to normal state by entering this command on *host*:

   ```
   # /usr/etc/ha_admin -rf
   ha_admin: host successfully reintegrated
   ```

   Both nodes are providing the highly available services for which they are the primary node.

Upgrade Procedure C

Follow the steps below to perform procedure C. It assumes that both nodes are running IRIS FailSafe and are in normal state. All of the commands shown can be entered on either node.

1. Follow steps 1 through 8 in the section “Creating a Configuration File” in Chapter 4 to prepare a 1.1 configuration file and verify it with *ha_cfgverify*, but not install it.
2. Enter this command to put the node you want to upgrade first (*host1*) into standby state (no highly available services running) and the other node (*host2*) into degraded state (providing all highly available services):

   ```
   # /usr/etc/ha_admin -s host1
   ha_admin: host1 successfully moved to standby
   ```

   host1 is a hostname.

3. On *host1*, enter these commands to install the configuration file in */var/ha/ha.conf*.

```
# mv conffile /var/ha/ha.conf
# chown root.sys /var/ha/ha.conf
# chmod 500 /var/ha/ha.conf
# /usr/etc/ha_cfgchksum
0x12a2390e
```

4. Put the node in degraded state (*host2*) into standby state by entering this command:

```
# /usr/etc/ha_admin -sf host2
ha_admin: host2 successfully moved to standby
```

   host2 is a hostname.

   At this point, no highly available services are available.

5. Immediately put *host1* into degraded state by entering this command:

```
# /usr/etc/ha_admin -rf host1
ha_admin: host1 successfully moved to degraded
```

   All highly available services are now being provided by *host1*.

6. On *host2*, enter these commands to install the configuration file in */var/ha/ha.conf*.

```
# mv conffile /var/ha/ha.conf
# chown root.sys /var/ha/ha.conf
# chmod 500 /var/ha/ha.conf
# /usr/etc/ha_cfgchksum
0x12a2390e
```

   The checksums output by *ha_cfgchksum* on each node must be identical.

7. Move both nodes to normal state by entering this command:

```
# /usr/etc/ha_admin -rf host2
ha_admin: host2 successfully reintegrated
```

   Both nodes are providing the highly available services for which they are the primary node.

Chapter 8

Performing Simple Hardware Maintenance

This chapter explains how to replace the serial cables that connect the nodes in an IRIS FailSafe cluster. It also explains how to replace the batteries used in the remote power control unit that is used in clusters with CHALLENGE S nodes.

The sections in this appendix are as follows:

Replacing the Serial Cable

To replace the serial cable to the system controller port (CHALLENGE DM/L/XL) or to the remote power control unit (CHALLENGE S), follow these steps on the node whose serial port is connected to the cable:

1. If IRIS FailSafe is running, stop the automatic communication on the serial line by entering this command:

   ```
   # /usr/etc/ha_admin -m stop hostname
   ha_admin: Stopped monitoring the serial connection to hostname
   ```

2. Replace the defective serial cable.
3. To verify that the new cable is functioning, enter this command:

   ```
   # /usr/etc/ha_spng -i 10 -f reset-tty
   ```

   reset-tty is the value of the reset-tty parameter in the configuration file */var/ha/ha.conf*.

4. Check the return code from the command in step 3 by entering the first command if you are using *csh* and the second command if you are using *sh*:

   ```
   # echo $status
   ```

   ```
   # echo $?
   ```

   The output 0 indicates normal operation.

5. To turn back on automatic monitoring of the serial connection, enter this command:

   ```
   # /usr/etc/ha_admin -m start hostname
   ha_admin: Started monitoring the serial connection to hostname
   ```

Replacing the Private Network Cable When hb-public-ipname Is Set

If the configuration parameter hb-public-ipname is set, heartbeat messages that normally use the private network use the public network instead when the Ethernet cable that provides the private network connection is defective. When heartbeat messages are using the public network instead of the private network, use this procedure to replace the defective private network cable:

1. Replace the private network cable.
2. Follow the procedure in the section “Moving Heartbeat Messages to the Private Network” in Chapter 6 to switch the heartbeat messages back to the private network.

Replacing the Private Network Cable When hb-public-ipname Is Not Set

If hb-public-ipname is not set (heartbeat messages do not use the public network when the private network is defective), follow these steps to replace the defective private network cable:

1. Get the state of each node by entering these commands on one node:

   ```
   # /usr/etc/ha_admin -i
   ha_admin: Node controller state standby
   # /usr/etc/ha_admin -i hostname
   ha_admin: Node controller state degraded
   ```

   hostname is the hostname of the other node.

2. If the nodes aren't in standby and degraded states, move them to these states by following an appropriate procedure in Chapter 6, "Administering IRIS FailSafe."
3. Replace the private network cable.
4. Move the node that is in standby state to normal state by entering this command:

   ```
   # /usr/etc/ha_admin -r
   ```

5. Verify that both nodes are in normal state by entering these commands:

   ```
   # /usr/etc/ha_admin -i
   ha_admin: Node controller state normal
   # /usr/etc/ha_admin -i hostname
   ha_admin: Node controller state normal
   ```

   hostname is the hostname of the other node.

Replacing Batteries in the Remote Power Control Unit

The remote power control unit accepts inlet power from a standard 12 VDC-AC wall adapter. It uses eight AA alkaline batteries as an on-board uninterruptable power supply to power the unit and sustain it for up to five hours during a power outage. The batteries are in a removable battery tray, which is accessed from the front of the unit.

An LED on the battery tray alerts you that the batteries have lost sufficient power to drive the remote power control unit. This LED also lights up immediately after you power on the remote power control unit.

If the batteries fail and are not replaced, the unit retains all configuration settings in the absence of power for ten years, although it cannot control the power control units.

You can replace batteries in the remote power control unit with the unit powered on and the IRIS FailSafe cluster running (the battery tray is hot-pluggable). To install fresh batteries, follow these steps:

1. Have ready eight fresh AA alkaline batteries. Do not mix fresh and used batteries.
2. Unlock the battery tray by turning the lock screw from the **LOCK** to the **UNLOCK** position. Pull out the battery tray using its handle.
3. Remove the three screws on the outer part of the battery holder cage; slide the holder cage off. The two inner battery holders are exposed; each holds four batteries.

4. With your fingers, carefully pry the used batteries out of one holder. Replace the batteries, following the correct orientation. (Use the other holder as a guide if necessary.) Repeat the process for the second holder.
5. When you have replaced the batteries, slide the outer battery holder cage back onto the battery tray, and replace the screws.
6. Replace the battery tray in the remote power control unit and turn the lock screw back to the **LOCK** position.

Appendix A

Messages About Configuration File Errors

This appendix lists error messages and warnings that can appear as output of the *ha_cfgverify* command. This command aids in verifying the validity of the configuration file *ha.conf*. *ha_cfgverify* can be run manually (see the section "Creating a Configuration File" in Chapter 4) and is run automatically each time IRIS FailSafe is started on a node and when a node moves from standby state to another state.

Note: *ha_cfgverify* generates warnings when a possible error exists, but it is unable to determine for sure. You must check each warning manually. A warning could indicate invalid IRIS FailSafe configuration.

The error messages are listed in alphabetical order.

`ha_cfgverify: action-timer section must be present for application-class <name>`

Every application class must have an action-timer section where different time intervals for monitoring the application are specified. The name of this section must be same as the application class name.

`ha_cfgverify: Agent <agent_name> is not valid in <app_class> application class`

The monitoring agent *agent_name* specified in the application class *app_class* does not exist or does not have execute permissions.

`ha_cfgverify: all nodes must be present in the server nodes entry for main application class`

Both nodes need server-node entries for main application class.

`ha_cfgverify: <name> application class does not have a server node`

All application class entries must have at least one server-node entry.

```
ha_cfgverify: <application> application class: start-monitor-time must be
greater than the long-timeout value
```

For application class section labelled application, start-monitor-time must be greater than the long-timeout value.

```
ha_cfgverify: backup-node entry in filesystem <name> must be present
```

The filesystem section must have a server-node and a backup-node.

```
ha_cfgverify: backup-node entry <node name> in filesystem <name> is not
a valid node
```

The backup node does not have a node section in the configuration file.

```
ha_cfgverify: backup-node entry in volume <node_name> is not a valid node
```

The backup node name must match one of the node block labels.

```
ha_cfgverify: broadcast address <address> is not valid for node
<node_name>
```

The broadcast address is either not in Internet standard "." notation or the name lookup of the broadcast address failed.

```
ha_cfgverify: broadcast address for interface block <block_name> is not
present for node <node_name>
```

The broadcast address must be specified for the interface block *block_name*.

```
ha_cfgverify: broadcast address for primary-ip not present for node <node
name>
ha_cfgverify: broadcast address for private-ip not present for node <node
name>
ha_cfgverify: broadcast address for secondary-ip not present for node
<node
name>
```

The broadcast address for primary-ip, secondary-ip, and private-ip must be specified in the *ha.conf* file.

```
ha_cfgverify: broadcast-addr entry <addr> is not valid in interface-pair
<ipair>
```

The broadcast address *addr* specified in interface pair section labelled *ipair* is not a valid internet address or the name lookup of the broadcast address failed.

`ha_cfgverify: broadcast-addr entry in interface-pair <ipair> must be present`

This entry specifies broadcast address for the list of IP addresses in interface-pair.

`ha_cfgverify: Check if rpc.statd has been started with -h option`

The *rpc.statd* process must be started with **-h** option. In IRIX 6.2, check if the */etc/config/statd.options* file has **-h** option. In IRIX 5.3, check if the latest networking rollup patch has been installed.

`ha_cfgverify: Configuration file <filename> opening/parsing error`

There was an error in opening or parsing the configuration file. Check the previous error message from *ha_cfgverify* for explanation.

`ha_cfgverify: device name entry in volume <vol_name> must be present`

All volume sections must have device-name entries.

`ha_cfgverify: Failed to find hostname`

The file */etc/sys_id* must have the hostname. The hostname must be configured using hostname *command hostname -s <host name>*.

`ha_cfgverify: filesystem type entry in filesystem <name> must be present`

All filesystem sections must have mount point, filesystem type, device name, and mount mode entries.

`ha_cfgverify: <type> filesystem type is not valid in filesystem <name>`

The filesystem must be XFS.

`ha_cfgverify: giveaway function <file name> is not valid in <name> application class`
`ha_cfgverify: giveback function <file name> is not valid in <name> application class`

Either the filename is not present in the node or it does not have execute permission.

`ha_cfgverify: heartbeat ip address not present for node <node name>`

The heartbeat IP address for the node must be specified. It is usually the private IP address.

`ha_cfgverify: heartbeat ip address <ip address> is not valid for the node <node name>`

The heartbeat IP address must be the private IP address. It must be same for all the nodes, that is, heartbeat IP address for all nodes must be the respective private IP addresses.

`ha_cfgverify: heartbeat lost count for the node <node name> must be specified`

The heartbeat lost count must be specified for each node.

`ha_cfgverify: heartbeat probe time for the node <node name> must be specified`

The heartbeat probe time (in seconds) must be specified for each node.

`ha_cfgverify: heartbeat timeout for the node <node name> must be specified`

The heartbeat timeout (in seconds) must be specified for each node.

`ha_cfgverify: heartbeat private ip name is not present for node <node_name>`

The heartbeat IP address for the node must be specified.

`ha_cfgverify: heartbeat public ip name <public_ip> is not present for node <node_name>`

If the private network between the nodes fail, the hb-public-ipname is used for heartbeat. This parameter must be set to a fixed IP address.

`ha_cfgverify: httpd-script <script> is not valid in <app> application`

The httpd-script value *script* specified for the webserver application is not present in the node or it does not have execute permission. This script is used to start and stop the Web server.

`ha_cfgverify: hostname <hostname> must be one of the node labels`

The node label of the node section must be the node's hostname. The *ha_cfgverify* command must be run in one of the nodes of the cluster.

`ha_cfgverify: interface block <block_name> does not have MAC-address entry`

The *re-mac*'ing option has been set to true but the MAC address has not been specified in the interface block.

`ha_cfgverify: interface block name <name> must be unique for the configuration file`

The name of the interface block in the node section must be unique for the configuration file.

`ha_cfgverify: interface name <ifname> for node <node_name> must be configured in /etc/config/netif.options file`

The interface name *ifname* must be assigned an IP address in the */etc/config/netif.options* file. This interface must be configured when the node boots up.

`ha_cfgverify: interface name for primary-ip not present for node <node name>`
`ha_cfgverify: interface name for private-ip not present for node <node name>`
`ha_cfgverify: interface name for secondary-ip not present for node <node name>`

The interface name for primary-ip, secondary-ip, and private-ip must be specified.

`ha_cfgverify: interface name <name> is not valid for node <node_name>`

The interface name is not valid in the node. Check the interface name using the netstat command.

```
ha_cfgverify: <agent_name> : interface-agent section must have
start-monitor-time entry
ha_cfgverify: <agent_name> : interface-agent section must have
interface-probe-interval entry
ha_cfgverify: <agent_name> : interface-agent section must have
interface-probe-timeout entry
ha_cfgverify: <agent_name> : interface-agent section must have
remote-send-probe-interval entry
ha_cfgverify: <agent_name> : interface-agent section must have
remote-send-timeout entry
```

Interface agent section must have all the following entries: start-monitor-time, interface-probe-interval, interface-probe-timeout, remote-send-probe-interval, and remote-send-timeout.

```
ha_cfgverify: interface-agent section must have interface-probe-interval
entry
ha_cfgverify: interface-agent section must have interface-probe-timeout
entry
ha_cfgverify: interface-agent section must have
remote-send-probe-interval entry
ha_cfgverify: interface-agent section must have start-monitor-time entry,
```

The interface-agent section must have interface-probe-interval, interface-probe-timeout, remote-send-probe-interval, start-monitor-time entries.

```
ha_cfgverify: internal section: version-major value is invalid (must be
<num>)
```

The value of version-major parameter in the internal section must be the number shown.

```
ha_cfgverify: internal section: version-minor value is invalid (must be
<num>)
```

The value of version-minor parameter in the internal section must be the number shown.

```
ha_cfgverify: ip address <value> is not present in the hosts database
ha_cfgverify: ip alias <value> is not present in the hosts database
ha_cfgverify: heartbeat private ip name <value> is not present in the
hosts database
ha_cfgverify: node name <node_name> is not present in the hosts database
ha_cfgverify: node name <node_name> is not present in the hosts database
```

The ip-address, ip-alias or node name could not be found in the hosts database. The host-name or address should be found either in the */etc/hosts* file, from NIS, or *named* server.

```
ha_cfgverify: ip address for interface block <block_name> is not present
for node <node_name>
```

The interface block of the node section must have an entry for ip-address that is a fixed IP address. This IP address is configured using */etc/config/netif.options*.

```
ha_cfgverify: ip address for primary-ip not present for node <node name>
ha_cfgverify: ip address for private-ip not present for node <node name>
ha_cfgverify: ip address for secondary-ip not present for node <node name>
```

The IP address for the primary-ip, secondary-ip, and private-ip must be specified in the *ha.conf* file.

```
ha_cfgverify: ip-aliases field must be present for nfs server node
<server_node>
```

The server-node section of application-class nfs must contain the ip-aliases parameter. NFS clients use one of the ip-aliases specified here for mounting filesystems from this server.

```
ha_cfgverify: kill function <file name> is not valid in <name> application
class
```

Either the filename is not present in the node or it does not have execute permission.

```
ha_cfgverify: local monitor <file name> is not valid in <name> application
class
```

Either the filename is not present in the node or it does not have execute permission.

```
ha_cfgverify: local monitor probe time not specified for <name>
application class
ha_cfgverify: local monitor timeout not specified for <name> application
class
```

If the application has a local monitor, local monitor timeout and probe time in seconds must be specified.

```
ha_cfgverify: long-timeout value missing
```

The long-timeout value must be specified in the internal section of the *ha.conf* file.

```
ha_cfgverify: mac address for primary-ip not present for node <node_name>
ha_cfgverify: mac address for private-ip not present for node <node_name>
ha_cfgverify: mac address for secondary-ip not present for node
<node_name>
```

If re-mac'ing of interfaces is needed (the re-mac variable in the system-configuration section of the configuration file is set to true), the MAC address of primary-ip, private-ip, and secondary-ip for the node *node_name* must be specified.

```
ha_cfgverify: MAC-address %s is not valid for node <node_name>
```

MAC address specified in the interface section of *node_name* does not match the physical address of network interface. The MAC address must be obtained using *macconfig* command.

```
ha_cfgverify: main application class does not have all server nodes
```

Either the main application class does not have server-node entries, or there is no main application class entry in application class section.

```
ha_cfgverify: main application class must have giveaway function
ha_cfgverify: main application class must have giveback function
ha_cfgverify: main application class must have kill function
ha_cfgverify: main application class must have local monitor function
ha_cfgverify: main application class must have takeback function
ha_cfgverify: main application class must have takeover function
```

The main application class must have *giveback*, *takeback*, *takeover*, *giveaway*, *kill*, and local monitor scripts.

```
ha_cfgverify: main application class must have start monitor time
```

The start monitor time entry for main application class must be specified in seconds.

```
ha_cfgverify: mount mode entry in filesystem <name> must be present
ha_cfgverify: mount point entry in filesystem <name> must be present
```

All filesystem sections must have mount point, filesystem type, volume, and mount mode entries.

```
ha_cfgverify: mount point entry <name> in filesystem <name> is present in
/etc/fstab file
```

The mount point must not be present in the */etc/fstab* file. If it is present in the */etc/fstab* file, the “noauto” mount option must be specified. The IRIS FailSafe software mounts the directory.

```
ha_cfgverify: name of the interface must be present for the interface
block <name>
ha_cfgverify: netmask for interface block <name> is not present for node
<node_name>
```

The interface block in the node section must specify the interface name and netmask.

```
ha_cfgverify: netmask <mask> is not valid for node <node_name>
ha_cfgverify: netmask entry <mask> is not valid in interface-pair
<pair_name>
```

The netmask entry *mask* must be a valid mask for an internet address.

```
ha_cfgverify: netmask entry in interface-pair <name> must be present
```

Every interface-pair block must have a netmask entry. It is missing from the interface-pair block name.

```
ha_cfgverify: netmask for primary-ip not present for node <node name>
ha_cfgverify: netmask for private-ip not present for node <node name>
ha_cfgverify: netmask for secondary-ip not present for node <node name>
```

The netmask for the primary-ip, secondary-ip, and private-ip must be specified in the *ha.conf* file.

`ha_cfgverify: nfs is not configured in the system`

Either the NFS software has not been configured in the system, or the *nfs* configuration flag is not *chkconfig*'d on.

`ha_cfgverify: NFS lockd/statd FailSafe compatible patch is not installed in the system`

The NFS *lockd*/*statd* patch required by IRIS FailSafe is not installed.

`ha_cfgverify: nfs <name> section: export point entry is invalid`

The export point entry must be a subdirectory of the mount-point entry of the filesystem.

`ha_cfgverify: nfs <name> section: export point entry <dir> or its subdirectory is present in /etc/exports file`

The export point entry must not be present in the */etc/exports* file. It is exported by the IRIS FailSafe software.

```
ha_cfgverify: nfs <name> section must have the export info entry
ha_cfgverify: nfs <name> section must have the export point entry
ha_cfgverify: nfs <name> section must have the filesystem entry
```

All NFS sections must have filesystem, export-point, and export-info entries.

`ha_cfgverify: nfs <name>: the filesystem entry <filesystem name> is invalid`

The filesystem entry does not match the filesystem section labels.

`ha_cfgverify: no application class section`

The *ha.conf* file does not have an application class section. The application class has an entry for main application and optional entries for NFS and Web server applications.

`ha_cfgverify: no interface section for the node <node_name>`

Every node block must have an interface section. Interface section is missing for the node block labelled *node_name*.

`ha_cfgverify: no interface-pair section`

The configuration file is missing the interface-pair section. The name must be unique in the IRIS FailSafe configuration file. There should be one interface-pair block for each combination of network interfaces.

`ha_cfgverify: No ip-aliases entry in interface-pair <name>`

The interface-pair section *name* does not contain the parameter ip-aliases. It contains the list of ip-addresses that are failed over.

`ha_cfgverify: No nfs section present in the file`

There is an NFS application class entry but there is no NFS section.

`ha_cfgverify: No node sections in the configuration file`

The *ha.conf* file must have two node sections. The name of the node block must be the hostname of the node.

`ha_cfgverify: No webserver section present in the file`

The configuration file has a Web server application class entry, but there is no Web server section.

`ha_cfgverify: node name <node_name> is not present in the hosts database`

The *node_name* must be present in the */etc/hosts* file, or its IP address can be obtained from the *named* server.

`ha_cfgverify: <node_name> section: hb_use_public entry must be set to "no" in hot standby configuration.`

In an active/backup configuration (each node in the cluster has only two network interfaces), the hb_use_public entry must be set to **no**.

`ha_cfgverify: primary-interface entry <p_interface> in interface-pair <pair_name> is invalid`
`ha_cfgverify: secondary-interface entry <p_interface> in interface-pair <pair_name> is invalid`

The value assigned to primary-interface and secondary-interface parameter must match one of the interface block labels of the node block.

```
ha_cfgverify: primary-interface entry in interface-pair <pair_name> must
be present
ha_cfgverify: secondary-interface entry in interface-pair <pair_name> be
present
```

Every interface-pair block must have entries for primary-interface and secondary-interface. The interface-pair labelled *pair_name* is missing the indicated entry.

```
ha_cfgverify: primary-interface entry in interface-pair <pair_name>
require both ip-aliasing and re-mac'ing. Re-macing requires a dedicated
secondary interface.
```

On a failure, IRIS FailSafe moves the MAC address of an interface along with the IP addresses. But, MAC addresses can be moved only between one set of primary and secondary interfaces. Check if the primary-interface entry has been specified in multiple interface-pair blocks.

```
ha_cfgverify: primary-ip and secondary-ip information must be present for
the node <node_name>
```

The primary-ip and secondary-ip information is needed for all nodes in a dual-active configuration. In an active/backup configuration, primary-ip information is needed for one node and secondary-ip information is needed for the other node.

```
ha_cfgverify: primary-ip broadcast address <address> is not valid for
node <node name>
```

The broadcast address is not a valid internet address.

```
ha_cfgverify: primary-ip information must be present for the node
<node_name>
```

The primary-ip and secondary-ip information is needed for all nodes in a dual-active configuration. In an active/backup configuration, primary-ip information is needed for one node and secondary-ip information is needed for the other node.

```
ha_cfgverify: primary-ip interface name <interf. name> for node <node
name> is present in /etc/config/netif.options file
```

The primary-ip interface must not be configured using the *netif.options* file. The IRIS FailSafe software configures the interface.

`ha_cfgverify: primary-ip interface name <interface name> is not valid for node <node name>`

The interface name is not valid in the node. Check the interface name using the *netstat* command.

`ha_cfgverify: primary-ip mac address <mac_address> is not valid for node <node_name>`

The MAC address / physical address specified for primary-ip and secondary-ip is incorrect. Use *macconfig*(1M) to find the correct physical address.

`ha_cfgverify: primary-ip netmask <net mask> is not valid for node <node name>`

The netmask for the primary-ip, secondary-ip, or private-ip is not a valid internet netmask.

`ha_cfgverify: private-ip broadcast address <address> is not valid for node <node name>`

The broadcast address is not a valid internet address.

`ha_cfgverify: private-ip interface name <interface name> for node <node name> must be present in /etc/config/netif.options file`

The private-ip must be specified in the *netif.options* file.

`ha_cfgverify: private-ip interface name <interface name> is not valid for node <node name>`

The interface name is not valid in the node. Check the interface name using the *netstat* command.

`ha_cfgverify: private-ip name <ip_name> for node <node_name> must match the hostname`

The private-ip name in the node section of the configuration file must be same as the hostname of that node.

`ha_cfgverify: private-ip netmask <net mask> is not valid for node <node name>`

The netmask for the primary-ip, secondary-ip, or private-ip is not a valid internet netmask.

```
ha_cfgverify: raw device name entry in filesystem <fs> must be present
ha_cfgverify: mount point entry in filesystem <fs> must be present
ha_cfgverify: filesystem type entry in filesystem <fs> must be present
ha_cfgverify: volume-name entry in filesystem <fs> must be present
ha_cfgverify: server-node entry in filesystem <fs> must be present
ha_cfgverify: backup-node entry in filesystem <fs> must be present
ha_cfgverify: device name entry in filesystem <fs> must be present
ha_cfgverify: mount mode entry in filesystem <fs> must be present
```

All filesystem sections must have raw device, mount point, filesystem type, volume name, server node, backup node, device name, and mount mode entry in them.

```
ha_cfgverify: raw device name entry in filesystem <fs> is invalid
ha_cfgverify: Warning: device name entry in filesystem <fs> is invalid
```

The device names do not exist or they are not special device names in the node.

```
ha_cfgverify: remote monitor probe time not specified for <name>
application class
ha_cfgverify: remote monitor timeout not specified for <name> application
class
```

If the application has a remote monitor, the remote monitor timeout and probe time (in seconds) must be specified.

```
ha_cfgverify: secondary-ip broadcast address <address> is not valid for
node <node name>
```

The broadcast address is not a valid internet address.

```
ha_cfgverify: secondary-ip information must be present for the node
<node_name>
```

The primary-ip and secondary-ip information is needed for all nodes in a dual-active configuration. In an active / backup configuration, primary-ip information is needed for one node and secondary-ip information is needed for the other node.

```
ha_cfgverify: secondary-ip interface name <interface name> for node <node
name> must be present in /etc/config/netif.options file
```

The secondary-ip must be specified in the *netif.options* file.

```
ha_cfgverify: secondary-ip interface name <interface name> is not valid
for node <node name>
```

The interface name is not valid in the node. Check the interface name using the *netstat* command.

```
ha_cfgverify: secondary-ip mac address <mac_address> is not valid for
node <node_name>
```

The MAC address / physical address specified for primary-ip and secondary-ip is incorrect. Use *macconfig*(1M) command to find the correct physical address.

```
ha_cfgverify: secondary-ip netmask <net mask> is not valid for node <node
name>
```

The netmask for the primary-ip, secondary-ip, or private-ip is not a valid internet netmask.

```
ha_cfgverify: <name> section: hb_use_public entry must be set to “no” in
active/standby configuration
```

In active / backup configuration hb-use-public entry must be set to **no**.

```
ha_cfgverify: server-node entry <node name> in filesystem <name> is not
a valid node
```

The server-node does not have a node section.

```
ha_cfgverify: server-node entry in volume <vol_name> is must be present
ha_cfgverify: backup-node entry in volume <vol_name> must be present
ha_cfgverify: device name entry in volume <vol_name> must be present
```

The volume block must have server-node, backup-node, device name entry. This entry is missing for volume block labelled *vol_name*.

```
ha_cfgverify: server-node entry <node_name> in volume <vol_name> is not
a valid node
ha_cfgverify: backup-node entry <node_name> in volume <vol_name> is not
a valid node
```

Server-node and backup-node value must match one of the node labels.

```
ha_cfgverify: server-node entry <node_name> in filesystem <fs> is not a
valid node
ha_cfgverify: backup-node entry <node_name> in filesystem <fs> is not a
valid node
ha_cfgverify: server node <node_name> is not a valid node in main
application class
```

The *node_name* must match one of the node labels.

```
ha_cfgverify: server node <node name> is not a valid node in <name>
application class
```

The server-node in the application class does not have a node section.

```
ha_cfgverify: short-timeout value must be smaller than long-timeout value
```

The long-timeout value in the internal section must be larger than the short-timeout value.

```
ha_cfgverify: Some xlv patches have not been installed in the system
```

All XLV patches must be installed on the node. Check the release notes for the latest XLV patch numbers.

```
ha_cfgverify: specify a reset-host, or a reset-tty for node <node_name>
```

reset-tty is the device filename of the serial link that is connected to remote power control unit.

```
ha_cfgverify: start-monitor-time (action-timer block) must be present for
application-class <app_class>
```

The action-timer block of application-class *app_class* does not contain the entry for start-monitor-time.

```
ha_cfgverify: statmon-dir <dir_name> must be called “statmon”
```

The name of the statmon directory must be **statmon**.

```
ha_cfgverify: statmon-dir must be present for nfs server node <node_name>
```

The statmon-dir parameter in the application classes block of the configuration file must be specified for the NFS primary node *node_name*.

```
ha_cfgverify: takeback function <file name> is not valid in <name>
application class
ha_cfgverify: takeover function <file name> is not valid in <name>
application class
```

Either the filename is not present in the node or it does not have execute permission.

```
ha_cfgverify: The /etc/config/routed.options must have -q option
```

The routed daemon must be started with the **-q** option. The routed options file */etc/config/routed.options* must have the **-q** option.

```
ha_cfgverify: The ip-alias <value> in interface-pair <name> is a fixed ip
address.
```

The ip-alias *value* in the interface-pair name cannot be a fixed IP address. It must be an IP alias that can be failed over.

```
ha_cfgverify: Too many application classes: <num>
```

The IRIS FailSafe software supports up to 16 application class entries.

```
ha_cfgverify: Too many filesystems <n>
```

The number of filesystems n specified in the configuration file is greater than maximum filesystems allowed.

```
ha_cfgverify: Too many node entries: <num>. Only 2 two nodes are permitted
```

The configuration file *ha.conf* can have only two node sections. The IRIS FailSafe software permits only clusters of two nodes.

```
ha_cfgverify: Too many volumes <n>
```

The number of volumes specified in the configuration file is greater than maximum volumes allowed.

```
ha_cfgverify: /usr/etc/ha_statd command missing
```

The command */usr/etc/ha_statd* is missing on the node.

```
ha_cfgverify: volume-name entry <vol_name> in filesystem <fs> is not a
valid volume.
```

The volume-name value must match one of the volume block labels.

`ha_cfgverify: Warning: check if xlv license has been installed`

If XLV plexing is being used, a plexing license must be installed in the */etc/nodelock* file.

`ha_cfgverify: Warning: device name entry in filesystem <name> is invalid`

Either the filename is not present in the node or it does not have execute permission.

`ha_cfgverify: Warning: mount point entry <name> in filesystem <name> is not a valid directory`

The mount point must be a valid directory.

`ha_cfgverify: Warning: netscape is not configured in the system`

Either the Web server software has not been configured in the system, or the *ns_httpd* configuration flag is not *chkconfig*'d on.

`ha_cfgverify : Warning NFS lockd/statd patch for Failsafe (1032) is not installed in the system`

The NFS *lockd/statd* patch (see the release notes for the patch number) is necessary for failing over NFS locks for the NFS service.

`ha_cfgverify: Warning: no filesystems found in the configuration file`

The *ha.conf* file does not have a filesystem section. Most IRIS FailSafe system configurations require one. Check if your system configuration requires a filesystem section in the *ha.conf* file.

`ha_cfgverify: webserver <name> : httpd-optionsfile and httpd-script must be specified`

Either both httpd-optionsfile and httpd-script must be specified or both must not be specified.

`ha_cfgverify: webserver <section_name> section must have the backup-node entry`
`ha_cfgverify: webserver <section_name> section must have the server-node entry`

The webserver *section_name* block of the configuration file must have server-node and backup-node entries.

```
ha_cfgverify: webserver <section_name> : the backup-node entry
<node_name> is invalid
ha_cfgverify: webserver <section_name> : the server-node entry
<node_name> is invalid
```

The server-node and backup-node entries must have a node section in the configuration file.

Appendix B

System Troubleshooting

This appendix explains how to troubleshoot system problems.

The major sections in this appendix are as follows:

General Troubleshooting Procedure

When you encounter a failure, follow this general procedure:

1. Use *df* to make sure that all filesystems on shared disks are mounted only on one node; no filesystem should be simultaneously mounted by two nodes.
2. Look in */var/adm/SYSLOG* on both nodes for causes of failure.
3. Diagnose and repair the problem using the information in the remainder of this appendix.
4. If the failure caused a failover and the failed node is in standby state, after repairing the problem you can bring both nodes back to normal state by following the procedure in the section "Moving a Node From Standby State to Normal or Degraded State" in Chapter 6.

IRIS Failsafe System Does Not Start

If the IRIS FailSafe system does not start, follow these steps:

1. Make sure that IRIS FailSafe is *chkconfig*'d on.

   Note: If two failovers have occurred within a period of time specified by MIN_UPTIME in */etc/init.d/failsafe*, 300 seconds by default, IRIS FailSafe software automatically performs *chkconfig failsafe off*.

2. Make sure that */var/ha/ha.conf* is identical on both nodes in the cluster by entering this command on each node:

   ```
   # /usr/etc/ha_cfgchksum
   0x12a2390e
   ```

   The checksums output by the commands should be identical.

3. Check the format and contents of */var/ha/ha.conf* using the *ha_cfgverify* command:

   ```
   # /usr/etc/ha_cfgverify
   ```

   Ensure that there are no errors from *ha_cfgverify*.

4. Verify the network interfaces and serial connections using the procedures in the sections "Testing the Public Network Interfaces" and "Testing the Serial Connections" in Chapter 5.

5. Look at */var/adm/SYSLOG* to see what errors are printed out by the IRIS FailSafe daemons. When a node in the IRIS FailSafe cluster starts up normally, the following *SYSLOG* messages appear:

```
ha_appmon[6141]: Received XRELEASE_PEER
ha_nc[6135]: Received JOINING
ha_nc[6135]: New state: NC_JOINING
ha_nc[6135]: Received REJOIN
ha_appmon[6141]: Received XACQUIRE
ha_appmon[6141]: Received START_REMMON
ha_nc[6135]: New state: NC_NORMAL
```

Duplicate SCSI IDs

If you see SCSI bus-related errors after configuring the cluster, or nonexistent devices show up in *hinv*, follow these steps:

1. Verify that the SCSI host IDs of the two nodes are different (by convention, 0 and 2) by running *nvram*.

   To change the SCSI host ID enter this command:

   # **nvram -v scsihostid** *id*

2. Verify that the SCSI IDs of all disks and other peripherals on the same SCSI bus have distinct SCSI unit numbers and that they are different from the SCSI host IDs of the two nodes in the cluster (usually 0 and 2).

   Note: This convention works when only one internal disk per node is used. If a second internal disk is used in either node, that disk's SCSI ID cannot be identical to either SCSI host ID.

Trouble Accessing a Network Interface

If you cannot access a network interface, check if the interface is configured up and the network interface is working as described below.

The procedure uses this *ha.conf* fragment as an example:

```
node xfs-ha1
{
        interface xfs-ha1-ec0
        {
                name = ec0
                ip-address = 190.0.2.2
                netmask = 0xffffff00
                broadcast-addr = 190.0.2.255
        }

...
}
```

Follow these steps:

1. Get information about the interface from this *ifconfig* command:

   ```
   # /usr/etc/ifconfig ec0
   ec0: flags=c63<UP,BROADCAST,NOTRAILERS,RUNNING,FILTMULTI,MULTICAST>
           inet 190.0.2.2 netmask 0xffffff00 broadcast 190.0.2.255
   ```

   The UP in the first line of output indicates that the interface is configured up.

2. If the interface is not configured up, add the interface to the */etc/config/netif.options* file as described in the section “Configuring Interfaces” in Chapter 3.

3. If the interface is configured up, check if the network interface is working by entering this command:

   ```
   # /usr/etc/netstat -I ec0
   ```

   Check the output to see if there are input errors or output errors for the interface.

Trouble Accessing a Node Over the Network

If you cannot access a node using the network, run *netstat -i* as described in the section "Getting Information About Interfaces" in Chapter 6 to see if the IP address to which you are trying to connect is configured on one of the node's public interfaces.

Note: You cannot access an IP address associated with a private interface from a node on the public network.

Because high availability IP addresses are configured by IRIS FailSafe, they might not be configured if IRIS FailSafe is not started. Also, the IP address might have been taken over by the other node.

Trouble With the Serial Connection

If you suspect a problem with one of the serial cables connected to the remote power control unit or to the system controller port of the other node (possibly because you have received mail that indicates a problem), use the procedure below to determine if there is a problem. If you suspect a particular cable, perform this procedure on the node whose serial port is connected to the cable. If necessary, follow this procedure on both nodes.

1. To stop the automatic communication on the serial line, enter this command:

   ```
   # /usr/etc/ha_admin -m stop hostname
   ha_admin: Stopped monitoring the serial connection to hostname
   ```

2. Manually send messages on the serial line by entering this command:

   ```
   # /usr/etc/ha_spng -i 10 -f reset-tty
   ```

   reset-tty is the value of the reset-tty parameter in the configuration file */var/ha/ha.conf*.

3. Check the return code from the command in step 2,

   ```
   # echo variable
   0
   ```

   If you are using *csh*, *variable* is **$status**. If you are running *sh*, *variable* is **$?**. The zero output indicates normal operation.

4. If the return code was zero, restart automatic communication on the serial line by entering this command:

   ```
   # /usr/etc/ha_admin -m start hostname
   ha_admin: Started monitoring the serial connection to hostname
   ```

   There is no problem with the serial connection, so skip the remainder of this procedure.

5. If the return code from *ha_spng* was non-zero, try re-seating the serial cable connectors.

6. Re-test the serial line by performing steps 2 and 3.

7. If re-seating the cables didn't work, replace the cable by following the directions in the section "Replacing the Serial Cable" in Chapter 8.

Trouble With Volumes

To check that a node is licensed for plexing of XLV logical volumes, XLV logical volumes are visible to the node, and XLV logical volumes are owned by the correct node, follow these steps:

1. Verify that the node is licensed for plexing and that the plexing software is installed:

   ```
   # xlv_mgr
   xlv_mgr> show config
   Allocated subvol locks: 30 locks in use: 7
   Plexing license: present
   Plexing support: present
   Maximum subvol block number: 0x7fffffff
   ```

   If you have just installed the plexing software, you must reboot the system for the plexing support to be included in the kernel.

2. Verify that the node sees the volume. In an IRIS FailSafe environment, a node accesses (and mounts filesystems on) only those XLV volumes that it owns. To see all the volumes in the cluster, enter these commands:

   ```
   xlv_mgr> show all
   Volume: vol1 (complete)
   Volume: vol2 (complete)
   Volume: shared_vol (complete)
   xlv_mgr> quit
   ```

   This command shows all the XLV volumes in the cluster.

3. If the cluster is using a CHALLENGE RAID storage system, stop the RAID agent by entering this command:

   ```
   # /etc/init.d/raid5 stop
   ```

4. To see volumes owned by a node, enter this command on that node:

   ```
   # xlv_assemble -ln

   VOL vol2        flags=0x1, [complete]
   DATA    flags=0x0()     open_flag=0x0() device=(192, 4)
   PLEX 0  flags=0x0
   VE 0    [active]
           start=0, end=687999, (cat)grp_size=1
           /dev/dsk/dks5d9s0 (688000 blks)
   PLEX 1  flags=0x0
   VE 0    [active]
           start=0, end=687999, (cat)grp_size=1
           /dev/dsk/dks5d9s1 (688000 blks)
   ```

   This command displays only the volumes owned by this node.

5. If a volume is owned by the wrong node, change the ownership of a volume (for example, to make *vol2* owned by *xfs-ha2*) using *xlv_mgr* after first unmounting the filesystem mounted on that volume (*vol2*):

   ```
   # umount /vol2
   # xlv_mgr
   xlv_mgr> change nodename xfs-ha2 vol2
   set node name "xfs-ha2" for object "vol2" done
   ```

6. Run *xlv_assemble -l* on both nodes:

   ```
   # xlv_assemble -l
   ```

7. Restart the RAID agent if you stopped it in step 3 by entering this command:

   ```
   # /etc/init.d/raid5 start
   ```

Trouble Mounting Filesystems

If you are having trouble mounting filesystems on shared disks, execute the mount directive that would be executed by the IRIS FailSafe software on each node. Follow these steps:

1. In your */var/ha/ha.conf* file, look for the filesystem block for a filesystem that does not mount successfully, for example:

```
filesystem fs1
{
        mount-point = /shared
        mount-info
        {
                fs-type = xfs
                volume-name = vol1
                mode = rw,noauto
        }
}
```

   This filesystem block is for the filesystem mounted at */shared*.

2. Look for the volume block for this filesystem; it is the volume block whose label is vol1, the value of volume-name, for example:

```
volume vol1
{
        server-node = xfs-ha1
        backup-node = xfs-ha2
        devname = /dev/dsk/xlv/vol1
        devname-owner = root
        devname-group = sys
        devname-mode = 0600
}
```

3. Follow the procedure in the section “Trouble Accessing a Network Interface” in this appendix to make sure that the volume is owned by this node.

4. Mount the filesystem with this command:

   # **mount -txfs -rw,noauto /dev/dsk/xlv/vol1 /shared**

5. Unmount the filesystem with this command:

   # **umount /shared**

   Note: Do not omit this step. Data corruption could result.

6. Repeat steps 1 through 5 for every filesystem that doesn’t mount successfully.

7. On the other node, repeat steps 1 through 6. While following the procedure in the section “Trouble Accessing a Network Interface,” do change the owner of the volume to the second node.

Trouble Accessing a Filesystem Over NFS

If you cannot access a filesystem on a shared disk over NFS, make sure that the network interface is *ifconfig*’d up and the filesystem is mounted, as explained in the sections “Error Message from ha_statd” and “Trouble Mounting Filesystems” in this appendix. Also verify that IP alias you are using is specified correctly in the configuration file */var/ha/ha.conf*.

The procedure below exports the filesystem manually from both nodes and checks to see if a client can access it. It uses this *ha.conf* fragment as an example:

```
nfs shared1
{
        filesystem = shared1
        export-point = /shared1
        export-info = rw
}
```

Follow these steps:

1. Verify that */shared1* is mounted. If it is not, follow instructions in the section “Trouble Accessing a Network Interface” in this appendix.
2. Export the filesystem by entering this command:

   ```
   # exportfs -i -o rw /shared1
   ```

3. Verify that the filesystem is exported:

   ```
   # exportfs
   /shared1 -rw
   ```

4. From a client, mount the filesystem and verify that you can access it.
5. Unexport and unmount the filesystem:

   ```
   # exportfs -u /shared1
   # umount /shared1
   ```

   Note: Do not omit this step. Data corruption could result.

6. Repeat steps 1 through 5 with the filesystem mounted on the other node.

Netscape Server Warning Messages at Startup

This type of error message is normal when nodes in the cluster boot up:

```
error: could not bind to 190.0.2.1 port 80 (Cannot assign requested
address)
error: could not bind to 190.0.2.2 port 80 (Cannot assign requested
address)
```

These messages appear because IRIS FailSafe starts up the Netscape Communications Server (*ns_httpd*) and the Netscape Commerce Server (*ns_commerce*) before it configures the network interfaces up. They are harmless and can be ignored.

Netscape Server Not Responding

If a Netscape server is not responding after the network and the Netscape server have been installed and configured, make sure that the configured addresses are accessible. Follow these steps:

1. If you have multiple Netscape servers, verify that the file */etc/config/ns_httpd.options* file exists (it is created by IRIS FailSafe when it starts Netscape servers):

   ```
   # ls /etc/config/ns_httpd.options
   ```

2. Enable and start the Netscape Communications server if used:

   ```
   # chkconfig ns_httpd on
   # /etc/init.d/ns_httpd start
   ```

3. Enable and start the Netscape Commerce server if used:

   ```
   # chkconfig ns_commerce on
   # /etc/init.d/ns_commerce start
   ```

4. Run a Web browser, such as Netscape, and try to access some Web pages exported by the server.

Netscape Daemons Not Responding to Monitoring

These messages are written to */var/adm/SYSLOG* when the IRIS FailSafe Web monitoring script detects that the Netscape *httpd* daemons are no longer responding:

```
Nov 29 11:25:36 6D:xfs-ha2 syslog[775]: /usr/etc/ha_exec: command
/usr/etc/http_ping failed error=1. retrying
Nov 29 11:25:36 5B:xfs-ha2 root: Failed to ping local webserver [port:
80]
Nov 29 11:25:36 6D:xfs-ha2 ha_appmon[241]: webserver_xfs-ha1 local
monitoring failed: status = 3
Nov 29 11:25:36 6D:xfs-ha2 ha_nc[235]: Received LOCMONFAIL
```

Notice that the first line reports that */usr/etc/http_ping* was executed by */usr/etc/ha_exec* and failed. Try to determine why the */usr/etc/http_ping* command is failing. Possible reasons are:

- The Netscape server is not configured correctly for a high availability IP address.
- The Netscape server has not been started. Check the webserver entries in */var/ha/ha.conf*.
- Check the interface-pair blocks of */var/ha/ha.conf* to verify that a high availability IP address has been configured on the interface.

After you have fixed the problem, try executing the command that failed, */usr/etc/http_ping* in this case, from the command line to verify that the problem has been solved.

Failover Script Failures

IRIS FailSafe uses application-specific failover scripts. If a script fails, the error is logged to */var/adm/SYSLOG*. A sample entry might look like the following:

```
Nov 29 11:07:32 5B:xfs-ha1 root: ERROR: /sbin/xlv_mgr -c "change
nodename xfs-ha2 shared_vol"
Nov 29 11:07:32 6D:xfs-ha1 ha_appmon[238]: takeback script exited with
status 3
Nov 29 11:07:32 6D:xfs-ha1 ha_nc[232]: process appmon died with status
1
```

Normally, the other node in the cluster restarts this node and takes over its services. To diagnose what caused the original script failure, follow these steps:

1. Shut the cluster down by following the procedure in the section "Shutting Down IRIS FailSafe" in Chapter 6.
2. Re-enter the command with the same command options that failed. In the example above, the command is the call to */sbin/xlv_mgr*.
3. Rerun the appropriate script from */var/ha/actions* (in this case, */var/ha/actions/takeback*). For example:

   # **/var/ha/actions/takeback '/usr/etc/ha_cfgchksum'**

4. Make the appropriate fixes, which are most likely to be fixing errors in configuration, and bring the cluster back up.

ha_admin Times Out

The *ha_admin* command times out when the node is transitioning from one state to another. Retry the command after a few minutes.

Error Message from ha_statd

When *ha_cfgverify* is run automatically as part of the startup of IRIS FailSafe, it can generate this message:

```
ha_statd: Error sending message to rpc.statd
ha_statd: rpc.statd is a back revision
```

If you see this message, follow these steps:

1. Check to see what options the network status monitor daemon, */usr/etc/rpc.statd*, was started with by entering this command:

   ```
   # ps -ef | grep rpc.statd
       root   211     1  0   May 31 ?       0:00 /usr/etc/rpc.statd -h
       root  4215  4213  2 14:48:57 ttyq3   0:00 grep rpc.statd
   ```

   Look for the **-h** option (shown in this output). The error message shown above appears if the **-h** option wasn't used.

2. If the **-h** option wasn't used, add `-h` to the first line of the file */etc/config/statd.options* as described in the section "Configuring NFS Filesystems" in Chapter 3.
3. If you edited */etc/config/statd.options*, restart *rpc.statd* by entering this command:

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

False Failovers

A false failover occurs when there is a failover for no apparent reason. IRIS FailSafe monitors the highly available services, the other node in the cluster, the networks, and the connections between the nodes. If a service, a network, or the other node doesn't respond to the monitoring within a timeout period and this monitoring fails a specified number of times (the retry value), a failure is declared and a failover is initiated.

Preventing false failovers is done by tuning the values of the timeout and retry values in the configuration file */var/ha/ha.conf* so that they provide timely detection of true failures, but do not falsely detect failure because of node or network loading. See the section "Preventing False Failovers" in Chapter 4 for more information.

Errors Logged to /var/adm/SYSLOG

This section lists errors that might be logged to */var/adm/SYSLOG*.

`wd95_5:  WD95 saw SCSI reset`

This message is benign. A CHALLENGE node resets the SCSI bus in the process of starting up. For a dual-hosted system like IRIS FailSafe, the other node on the shared SCSI bus also sees the resets.

`checksum mismatch error`

Either the configuration files on the two nodes do not match or the configuration file has been changed after the IRIS FailSafe daemons were started.

Make sure that the */var/ha/ha.conf* files are the same on both nodes and then reboot both nodes in the cluster.

```
read_conf error
error return from read_config
nc_readconfig: Bad config file ...
```

The IRIS FailSafe daemons could not start up because the */var/ha/ha.conf* file on the node is invalid. In some cases, the error message also indicates the missing or invalid parameters. Run *ha_cfgverify* and fix any problems identified.

```
Kill failed (%d) after remote monitor failed or no heartbeat
```

The local node tried to take over the other node's services but failed because it could not reset the other node. The local node went into the standby state.

Check the serial connection between the nodes. If used, check the remote power control unit. After the problem has been fixed, used the command *ha_admin -rf* as described in the section "Moving a Node From Standby State to Normal or Degraded State" in Chapter 6.

```
assumed power failed
```

This node detected that the heartbeat and the failsafe mechanism have both failed on the remote node and assumed that the remote node experienced a power failure. This node then took over the other node's services.

```
Monitoring of reset-tty on %s failed
```

The IRIS FailSafe system could not communicate with the either the system controller port (CHALLENGE L) or the remote power control unit (CHALLENGE S). Although the system continues to function, this situation prevents a node from taking over from another node if another failure occurs.

Check the physical connections. Test the serial connection using the procedure in the section "Netscape Daemons Not Responding to Monitoring" in this chapter.

```
mail script failed
```

The application monitor detected a change in the state of the cluster, tried to send mail to the recipient specified in the */var/ha/ha.conf* file, and failed.

Verify the mail configuration on your nodes.

```
xlv_plexd[31]: DIOCXLVPLEXCOPY on xxx (dev 192.4) failed: No such device
or address
```

The plex revive operation was interrupted because the underlying device went away. This situation is usually because the shared volume has been given away to the other node in the cluster.

This message is benign.

```
New state: NC_ERROR
```

The IRIS FailSafe software has detected an internal inconsistency and has suspended operation. The nodes of the cluster are still running.

Verify the software and hardware configuration; reboot this node. Report this problem to Silicon Graphics Technical Support.

```
lost_heartbeat: heartbeat_<nodenumber> failed
Retrying heartbeat monitor over <public interface>
```

The cluster is in normal state, but a failure was detected in the private network.

Repair the private network. Follow the instructions in the section “Moving Heartbeat Messages to the Private Network” in Chapter 6 to switch the heartbeat back to the private network.

```
connect to opsnc failed
```

In a mixed OPS/IRIS FailSafe configuration, the *ha_killd* daemon could not connect to the OPS node controller on the Indy workstation that is running the IRISconsole software.

Verify that the cables from the Indy workstation running the IRISconsole are properly attached to the nodes in the IRIS FailSafe cluster, that the Indy workstation is running, and that the *opsnc* daemon is running on the Indy workstation.

`open_tty failed -- cannot monitor node`

This message applies to clusters running both Oracle Parallel Server (OPS) and IRIS FailSafe. If you see this message and the cluster is not running OPS, the problem is that the configuration parameter reset-host has been set. Remove reset-host to solve the problem.

If the cluster is running OPS, the IRIS FailSafe system cannot open the serial connection to the remote power control unit or the system controller port of the other node in the cluster.

Verify that the serial connection is hooked up and that the remote power control unit is powered on if used. Test the serial connection using the procedure in the section “Netscape Daemons Not Responding to Monitoring” in this chapter.

Glossary

active/backup cluster

An active / backup cluster is a cluster in which one node provides highly available services and the other node serves only as a backup.

active/backup configuration

In an active / backup configuration, only one node is providing a particular highly available service. The other node is only a backup for that highly available service.

cluster

A cluster is a pair of nodes with shared disk storage and connected for IRIS FailSafe use.

configuration file

A configuration file is a file created by an administrator of a cluster that specifies information about the hardware and software configuration of the IRIS FailSafe system. It is installed in */var/ha/ha.conf* on both nodes in a cluster. These files must be identical on the two nodes.

dual-active cluster

A dual-active cluster is a cluster in which both nodes provide highly available services during normal operation.

dual-active configuration

In a dual-active configuration, both nodes are providing a particular highly available service during normal operation.

fixed IP address

A fixed IP address is an IP address that is not failed over (in other vendors' high availability implementations this is sometimes called a *node IP address*). Each interface used by IRIS FailSafe must have exactly one fixed IP address. Fixed IP addresses are configured in the file */etc/config/netif.options*. Usually, the hostname of a node is an IP name for one of the fixed IP addresses on the node. Typically, IP addresses used for primary interfaces and private interfaces are fixed IP addresses.

heartbeat messages

Heartbeat messages are messages sent between the nodes that indicates a node is up and running.

high availability IP address

A high availability IP address is an IP address that is failed over by IRIS FailSafe (in other vendors' high availability implementations this is sometimes called a *service IP address*). It is configured by IRIS FailSafe and used by highly available services.

highly available applications

Highly available applications are client/server applications that have been configured to run in an IRIS FailSafe cluster. If the IRIS FailSafe system detects a failure on the node on which a highly available application is running, all highly available applications and resources are shut down on the failed node and restarted on the other node in the cluster.

highly available resources

Highly available resources are the I/O devices (interfaces and disks) that have been configured as highly-available services on the nodes in a cluster.

highly available services

Highly available services are interfaces, disks, and applications that have been configured to fail over to the other node in a cluster under the direction of IRIS FailSafe software. Highly available services can be divided into two categories, highly available resources and highly available applications.

interface

An interface is a hardware connection on a node to a network. Typical interface names are ec0 and fxp0 for Ethernet networks and xpi0 for FDDI networks.

IP address

An IP address is a network address specified as a name (a string) or in internet address notation (*X.X.X.X*). The use of internet address notation is required in some cases and strongly recommended in other cases because the name lookup time (see the resolver(4) reference page) can be excessive and cause IRIS FailSafe to assume a failure. You can manually configure an IP address to an interface with this command:

`/usr/etc/ifconfig` *if_name* **`inet`** *ip_address*

if_name is the interface name and *ip_address* is the IP address (name or internet notation). Configuring an IP address to an interface is normally done by the startup script */etc/init.d/network*.

IP alias

An IP alias is an IP address that is an additional network address for an interface. It is also known as a high availability IP address. When there is a failure on a node, IP addresses move from their primary interface to their secondary interface (on the other node). IP aliases are listed in */etc/hosts*. Configuring an IP alias to an interface is done by IRIS FailSafe, according to specifications in the IRIS FailSafe configuration file.

IP name

An IP name is an IP address in name (string) form. Specifying an IP name for a network address in internet address notation (*X.X.X.X*) is done in */etc/hosts*.

IRIS FailSafe system

An IRIS FailSafe system is a cluster running IRIS FailSafe software.

MAC address

A MAC address is a hardware-level Ethernet address. In most situations, MAC addresses are failed over automatically using the IP *re-arp* protocol. On Ethernet networks, for the few situations where the re-arp protocol isn't supported, IRIS FailSafe provides an alternate mechanism for MAC address failover (this mechanism, called *re-mac*'ing, isn't available on FDDI networks). See the section "Network Interfaces and IP Addresses" in Chapter 1 for more information.

node

A node is a CHALLENGE Server.

private interface

A private interface is a network interface that is connected to the private network between nodes.

private network

A private network is a network connection between the nodes in a cluster that is not advertised to other networks.

public network

A public network is a network connection between the nodes in a cluster and the clients who use the highly available services provided by the cluster.

re-mac

Re-mac is a mechanism used to change the physical address of a network interface. The change is not persistent across reboots.

serial connection

A serial connection is the serial line connection between the nodes in a cluster that provides a mechanism for one node to reboot the other node immediately.

Tell Us About This Manual

As a user of Silicon Graphics products, you can help us to better understand your needs and to improve the quality of our documentation.

Any information that you provide will be useful. Here is a list of suggested topics:

- General impression of the document
- Omission of material that you expected to find
- Technical errors
- Relevance of the material to the job you had to do
- Quality of the printing and binding

Please send the title and part number of the document with your comments. The part number for this document is 007-3109-002.

Thank you!

Three Ways to Reach Us

- To send your comments by **electronic mail**, use either of these addresses:
  - On the Internet: techpubs@sgi.com
  - For UUCP mail (through any backbone site): *[your_site]*!sgi!techpubs
- To **fax** your comments (or annotated copies of manual pages), use this fax number: 415-965-0964
- To send your comments by **traditional mail**, use this address:

  Technical Publications
  Silicon Graphics, Inc.
  2011 North Shoreline Boulevard, M/S 535
  Mountain View, California 94043-1389