CXFS™ Administration Guide for SGI® InfiniteStorage

007–4016–023

CONTRIBUTORS

Written by Lori Johnson

Illustrated by Chrystie Danzer

Production by Karen Jacobson

Engineering contributions to the book by Rich Altmaier, Neil Bannister, François Barbou des Places, Ken Beck, Felix Blyakher, Laurie Costello, Mark Cruciani, Rupak Das, Dave Ellis, Brian Gaffey, Philippe Gregoire, Dean Jansa, Erik Jacobson, John Keller, Dennis Kender, Chris Kirby, Ted Kline, Dan Knappe, Kent Koeninger, Linda Lait, Bob LaPreze, Steve Lord, Aaron Mantel, Troy McCorkell, LaNet Merrill, Terry Merth, Nate Pearlstein, Bryce Petty, Alain Renaud, John Relph, Elaine Robinson, Dean Roehrich, Eric Sandeen, Wesley Smith, Kerm Steffenhagen, Paddy Sreenivasan, Andy Tran, Rebecca Underwood, Connie Woodward, Michelle Webster, Geoffrey Wehrman

New Features in This Guide

This guide contains the following:

- "Discovering the WWNs" on page 335
- Support for system reset for SGI Altix systems that use an integrated L2, such as a NUMAlink 4 R-brick, or SGI Altix 3000 Bx2 systems. Configuring a node of this type requires use of `cmgr`. See "Define a Node with `cmgr`" on page 224.
- Support for SGI ProPack for Linux 4 as a client-only node. See: "ProPack 4 Client-Only Installation Overview" on page 91
- Support for the `cxfsdump -secure` option for secure remote connections. In cluster mode (the default), `cxfsdump` requires `rsh/ssh` and `rcp/scp` access across all nodes in the cluster. See "Gather Cluster Configuration with `cxfsdump`" on page 389.

Record of Revision

Version	Description
001	September 1999 Supports the CXFS 1.1 product in the IRIX 6.5.6f release.
002	October 1999 Supports the CXFS 1.1 product in the IRIX 6.5.6f release.
003	December 1999 Supports the CXFS product in the IRIX 6.5.7f release.
004	March 2000 Supports the CXFS product in the IRIX 6.5.8f release.
005	June 2000 Supports the CXFS product in the IRIX 6.5.9f release.
006	September 2000 Supports the CXFS product in the IRIX 6.5.10f release.
007	January 2001 Supports the CXFS product in the IRIX 6.5.11f release.
008	March 2001 Supports the CXFS product in the IRIX 6.5.12f release.
009	June 2001 Supports the CXFS product in the IRIX 6.5.13f release.
011	September 2001 Supports the CXFS product in the IRIX 6.5.14f release. (Note, there was no 010 version due to an internal numbering mechanism.)
012	December 2001 Supports the CXFS Version 2 product in IRIX 6.5.15f.
013	March 2002 Supports the CXFS Version 2 product in IRIX 6.5.16f.

014	June 2002 Supports the CXFS Version 2 product in IRIX 6.5.17f.
015	September 2002 Supports the CXFS Version 2 product in IRIX 6.5.18f.
016	December 2002 Supports the CXFS Version 2 product in IRIX 6.5.19f.
017	March 2003 Supports the CXFS Version 2 product in IRIX 6.5.20f.
018	September 2003 Supports the CXFS 3.0 product in IRIX 6.5.22 and CXFS 3.0 for SGI Altix 3000 running SGI ProPack 2.3 for Linux.
019	December 2003 Supports the CXFS 3.1 product in IRIX 6.5.23 and CXFS 3.1 for SGI Altix 3000 running SGI ProPack 2.4 for Linux.
020	March 2004 Supports the CXFS 3.2 product in IRIX 6.5.24 and CXFS 3.2 for SGI Altix 3000 running SGI ProPack 3 for Linux.
021	November 2004 Supports the CXFS 3.2 product in IRIX 6.5.24 and CXFS 3.2 for SGI Altix 3000 running SGI ProPack 3 for Linux.
022	April 2005 Supports the CXFS 3.3 product
023	July 2005 Supports the CXFS 3.4 product

Contents

Figures

Tables

About This Guide

This publication documents CXFS 3.4 running on a storage area network (SAN). It supports CXFS 3.4. It assumes that you are already familiar with the XFS filesystem and you have access to the *XVM Volume Manager Administrator's Guide*.

You should read through this entire book, especially Chapter 17, "Troubleshooting" on page 381, before attempting to install and configure a CXFS cluster.

Related Publications

The following documents contain additional information:

- *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*
- *FailSafe Administrator's Guide for SGI InfiniteStorage*
- *SGI InfiniteStorage Cluster Manager for Linux Administrator's Guide*
- *XVM Volume Manager Administrator's Guide*
- Storage area network (SAN) documentation:
  - *EL Serial Port Server Installation Guide* (provided by Digi International)
  - *EL Serial Port Server Installation Guide Errata*
  - *FDDIXPress Administration Guide*
  - *SGI® InfiniteStorage TP9400 and SGI® InfiniteStorage TP9500 and TP9500S RAID User's Guide*
  - *SGI InfiniteStorage TP9300 and TP9300S RAID User's Guide*
  - *SGI Total Performance 9100 Storage System Owner's Guide*
  - *SGI TPSSM Administration Guide*
- IRIX documentation:
  - *IRIX 6.5 Installation Instructions*
  - *IRIX Admin: Disks and Filesystems*

- *IRIX Admin: Networking and Mail*
- *Personal System Administration Guide*
- *Performance Co-Pilot for IRIX Advanced User's and Administrator's Guide*
- *Performance Co-Pilot Programmer's Guide*
- *Trusted IRIX Read Me First Notice*
- *Trusted IRIX/CMW Security Features User's Guide*

• SGI ProPack for Linux and SGI Altix documentation:
  - *NIS Administrator's Guide*
  - *Personal System Administration Guide*
  - *SGI ProPack for Linux Start Here*
  - *SGI Altix 3000 User's Guide*
  - *SGI Altix 350 System User's Guide*
  - *Performance Co-Pilot for IA-64 Linux User's and Administrator's Guide*
  - *SGI L1 and L2 Controller Software User's Guide*

The following man pages are provided with CXFS:

- `build_cmgr_script`
- `cbeutil`
- `cdbBackup`
- `cdbRestore`
- `cdbutil`
- `cmgr`
- `cmond`
- `cms_failconf`
- `crsd`
- `cxfsdump`

- `fs2d`
- `hafence`

Obtaining Publications

You can obtain SGI documentation as follows:

- See the SGI Technical Publications Library at http://docs.sgi.com. Various formats are available. This library contains the most recent and most comprehensive set of online books, release notes, man pages, and other information.
- If it is installed on your IRIX SGI system, you can use InfoSearch, an online tool that provides a more limited set of online books, release notes, and man pages. On an IRIX system, enter `infosearch` at a command line or select **Help > InfoSearch** from the Toolchest.
- You can view the release notes as follows:
- On IRIX systems, use either `grelnotes` or `relnotes`
- On SGI for ProPack Linux systems, see `linux-64/README_CXFS_LINUX64_3.4.0.txt` on the *CXFS 3.4 Altix Server/Client and XVM Plexing for SGI ProPack 2.4* CD
- You can view man pages by typing `man` *title* at a command line.

Conventions

This guide uses the following terminology abbreviations:

- *Solaris* to Solaris 8 and Solaris 9
- *Windows* to refer to Microsoft Windows 2000, Microsoft Windows 2003, and Microsoft Windows XP
- *Linux* used alone refers to the SGI ProPack for Linux operating system running on SGI hardware

The following conventions are used throughout this document:

Convention	**Meaning**
`command`	This fixed-space font denotes literal items such as commands, files, routines, path names, signals, messages, and programming language structures.
variable	Italic typeface denotes variable entries and words or concepts being defined.
`user input`	This bold, fixed-space font denotes literal items that the user enters in interactive sessions. (Output is shown in nonbold, fixed-space font.)
[ ]	Brackets enclose optional portions of a command or directive line.
GUI element	This bold font denotes the names of graphical user interface (GUI) elements, such as windows, screens, dialog boxes, menus, toolbars, icons, buttons, boxes, and fields.

This guide uses *Windows* to refer to both Microsoft Windows 2000 and Microsoft Windows XP nodes when the information applies equally to both. Information that applies to only one of these types of nodes is identified.

Reader Comments

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

You can contact SGI in any of the following ways:

- Send e-mail to the following address:

  techpubs@sgi.com

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

  http://docs.sgi.com

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

  Technical Publications
  SGI
  1500 Crittenden Lane, M/S 535
  Mountain View, California 94043–1351

SGI values your comments and will respond to them promptly.

Chapter 1

Introduction to CXFS

Note: You should read through this entire book, especially Chapter 17, "Troubleshooting" on page 381, before attempting to install and configure a CXFS cluster.

This chapter discusses the following:

- "What is CXFS?"
- "Comparison of XFS and CXFS" on page 2
- "Comparison of Network and CXFS Filesystems" on page 6
- "Cluster Environment" on page 9
- "Hardware and Software Support" on page 37
- "Overview of FailSafe Coexecution" on page 39
- "CXFS Tools Overview" on page 40
- "Guaranteed-Rate I/O (GRIO) Version 2 and CXFS" on page 42
- "XMV Failover" on page 45
- "Overview of the Installation and Configuration Steps" on page 47

Linux used alone refers to the SGI ProPack for Linux operating system running on SGI hardware. For information about Linux systems running on third-party hardware, see *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

Note: In this release, ProPack 4 nodes are always client-only nodes.

What is CXFS?

CXFS is clustered XFS, a clustered filesystem for high-performance computing environments.

CXFS allows groups of computers to coherently share XFS filesystems among multiple hosts and storage devices while maintaining high performance. CXFS runs on storage area network (SAN) disks, such as Fibre Channel. A SAN is a high-speed, scalable network of servers and storage devices that provides storage resource consolidation, enhanced data access/availability, and centralized storage management. CXFS filesystems are mounted across the cluster by CXFS management software. All files in the filesystem are available to all nodes that mount the filesystem.

Comparison of XFS and CXFS

CXFS uses the same filesystem structure as XFS. A CXFS filesystem is initially created using the same `mkfs` command used to create standard XFS filesystems.

The primary difference between XFS and CXFS filesystems is the way in which filesystems are mounted and managed:

- In XFS:
  - Filesystems are mounted with the `mount` command directly by the system during boot via an entry in `/etc/fstab` or by the IRIX Filesystem Manager.
  - A filesystem resides on only one host.
  - The `/etc/fstab` file contains static information about filesystems. For more information, see the `fstab` man page.
- In CXFS:
  - Filesystems are mounted using the CXFS Manager graphical user interface (GUI) or the `cmgr` command.
  - A filesystem is accessible to those hosts (nodes) in the cluster that are defined to mount it. CXFS filesystems are mounted across the cluster by CXFS management software. All files in the filesystem are visible to those hosts that are defined to mount the filesystem.
  - One node coordinates the updating of *metadata* (information that describes a file, such as the file's name, size, location, and permissions) on behalf of all nodes in a cluster; this is known as the *metadata server*.

    There is one *active metadata server* per CXFS filesystem; there can be multiple active metadata servers in a cluster, one for each CXFS filesystem.

– The filesystem information is stored in the *cluster database* (CDB), which contains persistent static configuration information about the filesystems, nodes, and cluster. The CXFS cluster daemons manage the distribution of multiple synchronized copies of the cluster database across the *CXFS administration nodes* in the pool. The administrator can view the database and modify it using the GUI or the `cmgr` command.

  The GUI shows the static and dynamic state of the cluster. For example, suppose the database contains the static information that a filesystem is enabled for mount; the GUI will display the dynamic information showing one of the following:

  - A blue icon indicating that the filesystem is mounted (the static and dynamic states match).
  - A grey icon indicating that the filesystem is configured to be mounted but the procedure cannot complete because CXFS services have not been started (the static and dynamic states do not match, but this is expected under the current circumstances). See "CXFS Services" on page 25.
  - An error (red) icon indicating that the filesystem is supposed to be mounted (CXFS services have been started), but it is not (the static and dynamic states do not match, and there is a problem).

  The following commands can also be used to view the cluster state:

  - `cmgr` and `cxfs-config` show the static cluster state. These commands are available on nodes used for cluster administration.
  - `clconf_info` shows both the static and dynamic cluster states. These commands are available on nodes used for cluster administration.
  - `cxfs_info` provides status information. This command is available on nodes that are CXFS clients but are not used for administration.

– Information is **not** stored in the `/etc/fstab` file. (However, the CXFS filesystems do show up in the `/etc/mtab` file.) For CXFS, information is instead stored in the cluster database.

Supported XFS Features

XFS features that are also present in CXFS include the following:

- Reliability and fast (subsecond) recovery of a log-based filesystem.
- 64-bit scalability to 9 million terabytes (9 exabytes) per file.
- Speed: high *bandwidth* (megabytes per second), high *transaction rates* (I/O per second), and fast metadata operations.
- Dynamically allocated metadata space.
- Quotas. You can administer quotas from any administration node in the cluster just as if this were a regular XFS filesystem.
- Filesystem reorganizer (defragmenter), which must be run from the CXFS metadata server for a given filesystem. See the `fsr_xfs` man page.
- Restriction of access to files using file permissions and access control lists (ACLs). You can also use logical unit (lun) masking or physical cabling to deny access from a specific host to a specific set of disks in the SAN.
- Real-time volumes. CXFS can write to real-time files in real-time volumes on IRIX nodes. For more information about real-time volumes, see *XVM Volume Manager Administrator's Guide*.

CXFS preserves these underlying XFS features while distributing the I/O directly between the disks and the hosts. The efficient XFS I/O path uses asynchronous buffering techniques to avoid unnecessary physical I/O by delaying writes as long as possible. This allows the filesystem to allocate the data space efficiently and often contiguously. The data tends to be allocated in large contiguous chunks, which yields sustained high bandwidths.

The XFS directory structure is based on B-trees, which allow XFS to maintain good response times, even as the number of files in a directory grows to tens or hundreds of thousands of files.

When to Use CXFS

You should use CXFS when you have multiple nodes running applications that require high-bandwidth access to common filesystems.

CXFS performs best under the following conditions:

- Data I/O operations are greater than 16 KB
- Large files are being used (a lot of activity on small files will result in slower performance)
- Read/write conditions are one of the following:
  - All processes that perform reads/writes for a given file reside on the same node.
  - The same file is read by processes on multiple nodes using buffered I/O, but there are no processes writing to the file.
  - The same file is read and written by processes on more than one node using direct-access I/O.

For most filesystem loads, the scenarios above represent the bulk of the file accesses. Thus, CXFS delivers fast local file performance. CXFS is also useful when the amount of data I/O is larger than the amount of metadata I/O. CXFS is faster than NFS because the data does not go through the network.

Performance Considerations

CXFS may not give optimal performance under the following circumstances, and extra consideration should be given to using CXFS in these cases:

- When you want to access files only on the local host.
- When distributed applications write to shared files that are memory mapped.
- When exporting a CXFS filesystem via NFS, be aware that performance will be much better when the export is performed from an active CXFS metadata server than when it is performed from a CXFS client. (Exporting from a backup metadata server is not supported. In order to support relocation and recovery, a backup server cannot run any applications that will use the filesystem. For more information, see "Node Functions" on page 11.)

- When access would be as slow with CXFS as with network filesystems, such as with the following:
  - Small files
  - Low bandwidth
  - Lots of metadata transfer

  Metadata operations can take longer to complete through CXFS than on local filesystems. Metadata transaction examples include the following:
  - Opening and closing a file
  - Changing file size (usually extending a file)
  - Creating and deleting files
  - Searching a directory

  In addition, multiple processes on multiple hosts that are reading and writing the same file using buffered I/O can be slower with CXFS than when using a local filesystem. This performance difference comes from maintaining coherency among the distributed file buffers; a write into a shared, buffered file will invalidate data (pertaining to that file) that is buffered in other hosts.

Comparison of Network and CXFS Filesystems

Network filesystems and CXFS filesystems perform many of the same functions, but with important performance and functional differences noted here.

Network Filesystems

Accessing remote files over local area networks (LANs) can be significantly slower than accessing local files. The network hardware and software introduces delays that tend to significantly lower the transaction rates and the bandwidth. These delays are difficult to avoid in the client-server architecture of LAN-based network filesystems. The delays stem from the limits of the LAN bandwidth and latency and the shared path through the data server.

LAN bandwidths force an upper limit for the speed of most existing shared filesystems. This can be one to several orders of magnitude slower than the bandwidth possible across multiple disk channels to local or shared disks. The layers of network protocols and server software also tend to limit the bandwidth rates.

A shared fileserver can be a bottleneck for performance when multiple clients wait their turns for data, which must pass through the centralized fileserver. For example, NFS and Samba servers read data from disks attached to the server, copy the data into UDP/IP or TCP/IP packets, and then send it over a LAN to a client host. When many clients access the server simultaneously, the server's responsiveness degrades.

Note: You should not use multiple Samba servers to export the same CXFS filesystem. For more information, see "Samba" on page 310.

CXFS Filesystems

CXFS is a clustered XFS filesystem that allows for logical file sharing, as with network filesystems, but with significant performance and functionality advantages. CXFS runs on top of a storage area network (SAN), where each host in the cluster has direct high-speed data channels to a shared set of disks.

Features

CXFS has the following unique features:

- A *peer-to-disk* model for the data access. The shared files are treated as local files by all of the hosts in the cluster. Each host can read and write the disks at near-local disk speeds; the data passes directly from the disks to the host requesting the I/O, without passing through a data server or over a local area network (LAN). For the data path, each host is a peer on the SAN; each can have equally fast direct data paths to the shared disks.

  Therefore, adding disk channels and storage to the SAN can scale the bandwidth. On large systems, the bandwidth can scale to gigabytes and even tens of gigabytes per second. Compare this with a network filesystem with the data typically flowing over a 1- to 100-MB-per-second LAN.

  This peer-to-disk data path also removes the file-server data-path bottleneck found in most LAN-based shared filesystems.

- Each host can buffer the shared disk much as it would for locally attached disks. CXFS maintains the coherency of these distributed buffers, preserving the advanced buffering techniques of the XFS filesystem.

- A flat, single-system view of the filesystem; it is identical from all hosts sharing the filesystem and is not dependent on any particular host. The pathname is a normal POSIX pathname; for example, `/u/`*username*`/`*directory*.

  Note: A Windows CXFS client uses the same pathname to the filesystem as other clients beneath a preconfigured drive letter.

  The path does not vary if the metadata server moves from one node to another, if the metadata server name is changed, or if a metadata server is added or replaced. This simplifies storage management for administrators and users. Multiple processes on one host and processes distributed across multiple hosts have the same view of the filesystem, with performance similar on each host.

  This differs from typical network filesystems, which tend to include the name of the fileserver in the pathname. This difference reflects the simplicity of the SAN architecture with its *direct-to-disk* I/O compared with the extra hierarchy of the LAN filesystem that goes through a named server to get to the disks.

- A full UNIX filesystem interface, including POSIX, System V, and BSD interfaces. This includes filesystem semantics such as mandatory and advisory record locks. No special record-locking library is required.

Restrictions

CXFS has the following restrictions:

- Some filesystem semantics are not appropriate and not supported in shared filesystems. For example, the root filesystem is not an appropriate shared filesystem. Root filesystems belong to a particular host, with system files configured for each particular host's characteristics.
- All processes using a named pipe must be on the same node.
- Hierarchical storage management (HSM) applications must run on the metadata server.
- The inode monitor device (`imon`) is not supported on CXFS filesystems.

The following XFS features are not supported in CXFS:

- The original XFS guaranteed-rate I/O (GRIO) implementation, GRIO version 1. (GRIO version 2 is supported, see "Guaranteed-Rate I/O (GRIO) Version 2 and CXFS" on page 42).
- Swap to a file residing on a CXFS file system.

Cluster Environment

This section discusses the following:

- "Terminology"
- "Isolating Failed Nodes" on page 27
- "The Cluster Database and CXFS Clients" on page 34
- "Metadata Server Functions" on page 35
- "System View" on page 36
- "CXFS and Highly Available Services" on page 36

For details about CXFS daemons, communication paths, and the flow of metadata, see Appendix A, "CXFS Software Architecture" on page 445.

Terminology

This section defines the terminology necessary to understand CXFS. Also see the Glossary on page 521.

Cluster

A *cluster* is the set of systems (nodes) configured to work together as a single computing resource. A cluster is identified by a simple name and a cluster ID. A cluster running multiple operating systems is known as a *multiOS cluster*.

Only one cluster may be formed from a given pool of nodes.

Disks or logical units (LUNs) are assigned to a cluster by recording the name of the cluster on the disk (or LUN). Thus, if any disk is accessible (via a Fibre Channel connection) from nodes in different clusters, then those clusters must have unique names. When members of a cluster send messages to each other, they identify their cluster via the cluster ID. Cluster names and IDs must be unique.

Because of the above restrictions on cluster names and cluster IDs, and because cluster names and cluster IDs cannot be changed once the cluster is created (without deleting the cluster and recreating it), SGI advises that you choose unique names and cluster IDs for each of the clusters within your organization.

Node

A *node* is an operating system (OS) image, usually an individual computer. (This use of the term *node* does not have the same meaning as a node in an SGI Origin 3000 or SGI 2000 system.)

A given node can be a member of only one pool and therefore only one cluster.

Pool

The *pool* is the set of nodes from which a particular cluster may be formed. Only one cluster may be configured from a given pool, and it need not contain all of the available nodes. (Other pools may exist, but each is disjoint from the other. They share no node or cluster definitions.)

A pool is first formed when you connect to a given CXFS administration node (one that is installed with `cluster_admin`) and define that node in the cluster database using the CXFS GUI or `cmgr` command. You can then add other nodes to the pool by defining them while still connected to the first node. (If you were to connect to a different node and then define it, you would be creating a second pool).

Figure 1-1 shows the concepts of pool and cluster.

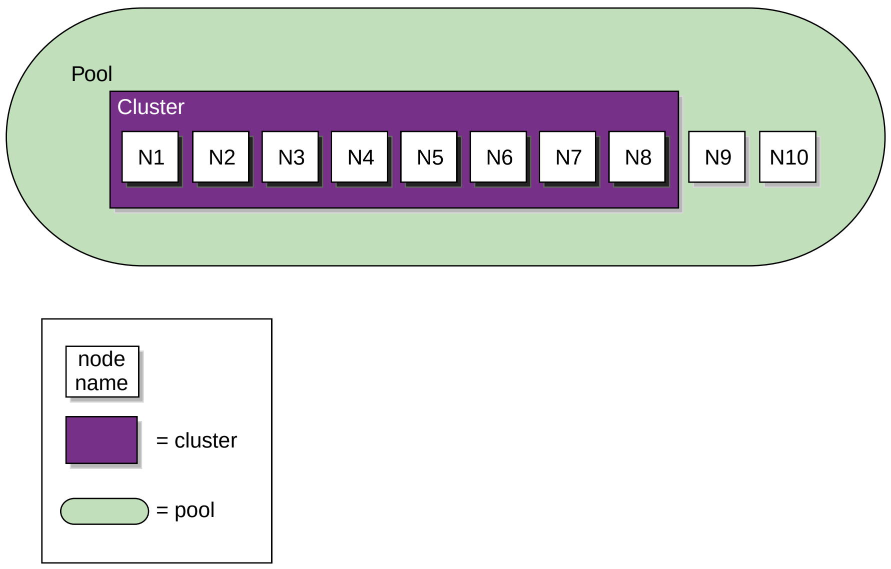


Figure 1-1 Pool and Cluster Concepts

Cluster Database

The *cluster database* contains configuration information about nodes, the cluster, logging information, and configuration parameters. The *cluster administration daemons* manage the distribution of the cluster database (CDB) across the CXFS administration nodes in the pool. See "Cluster Administration Daemons" on page 24.

The database consists of a collection of files; you can view and modify the contents of the database by using the following:

- CXFS Manager GUI connected to a CXFS administration node
- `cmgr`, `clconf_info`, and `cxfs-config` commands on a CXFS administration node
- `cxfs_info` command on a client-only nodes

Node Functions

A node can have one of the following functions:

- *CXFS metadata server-capable administration node* (IRIX or Linux).

This node is installed with the `cluster_admin` software product, which contains the full set of cluster administration daemons (`fs2d`, `crsd`, `cad`, and `cmond`) and the CXFS control daemon (`clconfd`). For more details about daemons, see "Cluster Administration Daemons" on page 24, "CXFS Control Daemon" on page 26, and Appendix A, "CXFS Software Architecture" on page 445.

This node type is capable of coordinating cluster activity and metadata. *Metadata* is information that describes a file, such as the file's name, size, location, and permissions. Metadata tends to be small, usually about 512 bytes per file in XFS. This differs from the *data*, which is the contents of the file. The data may be many megabytes or gigabytes in size.

For each CXFS filesystem, one node is responsible for updating that filesystem's metadata. This node is referred to as the *metadata server*. Only nodes defined as server-capable nodes are eligible to be metadata servers.

Multiple CXFS administration nodes can be defined as *potential metadata servers* for a given CXFS filesystem, but only one node per filesystem is chosen to be the *active metadata server*. All of the potential metadata servers for a given cluster must be either all IRIX or all Linux. There can be multiple active metadata servers in the cluster, one per CXFS filesystem.

Other nodes that mount a CXFS filesystem are referred to as *CXFS clients*. A CXFS administration node can function as either a metadata server or CXFS client, depending upon how it is configured and whether it is chosen to be the active metadata server.

Note: Do not confuse *metadata server* and *CXFS client* with the traditional data-path client/server model used by network filesystems. Only the metadata information passes through the metadata server via the private Ethernet network; the data is passed directly to and from disk on the CXFS client via the Fibre Channel connection.

You perform cluster administration tasks by using the `cmgr` command running on a CXFS administration node or by using the CXFS Manager GUI and connecting it to a CXFS administration node. For more details, see:

– Chapter 9, "Reference to GUI Tasks for CXFS" on page 149

– Chapter 10, "Reference to `cmgr` Tasks for CXFS" on page 217

There should be an odd number of server-capable administration nodes for quorum calculation purposes. If you have an even number of server-capable administration nodes, define a CXFS tiebreaker node (see "CXFS Tiebreaker" on page 23.

- *CXFS client administration node* (IRIX or Linux).

  This is a node that is installed with the `cluster_admin` software product but it cannot be a metadata server. This node type should only be used when necessary for coexecution with FailSafe.

- *CXFS client-only node* (any supported CXFS operating system).

  This node is one that has a minimal implementation of CXFS that runs a single daemon, the CXFS client daemon (`cxfs_client`). For more details, see Appendix A, "CXFS Software Architecture" on page 445.

  This node can safely mount CXFS filesystems but it cannot become a CXFS metadata server or perform cluster administration. Client-only nodes retrieve the information necessary for their tasks by communicating with an administration node. This node does not contain a copy of the cluster database.

  IRIX and Linux nodes are client-only nodes if they are installed with the `cxfs_client` software package and defined as client-only nodes. Nodes that are running supported operating systems other than IRIX or Linux are always configured as CXFS client-only nodes.

  For more information, see *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

Figure 1-2 shows nodes in a pool that are installed with `cluster_admin` and others that are installed with `cxfs_client`. Only those nodes with `cluster_admin` have the `fs2d` daemon and therefore a copy of the cluster database.

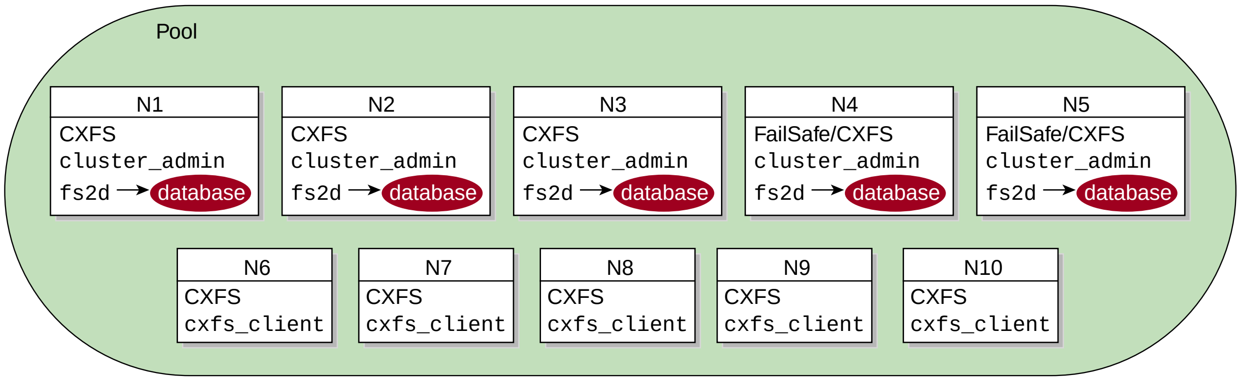


Figure 1-2 Installation Differences

A *standby node* is a server-capable administration node that is configured as a potential metadata server for a given filesystem, but does not currently run any applications that will use that filesystem. (The node can run applications that use other filesystems.)

Ideally, all administration nodes will run the same version of the operating system. However, as of IRIX 6.5.18f, SGI supports a policy for CXFS that permits a rolling annual upgrade; see "Rolling Upgrades" on page 106.

The following figures show different possibilities for metadata server and client configurations. The potential metadata servers are required to be CXFS administration nodes and must all run IRIX or all run Linux; the other nodes could be client-only nodes.

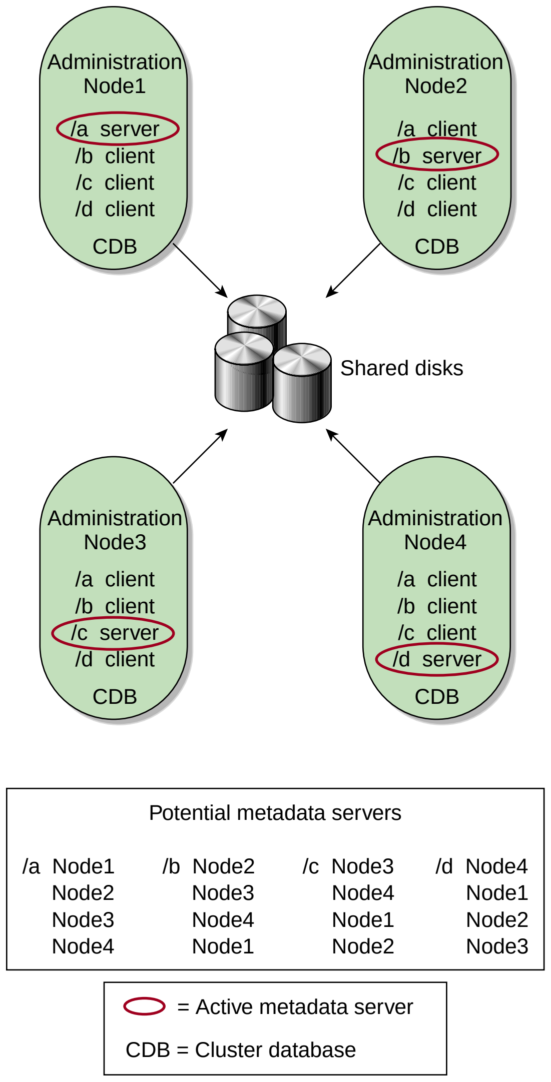
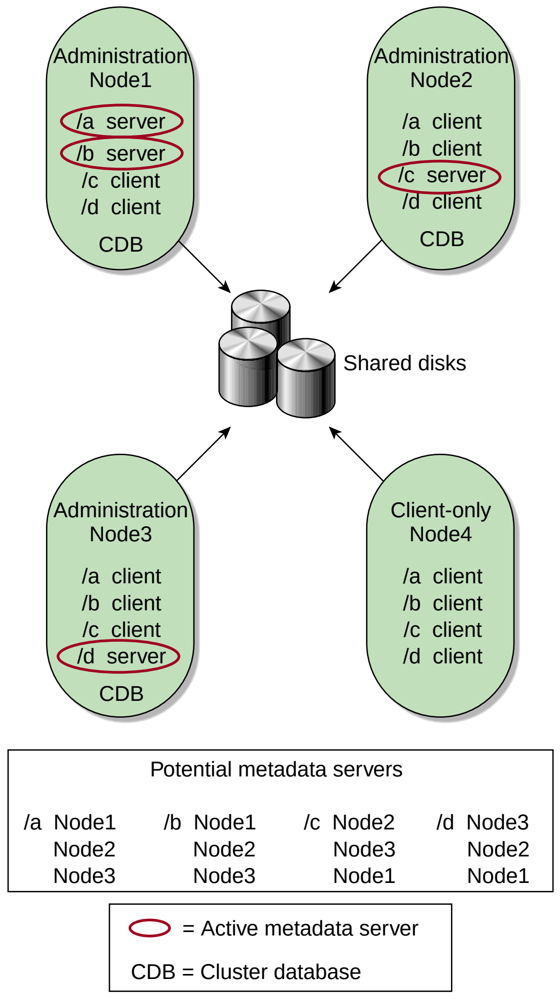


Figure 1-3 Evenly Distributed Metadata Servers



Potential metadata servers

/a Node1	/b Node1	/c Node2	/d Node3
Node2	Node2	Node3	Node2
Node3	Node3	Node1	Node1

⭕ = Active metadata server

CDB = Cluster database

Figure 1-4 Multiple Metadata Servers

In Figure 1-4, Node4 could be running any supported OS because it is a client-only node; it is not a potential metadata server.

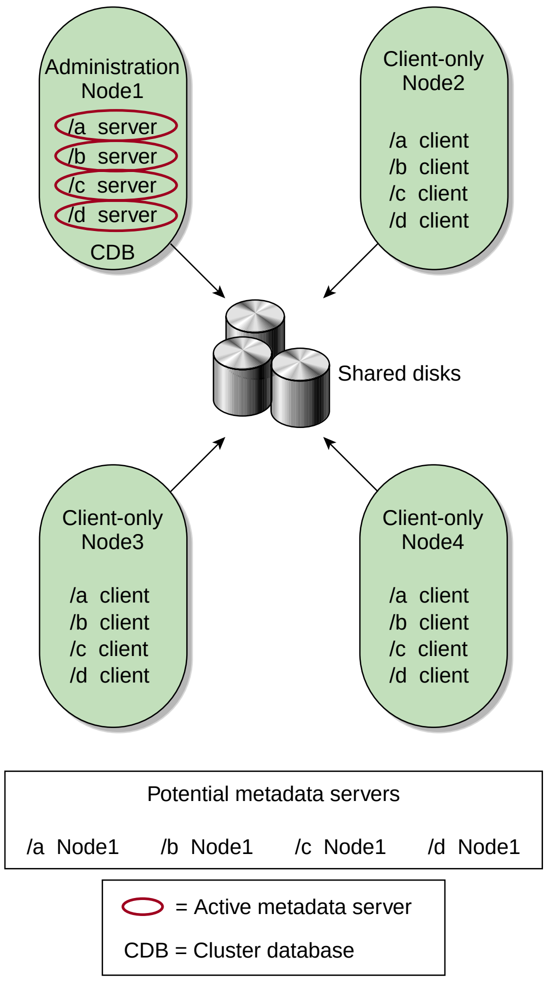


Figure 1-5 One Metadata Server

In Figure 1-5, Node2, Node3, and Node4 could be running any supported OS because they are client-only nodes; they are not potential metadata servers.

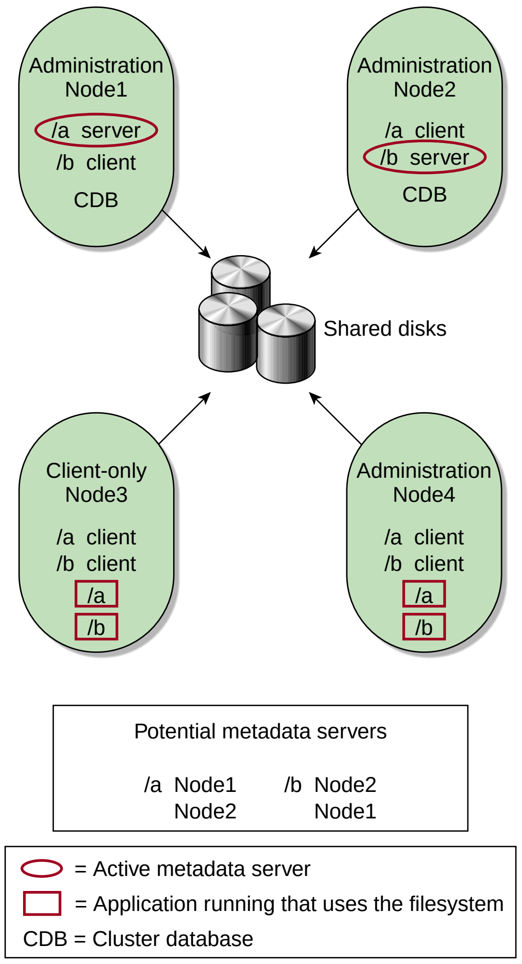


Figure 1-6 Standby Mode

Figure 1-6 shows a configuration in which Node1 and Node2 are potential metadata servers for filesystems `/a` and `/b`:

- Node1 is the active metadata server for `/a`
- Node2 is the active metadata server for `/b`

Because standby mode is used, neither Node1 nor Node2 runs applications that use `/a` or `/b`. The figure shows one client-only node, but there could be several.

Membership

The nodes in a cluster must act together to provide a service. To act in a coordinated fashion, each node must know about all the other nodes currently active and providing the service. The set of nodes that are currently working together to provide a service is called a *membership*:

- *Cluster database membership* (also known as `fs2d` *membership* or *user-space membership*) is the group of administration nodes that are accessible to each other. (client-only nodes are not eligible for cluster database membership.) The nodes that are part of the the cluster database membership work together to coordinate configuration changes to the cluster database.
- *CXFS kernel membership* is the group of CXFS nodes in the **cluster** that can actively share filesystems, as determined by the the CXFS kernel, which manages membership and heartbeating. The CXFS kernel membership may be a subset of the nodes defined in a cluster. All nodes in the cluster are eligible for CXFS kernel membership.

Heartbeat messages for each membership type are exchanged via a private network so that each node can verify each membership.

A cluster that is also running FailSafe has a *FailSafe membership*, which is the group of nodes that provide highly available (HA) resources for the cluster. For more information, see Appendix B, "Memberships and Quorums" on page 459, and the *FailSafe Administrator's Guide for SGI InfiniteStorage*.

Private Network

A *private network* is one that is **dedicated** to cluster communication and is accessible by administrators but not by users.

Note: A virtual local area network (VLAN) is not supported for a private network.

CXFS uses the private network for metadata traffic. The cluster software uses the private network to send the heartbeat/control messages necessary for the cluster configuration to function. Even small variations in heartbeat timing can cause problems. If there are delays in receiving heartbeat messages, the cluster software may determine that a node is not responding and therefore revoke its CXFS kernel membership; this causes it to either be reset or disconnected, depending upon the configuration.

Rebooting network equipment can cause the nodes in a cluster to lose communication and may result in the loss of CXFS kernel membership and/or cluster database membership ; the cluster will move into a degraded state or shut down if communication between nodes is lost. Using a private network limits the traffic on the network and therefore will help avoid unnecessary resets or disconnects. Also, a network with restricted access is safer than one with user access because the messaging protocol does not prevent *snooping* (illicit viewing) or *spoofing* (in which one machine on the network masquerades as another).

Therefore, because the performance and security characteristics of a public network could cause problems in the cluster and because heartbeat is very timing-dependent, **a private network is required**. The private network should be used for metadata traffic only.

The heartbeat and control network must be connected to all nodes, and all nodes must be configured to use the same subnet for that network.

Caution: If there are any network issues on the private network, fix them before trying to use CXFS. A stable private network is important for a stable CXFS cluster network.

For more information about network segments and partitioning, see Appendix B, "Memberships and Quorums" on page 459. For information about failover from the private network to another network, see information about the `add net network` command to `cmgr` in "Define a Cluster with `cmgr`" on page 246. (Although the primary network must be private, the backup network may be public.) For information about using IP filtering for the private network, see Appendix C, "IP Filtering Example for the CXFS Private Network" on page 479.

Relocation

Relocation is the process by which the metadata server moves from one node to another due to an administrative action; other services on the first node are not interrupted.

Note: Relocation is supported only to standby nodes. Relocation is disabled by default.

A *standby node* is a metadata server-capable administration node that is configured as a potential metadata server for a given filesystem, but does not currently run any applications that will use that filesystem. To use relocation, you must not run any applications on any of the potential metadata servers for a given filesystem; after the active metadata server has been chosen by the system, you can then run applications that use the filesystem on the active metadata server and client-only nodes.

To use relocation to a standby node, you must enable relocation on the active metadata server (relocation is disabled by default.) To enable relocation, reset the `cxfs_relocation_ok` parameter as follows:

- IRIX:
  - Enable:
    ```
    irix# systune cxfs_relocation_ok 1
    ```
  - Disable:
    ```
    irix# systune cxfs_relocation_ok 0
    ```
- Linux:
  - Enable at run time:
    ```
    [root@linux64 root]# sysctl -w fs.cxfs.cxfs_relocation_ok=1
    ```
  - Enable at reboot by adding the following line to `/etc/modules.conf`:
    ```
    options sgi-cxfs cxfs_relocation_ok=1
    ```
  - Disable:
    ```
    [root@linux64 root]# sysctl -w fs.cxfs.cxfs_relocation_ok=0
    ```
  - Disable at reboot by adding the following line to `/etc/modules.conf`:
    ```
    options sgi-cxfs cxfs_relocation_ok=0
    ```

CXFS kernel membership is not affected by relocation. However, users may experience a degradation in filesystem performance while the metadata server is relocating.

The following are examples of relocation triggers:

- The system administrator uses the GUI or `cmgr` to relocate the metadata server.

- The FailSafe `CXFS` resource relocates the IRIX CXFS metadata server. The SGI Cluster Manager for Linux CXFS plug-in relocates the Linux metadata server.
- The system administrator unmounts the CXFS filesystem on an IRIX metadata server. (Unmounting on a Linux metadata server does not trigger relocation; the Linux server will just return an `EBUSY` flag.)

Recovery

Recovery is the process by which the metadata server moves from one node to another due to an interruption in services on the first node.

Note: Recovery is supported only to standby nodes.

To use recovery to a standby node, you must not run any applications on any of the potential metadata servers for a given filesystem; after the active metadata server has been chosen by the system, you can then run applications that use the filesystem on the active metadata server and client-only nodes.

The following are examples of recovery triggers:

- A metadata server panic
- A metadata server locks up, causing heartbeat timeouts on metadata clients
- A metadata server loses connection to the heartbeat network

Figure 1-7 describes the difference between relocation and recovery for a metadata server. (Remember that there is one active metadata server per CXFS filesystem. There can be multiple active metadata servers within a cluster, one for each CXFS filesystem.)

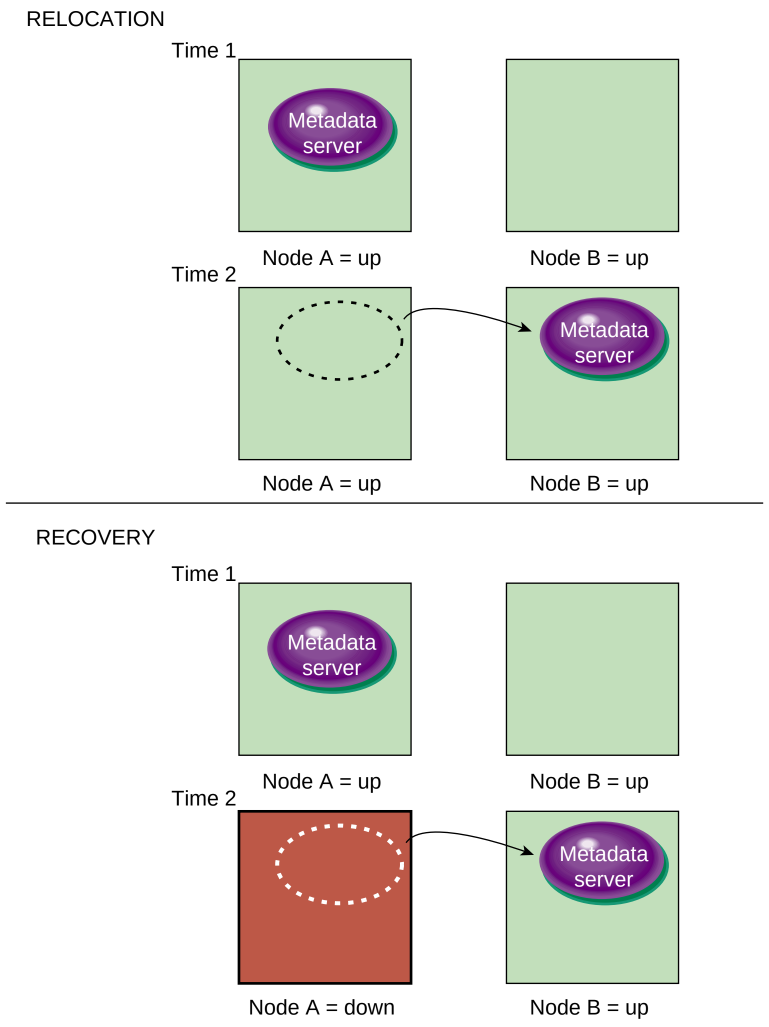


Figure 1-7 Relocation versus Recovery

CXFS Tiebreaker

The *CXFS tiebreaker node* is used in the process of computing the CXFS kernel membership for the cluster when exactly half the nodes in the cluster are up and can communicate with each other. There is no default CXFS tiebreaker.

A tiebreaker should be used in addition to *I/O fencing* or *system reset*; see "Isolating Failed Nodes" on page 27. SGI recommends making a client administration or client-only node the tiebreaker to avoid losing the cluster if the tiebreaker node fails.

Note: No matter what the cluster components are, SGI recommends a system reset configuration on potential metadata servers to protect data integrity and improve server reliability. I/O fencing (or system reset when available) must be used on client-only nodes. See "Use a Client-Only Tiebreaker" on page 116.

The CXFS tiebreaker differs from the FailSafe tiebreaker; see *FailSafe Administrator's Guide for SGI InfiniteStorage*.

Cluster Administration Daemons

The following set of daemons, which control various cluster infrastructure needs:

Daemon	Description
`fs2d`	Manages the cluster database (CDB) on the local administration node and keeps the copy on all administration nodes synchronized.
`cad`	Provides administration status services to the CXFS GUI.
`cmond`	Manages all other cluster administration daemons and the CXFS control daemon (`clconfd`). The `cmond` daemon starts the other daemons on the node and restarts them on failure.
`crsd`	Monitors the serial connection to other nodes. Has the ability to reset other nodes.

You can start and stop the cluster administration daemons with the following commands:

- IRIX:

  ```
  /etc/init.d/cluster {start|stop}
  ```

- Linux:

  ```
  /etc/init.d/cxfs_cluster {start|stop}
  ```

Note: You could also use the `restart` option to stop and start.

You can also use the following `chkconfig` commands to specify that the daemons will be restarted upon reboot:

- IRIX:

  ```
  chkconfig cluster on
  ```

- Linux:

  ```
  chkconfig cxfs_cluster on
  ```

For more information, see Appendix A, "CXFS Software Architecture" on page 445.

CXFS Services

The enabling/disabling of a node, which changes a flag in the cluster database. Starting or stopping CXFS services does not affect the daemons involved. The daemons that control CXFS services are as follows:

- `clconfd` on administration nodes, see "CXFS Control Daemon" on page 26.
- `cxfs_client` on client-only nodes, see "CXFS Client Daemon" on page 26.

To *start CXFS services* means to enable a node, which changes a flag in the cluster database by performing an administrative task using the CXFS graphical user interface (GUI) or the `cmgr` command:

- "Start CXFS Services with the GUI" on page 191
- "Start CXFS Services with `cmgr`" on page 254

To *stop CXFS services* means to disable a node, which changes a flag in the cluster database, by performing an administrative task using the GUI or `cmgr`:

- "Stop CXFS Services with the GUI" on page 191
- "Stop CXFS Services with `cmgr`" on page 254

To *shutdown CXFS services* means to withdraw a node from the CXFS kernel membership, either due to the fact that the node has failed somehow or by issuing an `admin cxfs_stop` command. The node remains enabled in the cluster database. See "Forced CXFS Shutdown: Revoke Membership of Local Node" on page 304.

CXFS Control Daemon

The `clconfd` daemon, which controls CXFS services on an administration node. It does the following:

- Obtains the cluster configuration from the `fs2d` daemon and manages the local CXFS administration node's CXFS kernel membership services and filesystems accordingly
- Obtains membership and filesystem status from the kernel
- Issues reset commands to the `crsd` daemon
- Issues I/O fencing commands to configured Fibre Channel switches

You can start/stop `clconfd` with the following command on an IRIX or Linux administration node:

```
/etc/init.d/cxfs {start/stop}
```

The `clconfd` daemon may still be running when CXFS services are disabled.

You can also use the following `chkconfig` command to specify that `clconfd` will be restarted upon reboot:

```
chkconfig cxfs on
```

For more information, see Appendix A, "CXFS Software Architecture" on page 445.

CXFS Client Daemon

The `cxfs_client` daemon, which controls CXFS services on a client-only node. It does the following:

- Obtains the cluster configuration from a remote `fs2d` daemon and manages the local client-only node's CXFS kernel membership services and filesystems accordingly.
- Obtains membership and filesystem status from the kernel.

You can start/stop `cxfs_client` with the following command on a client-only IRIX or Linux node:

```
/etc/init.d/cxfs_client {start|stop}
```

Note: The path to the `cxfs_client` command varies among the other platforms supported. See the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

You can also use the following `chkconfig` command to specify that `cxfs_client` will be restarted upon reboot:

- IRIX:

  ```
  chkconfig cxfs_cluster on
  ```

- Linux:

  ```
  chkconfig cxfs on
  ```

The `cxfs_client` daemon may still be running when CXFS services are disabled.

For more information, see Appendix A, "CXFS Software Architecture" on page 445.

Forced CXFS Shutdown

Withdraws a node from cluster membership, which disables filesystem and cluster volume access for the node. This is either due to the fact that the node has failed somehow or by issuing an `admin cxfs_stop` command. The node remains enabled in the cluster database. See "Forced CXFS Shutdown: Revoke Membership of Local Node" on page 304.

Isolating Failed Nodes

CXFS uses the following methods to isolate failed nodes:

- *I/O fencing*, which isolates a problem node from the SAN by disabling a node's Fibre Channel ports so that it cannot access I/O devices, and therefore cannot corrupt data in the shared CXFS filesystem. I/O fencing can be applied to any node in the cluster. When fencing is applied, the rest of the cluster can begin immediate recovery.

  I/O fencing is required to protect data for client-only nodes without system controllers. I/O fencing is also available for client-only nodes with system controllers; client-only nodes with system controllers can choose either I/O fencing, system reset, or I/O fencing with system reset. SGI recommends that potential metadata servers use system reset.

To support I/O fencing, platforms require a Fibre Channel switch; for supported switches, see the release notes. You must put switches used for I/O fencing on a network other than the primary CXFS private network so that problems on the CXFS private network can be dealt with by the fencing process and thereby avoid data corruption issues. The network to which the switch is connected must be accessible by all administration nodes in the cluster.

Note: I/O fencing differs from zoning. *Fencing* is a generic cluster term that means to erect a barrier between a host and shared cluster resources. *Zoning* is the ability to define logical subsets of the switch (zones), with the ability to include or exclude hosts and media from a given zone. A host can access only media that are included in its zone. Zoning is one possible implementation of fencing.

Zoning implementation is complex and does not have uniform availability across switches. Therefore, SGI chose to implement a simpler form of fencing: enabling/disabling a host's Fibre Channel ports.

- *System reset*, which performs a system reset via a serial line to the system controller. Reset is recommended for all potential metadata servers.
- *I/O fencing and system reset*, which disables access to the SAN from the problem node and then, if the node is successfully fenced, performs an asynchronous system reset if the node has a system reset capability; recovery begins without waiting for reset acknowledgment. If the fencing action fails, the reset is not performed; therefore, system reset alone is also recommended for all potential metadata servers. This method applies to nodes with system reset capability.
- *CXFS shutdown*, which stops CXFS kernel-based services on the local node in response to a loss of CXFS kernel membership. If the `shutdown` failure hierarchy setting is used, the surviving cluster delays the beginning of recovery to allow the node time to complete the shutdown. However, there is no notification that the shutdown has actually taken place.

⚠ **Caution:** Because there is no notification that a shutdown has occurred, if you have a cluster with an even number of server-capable nodes and no tiebreaker, you should not use the `shutdown` setting for any server-capable node in order to avoid multiple clusters being formed. See "Issues with the Shutdown Fail Action Setting" on page 29.

On client-only nodes without system controllers, data integrity protection requires I/O fencing.

On client-only nodes with system reset capability, you would want to use I/O fencing for data integrity protection when CXFS is just a part of what the node is doing and therefore losing access to CXFS is preferable to having the system rebooted. An example of this would be a large compute server that is also a CXFS client. However, I/O fencing cannot return a nonresponsive node to the cluster; this problem will require intervention from the system administrator. You would want to use system reset for I/O protection on a client-only node when CXFS is a primary activity and you want to get it back online fast; for example, a CXFS fileserver.

You can specify how these methods are implemented by defining the *failure action hierarchy*, the set of instructions that determines which method is used; see "Define a Node with the GUI" on page 172, and "Define a Node with `cmgr`" on page 224. The default for a CXFS administration node defined in the GUI is to perform a system reset and a CXFS shutdown (without a delay on behalf of the remaining cluster). The default for a client-only node defined in the GUI is to perform a CXFS shutdown.

The rest of this section provides more details about I/O fencing.

Issues with the Shutdown Fail Action Setting

The shutdown fail action setting tells the other nodes in the cluster to delay before resuming the cluster, in order to give the failed node time to complete a shutdown on itself. However, there is no guarantee that the shutdown was successfully performed. In the case of a cluster with an even number of server-capable administration nodes and no tiebreaker node, it is possible that using the shutdown setting could cause a split-brain scenario in which multiple clusters could be formed and data could therefore be corrupted.

For example, suppose the cluster has just two server-capable nodes `NodeA` and `NodeB`, and there is no tiebreaker. The nodes have the following fail actions:

```
NodeA           NodeB
-----           -----
fence           fence
reset           reset
shutdown        shutdown
```

If the CXFS private network between NodeA and NodeB fails, the following could occur:

1. Each node will try to fence the other. (That is, NodeA will try to fence NodeB, and NodeB will try to fence NodeA).
2. If the fence fails, each node will try to reset the other.
3. If the system reset fails, each assumes that the other will shut itself down. Each will wait for a few moments and will then try to maintain the cluster.
4. If the shutdown of NodeA is not successful, NodeA will try to maintain the cluster. If the shutdown of NodeB is not successful, NodeB will also try to maintain the cluster. This could result in two clusters that are unaware of each other (a split-brain situation) and data corruption will likely occur.

Suppose another configuration, in which neither node has shutdown set:

```
NodeA           NodeB
-----           -----
fence           fence
reset           reset
```

If the CXFS private network between NodeA and NodeB fails in this situation, each node would first try to fence the other and then try to reset the other, as before. However, if both of those actions fail, each would assume that the state of the other node is unknown. Therefore, neither node would try to maintain the cluster. The cluster will go down, but no data corruption will occur.

The split-brain problem may be avoided by using a tiebreaker node, by using an odd number of server-capable nodes, or by not using the shutdown setting on any server-capable node.

I/O Fencing

I/O fencing does the following:

- Preserves data integrity by preventing I/O from nodes that have been expelled from the cluster
- Speeds the recovery of the surviving cluster, which can continue immediately rather than waiting for an expelled node to reset under some circumstances

When a node joins the CXFS kernel membership, the worldwide port name (WWPN) of its host bus adapter (HBA) is stored in the cluster database. If there are problems with the node, the I/O fencing software sends a message via the `telnet` protocol to the appropriate switch and disables the port.

Caution: The `telnet` port must be kept free in order for I/O fencing to succeed.

Brocade switches running 4.*x.x.x* or later firmware by default permit multiple `telnet` sessions. However, in the case of a network partition, a server-capable administration node from each side of the partition will attempt to issue the fence commands, but only the node that is able to log in will succeed. Therefore, on a Brocade switch running 4.*x.x.x* or later firmware, you must modify the `admin` account to restrict it to a single `telnet` session. For details, see the release notes.

The switch then blocks the problem node from communicating with the storage area network (SAN) resources via the corresponding HBA. Figure 1-8 on page 33, describes this.

If users require access to nonclustered LUNs or devices in the SAN, these LUNs/devices must be accessed or mounted via an HBA that has been explicitly masked from fencing. For details on how to exclude HBAs from fencing for nodes, see:

- "Define a Switch with the GUI" on page 196
- "Define a Switch with `cmgr`" on page 273

For nodes running other supported operating systems, see *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

To recover, the affected node withdraws from the CXFS kernel membership, unmounts all file systems that are using an I/O path via fenced HBA(s), and then rejoins the cluster. This process is called *fencing recovery* and is initiated automatically. Depending on the failure action hierarchy that has been configured, a node may be reset (rebooted) before initiating fencing recovery. For information about setting the failure action hierarchy, see "Define a Node with `cmgr`" on page 224, and "Define a Node with the GUI" on page 172.

In order for a fenced node to rejoin the CXFS kernel membership, the current cluster leader must lower its fence to allow it to reprobe its XVM volumes and then remount its filesystems. If a node fails to rejoin the CXFS kernel membership, it may remain fenced. This is independent of whether the node was rebooted, because fencing is an operation applied on the switch, not the affected node. In certain cases, it may therefore be necessary to manually lower a fence. For instructions, see "Lower the

I/O Fence for a Node with the GUI" on page 200, and "Lower the I/O Fence for a Node with `cmgr`" on page 275.

Caution: When a fence is raised on an HBA, **no further I/O is possible to the SAN via that HBA until the fence is lowered**. This includes the following:

- I/O that is queued in the kernel driver, on which user processes and applications may be blocked waiting for completion. These processes will return the `EIO` error code under UNIX, or display a warning dialog that I/O could not be completed under Windows.
- I/O issued via the affected HBAs to nonclustered (local) logical units (LUNs) in the SAN or to other Fibre Channel devices such tape storage devices.

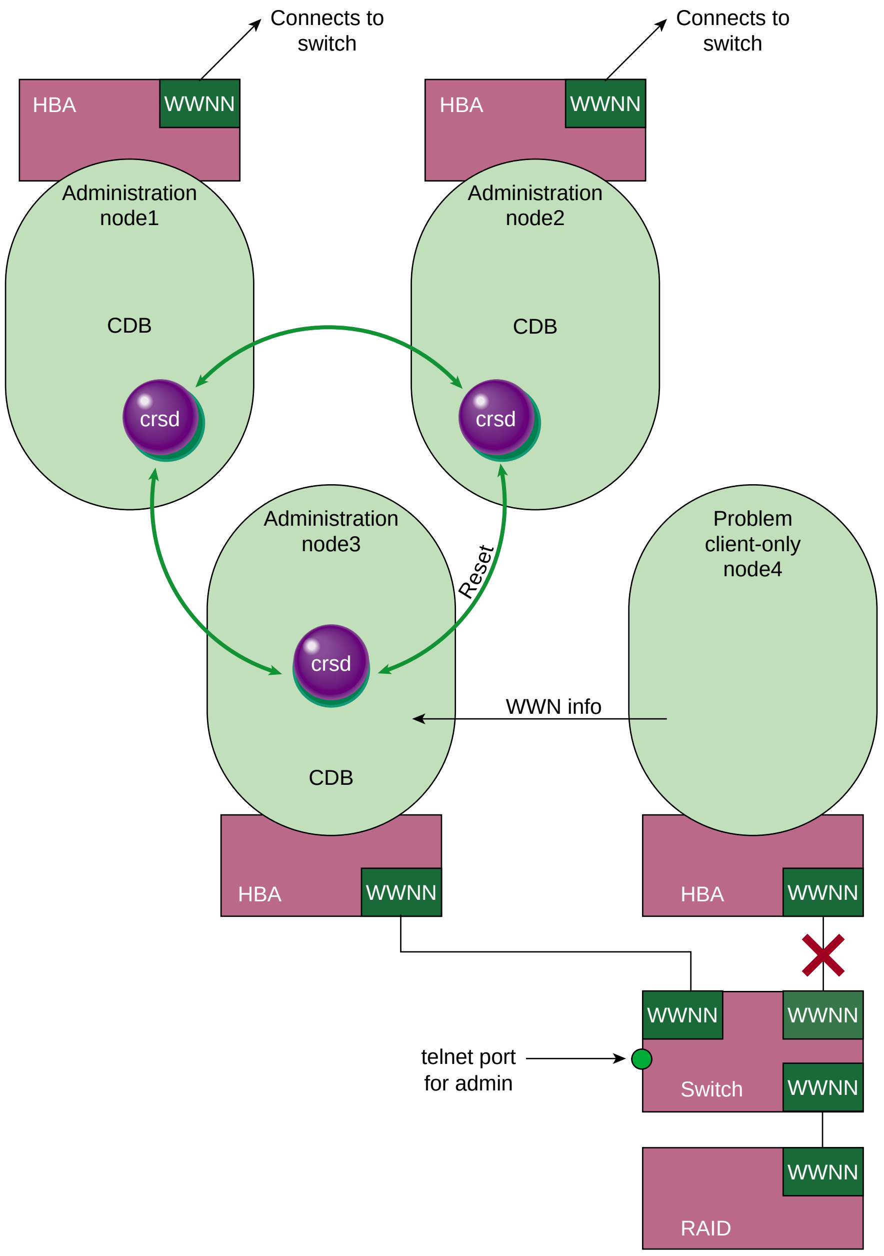


Figure 1-8 I/O Fencing

For more information, see "Switches and I/O Fencing Tasks with the GUI" on page 196, and "Switches and I/O Fencing Tasks with `cmgr`" on page 273.

Note: I/O fencing cannot be used for FailSafe nodes. FailSafe nodes require the system reset capability.

System Reset

The system reset can be one of the following methods:

- **Power Cycle** shuts off power to the node and then restarts it
- **Reset** simulates the pressing of the reset button on the front of the machine
- **NMI** (nonmaskable interrupt) performs a core-dump of the operating system kernel, which may be useful when debugging a faulty machine

On IRIX, the system reset connection has the same connection configuration as FailSafe; for more information, contact SGI professional or managed services. Also see Appendix F, "System Reset Configuration" on page 487.

The Cluster Database and CXFS Clients

The distributed cluster database (CDB) is central to the management of the CXFS cluster. Multiple synchronized copies of the database are maintained across the CXFS administration nodes in the pool (that is, those nodes installed with the `cluster_admin` software package). For any given CXFS Manager GUI task or `cmgr` task, the CXFS cluster daemons must apply the associated changes to the cluster database and distribute the changes to each CXFS administration node before another task can begin.

The client-only nodes in the pool do not maintain a local synchronized copy of the full cluster database. Instead, one of the daemons running on a CXFS administration node provides relevant database information to those nodes. If the set of CXFS administration nodes changes, another node may become responsible for updating the client-only nodes.

Metadata Server Functions

The metadata server must perform cluster-coordination functions such as the following:

- Metadata logging
- File locking
- Buffer coherency
- Filesystem block allocation

All CXFS requests for metadata are routed over a TCP/IP network and through the metadata server, and all changes to metadata are sent to the metadata server. The metadata server uses the advanced XFS journal features to log the metadata changes. Because the size of the metadata is typically small, the bandwidth of a fast Ethernet local area network (LAN) is generally sufficient for the metadata traffic.

The operations to the CXFS metadata server are typically infrequent compared with the data operations directly to the disks. For example, opening a file causes a request for the file information from the metadata server. After the file is open, a process can usually read and write the file many times without additional metadata requests. When the file size or other metadata attributes for the file change, this triggers a metadata operation.

The following rules apply:

- Any node installed with the `cluster_admin` product can be defined as a server-capable administration node.
- Although you can configure multiple server-capable CXFS administration nodes to be potential metadata servers for a given filesystem, only the first of these nodes to mount the filesystem will become the *active* metadata server. The list of potential metadata servers for a given filesystem is ordered, but because of network latencies and other unpredictable delays, it is impossible to predict which node will become the active metadata server.
- A single server-capable node in the cluster can be the active metadata server for multiple filesystems at once.
- There can be multiple server-capable nodes that are active metadata servers, each with a different set of filesystems. However, a given filesystem has a single active metadata server on a single node.

- If the last potential metadata server for a filesystem goes down while there are active CXFS clients, all of the clients will be forced out of the filesystem. (If another potential metadata server exists in the list, recovery will take place. For more information, see "Metadata Server Recovery" on page 299.)
- If you are exporting the CXFS filesystem to be used with other NFS clients, the filesystem should be exported from the active metadata server for best performance. For more information on NFS exporting of CXFS filesystems, see "CXFS Mount Scripts" on page 293.

For more information, see "Flow of Metadata for Reads and Writes" on page 454.

System View

CXFS provides a single-system view of the filesystems; each host in the SAN has equally direct access to the shared disks and common pathnames to the files. CXFS lets you scale the shared-filesystem performance as needed by adding disk channels and storage to increase the direct host-to-disk bandwidth. The CXFS shared-file performance is not limited by LAN speeds or a bottleneck of data passing through a centralized fileserver. It combines the speed of near-local disk access with the flexibility, scalability, and reliability of clustering.

CXFS and Highly Available Services

You can use one of the following products or to provide highly available services (such as for NFS or Web) running on a CXFS filesystem:

- IRIX systems: IRIX FailSafe on IRIX systems
- Linux systems: SGI Cluster Manager for Linux (requires both the base and storage software plug-in packages)

The CXFS plug-in moves the CXFS metadata server along with applications that must run on the metadata server, such as DMF. This combination of CXFS and FailSafe or SGI Cluster Manager for Linux provides high-performance shared data access for highly available applications.

CXFS and IRIX FailSafe share the same infrastructure. SGI Cluster Manager for Linux has a separate infrastructure.

Hardware and Software Support

This section discusses the following:

- "Requirements"
- "Compatibility" on page 39

Requirements

CXFS requires the hardware and software specified in the release notes:

- All server-capable administration nodes must run the same type of operating system.
- A supported SAN hardware configuration.

  Note: For details about supported hardware, see the Entitlement Sheet that accompanies the release materials. Using unsupported hardware constitutes a breach of the CXFS license.

- Use a network switch. (A network hub is not supported.) The switch should be at least 100baseT.
- A private 100baseT or Gigabit Ethernet TCP/IP network connected to each node.

  Note: When using Gigabit Ethernet, do not use jumbo frames. For more information, see the `tgconfig` man page.

- Serial lines and/or supported Fibre Channel switches. For supported switches, see the release notes.

  Either system reset or I/O fencing is required for all nodes. SGI recommends system reset for potential metadata servers. A cluster should have an odd number of server-capable nodes.

- At least one host bus adapter (HBA) as specified in the release notes.
- RAID hardware as specified in the release notes.

- Adequate compute power for CXFS nodes, particularly metadata servers, which must deal with the required communication and I/O overhead. There should be at least 2 GB of RAM on the system.

  A metadata server must have at least 1 processor and 1 GB of memory more that what it would need for its normal workload (non-CXFS work). In general, this means that the minimum configuration would be 2 processors and 2 GB of memory. If the metadata server is also doing NFS or Samba serving, then more memory is recommended (and the `nbuf` and `ncsize` kernel parameters should be increased from their defaults).

  CXFS makes heavy use of memory for caching. If a very large number of files (tens of thousands) are expected to be open at any one time, additional memory over the minimum is also recommended. Use the following to determine the amount of memory required for your system:

  2KB x *number_of_inodes* = *metadata_server_memory*

  In addition, about half of a CPU should be allocated for each Gigabit Ethernet interface on the system if it is expected to be run a close to full speed.

- To avoid problems during metadata server recovery/relocation, all potential metadata servers should have as much memory as the active metadata server.

- A FLEXlm license key for CXFS. Linux also requires a license for XVM.

  XVM provides a mirroring feature. If you want to access a mirrored volume from a given node in the cluster, you must purchase the XFS Volume Plexing software option and obtain and install a FLEXlm license. Except for Linux systems, which always require an XVM license, only those nodes that will access the mirrored volume must be licensed. For information about purchasing this license, see your sales representative.

- The XVM volume manager, which is provided as part of the IRIX release.

- If you use I/O fencing and `ipfilterd` on a node, the `ipfilterd` configuration must allow communication between the node and the `telnet` port on the switch.

A cluster is supported with as many as 64 nodes, of which as many as 16 can be server-capable administration nodes.

A cluster in which both CXFS and FailSafe are run (known as *coexecution*) is supported with a maximum of 64 nodes, as many as 8 of which can run FailSafe. The administration nodes must run IRIX; FailSafe is not supported on Linux nodes. Even when running with FailSafe, there is only one pool and one cluster. See "Overview of FailSafe Coexecution" on page 39, for further configuration details.

Compatibility

CXFS is compatible with the following:

- Data Migration Facility (DMF) and Tape Management Facility (TMF).
- Trusted IRIX. CXFS has been qualified in an SGI Trusted IRIX cluster with the Data Migration Facility (DMF) and Tape Management Facility (TMF).

  If you want to run CXFS and Trusted IRIX, all server-capable administration nodes must run Trusted IRIX. Client-only nodes can be running IRIX. For more information, see Chapter 14, "Trusted IRIX and CXFS" on page 357.
- FailSafe (coexecution). See the "Overview of FailSafe Coexecution" on page 39, and the *FailSafe Administrator's Guide for SGI InfiniteStorage*.
- SGI Cluster Manager for Linux. See the *SGI InfiniteStorage Cluster Manager for Linux Administrator's Guide*.

Overview of FailSafe Coexecution

CXFS 6.5.10 or later and IRIS FailSafe 2.1 or later (plus relevant patches) may be installed and run on the same system.

A subset of nodes in a coexecution cluster can be configured to be used as FailSafe nodes; a coexecution cluster can have up to eight nodes that run FailSafe.

The cluster database contains configuration information about nodes, the cluster, logging information, and configuration parameters. If you are running CXFS, it also contains information about CXFS filesystems and CXFS metadata servers, which coordinate the information that describes a file, such as the file's name, size, location, and permissions; there is one active metadata server per CXFS filesystem. If you are running FailSafe, it also contains information about resources, resource groups, and failover policies. Figure 1-9 depicts the contents of a coexecution cluster database.

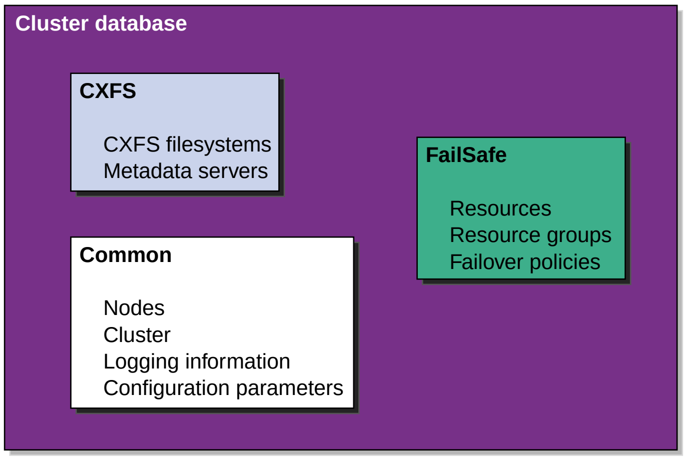


Figure 1-9 Contents of a Coexecution Cluster Database

In a coexecution cluster, a subset of the nodes can run FailSafe but all of the nodes must run CXFS. If you have both FailSafe and CXFS running, the products share a single cluster and a single database. There are separate configuration GUIs for FailSafe and CXFS; the `cmgr` command performs configuration tasks for both CXFS and FailSafe in command-line mode. You can also view cluster information with the `clconf_info` command.

The administration nodes can perform administrative tasks for FailSafe or CXFS and they run the `fs2d` cluster database daemon, which manages the cluster database and propagates it to each administration node in the pool. All FailSafe nodes are administration nodes, but some CXFS nodes do not perform administration tasks and are known as client-only nodes.

For more information, see Chapter 13, "Coexecution with FailSafe" on page 349.

CXFS Tools Overview

CXFS provides a set of tools to manage the cluster. These tools execute only on the appropriate node types:

- Administration nodes:
  - `cxfsmgr`, which invokes the CXFS Manager graphical user interface (GUI)

  - `cmgr` (also known as `cluster_mgr`)
  - `clconf_info`
  - `cxfs-config`
- Client-only nodes:
  - `cxfs_info`

Note: The GUI must be connected to a CXFS administration node, but it can be launched elsewhere; see "Starting the GUI" on page 150.

You can perform CXFS configuration tasks using either the GUI or the `cmgr` cluster manager command. These tools update the cluster database, which persistently stores metadata and cluster configuration information.

Although these tools use the same underlying software command line interface (CLI) to configure and monitor a cluster, the GUI provides the following additional features, which are particularly important in a production system:

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

The `cmgr` command is more limited in its functions. It enables you to configure and administer a cluster system only on a CXFS administration node (one that is installed with the `cluster_admin` software product). It provides a minimum of help and formatted output and does not provide dynamic status except when queried. However, an experienced administrator may find `cmgr` to be convenient when performing basic configuration tasks or isolated single tasks in a production environment, or when running scripts to automate some cluster administration tasks. You can use the `build_cmgr_script` command to automatically create a `cmgr` script based on the contents of the cluster database.

After the associated changes are applied to all online database copies in the pool, the view area in the GUI will be updated. You can use the GUI or the `cmgr`, and `clconf_info` commands to view the state of the database. (The database is a collection of files, which you cannot access directly.) On a client-only node, you can use the `cxfs_info` command.

For more details, see the following:

- "GUI Overview" on page 149
- "`cmgr` Overview" on page 218
- "Creating a `cmgr` Script Automatically" on page 280
- Chapter 15, "Monitoring Status" on page 359

Guaranteed-Rate I/O (GRIO) Version 2 and CXFS

CXFS supports guaranteed-rate I/O (GRIO) version 2 clients on all platforms, with a GRIO server on IRIX nodes. GRIO does the following:

- Enables a cluster node and/or a user application to reserve part of a filesystem's I/O bandwidth for its exclusive use
- Guarantees delivery of data from a storage device at a predefined rate, regardless of any other I/O activity on the system or on other nodes in the cluster
- Ensures that the rate at which a process issues I/O does not exceed its guarantee and will throttle the I/O if necessary

GRIO includes the following features:

- Support for CXFS filesystems shared among nodes in a cluster as well as locally attached XFS filesystems
- A simple filesystem-level performance qualification model
- A range of tools for monitoring and measuring delivered bandwidth and I/O service time

GRIO is disabled by default on Linux. To enable GRIO, change the following line in `/etc/cluster/config/cxfs_client.options` from:

```
export GRIO2=off
```

to:

```
export GRIO2=on
```

For other platform-specific limitations and considerations, see the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

For details about GRIO installation, configuration, and use, see the *Guaranteed-Rate I/O Version 2 Guide*.

GRIO Commands

Once installed in a cluster, the following commands can be run from any node in the cluster as a superuser:

- `grioadmin` provides stream and bandwidth management
- `grioqos` is the comprehensive stream quality-of-service monitoring tool

Run any of the above tools with the `-h` (help) option for a full description of all available options.

The paths to the GRIO commands differs by platform. See Appendix D, "Operating System Path Differences" on page 483.

GRIO Examples

The following sections show example GRIO commands. The examples are taken from a cluster with the following GRIO-managed filesystems:

```
highbw
lowbw
largefs
```

List GRIO-Managed Volumes

To list the GRIO-managed volumes:

```
grioadmin -l
```

For example:

```
irix# grioadmin -l
/mnt/highbw  DYNAMIC 001e88dd-77a4-1027-834c-0800690d75f7 bytes:  65536 msecs: 1000
/mnt/lowbw  DYNAMIC 9eb10270-7871-1027-8482-0800690d75f7 bytes:  65536 msecs: 1000
/mnt/largefs  DYNAMIC de7f81d5-76af-1027-8cfa-0800690d8e03 bytes:  65536 msecs: 1000
```

Monitor a Particular Stream

To monitor a particular GRIO stream, use the following command:

`grioqos -I` *intervals stream* `[`*delay*`|`*count*`]`

For example, to monitor the `NONGRIO` stream associated with `highbw` filesystem:

```
irix# grioqos -I 1000ms 001e88dd-77a4-1027-834c-0800690d75f7 1

IRIX64 irix 6.5 04070317 IP27         04/06/05

Filesystem: /mnt/highbw

Dynamic    0.06 MB/s 001e88dd-77a4-1027-834c-0800690d75f7
Static     None

-            interval   minbw   maxbw  lastbw minio maxio lastio
-                   -    MB/s    MB/s    MB/s    ms    ms     ms
10:28:38       1000ms    0.00    0.20    0.19   0.3   3.3    1.6
10:28:39       1000ms    0.00    0.20    0.19   0.3   3.3    1.6
10:28:40       1000ms    0.00    0.20    0.19   0.3   3.3    1.6
```

To terminate `grioqos`, press `Ctrl-C` on Windows or send a `SIGINT` on other platforms.

XMV Failover

There are two versions of XVM failover. CXFS support varies by platform.

XVM Failover Version 1

CXFS supports XVM failover version 1 on the following platforms:

- IRIX
- SGI ProPack for Linux

Version 1 failover requires that the RAID be configured in RDAC mode. It uses the `scsifo`(1M) command and `/etc/failover.conf` file. For more information, see the `scsifo`(1M) and `failover`(1M) man pages.

Note: Failover is independent of `xvm`, and the `xvm` failover version 2 commands do not support failover version 1. See "XVM Failover Version 2" on page 45.

XVM Failover Version 2

CXFS supports XVM failover version 2 on the following platforms:

- IRIX
- SGI ProPack for Linux
- AIX
- Linux third-party
- Mac OS X
- Solaris

To configure version 2 failover, you must edit the `/etc/failover2.conf` file. For more information, see the comments in the `/etc/failover2.conf` file and the *XVM Volume Manager Administrator's Guide*.

Following is an example of the `/etc/failover2.conf` file on IRIX for a configuration in which each node has dual ports connecting to separate Fibre Channel

(FC) switches. Each FC switch is connected to a single RAID controller with no interconnect between the FC switches (the RAID controllers are on the same RAID).

```
# phys/d9400_0
/dev/dsk/200500a0b80cedb3/lun0vol/c2p1 affinity=0 preferred
/dev/dsk/200400a0b80cedb3/lun0vol/c3p1 affinity=1

# phys/d9400_1
/dev/dsk/200400a0b80cedb3/lun1vol/c3p1 affinity=0 preferred
/dev/dsk/200500a0b80cedb3/lun1vol/c2p1 affinity=1

# phys/wally
/dev/dsk/200400a0b80cedb3/lun2vol/c3p1 affinity=0 preferred
/dev/dsk/200500a0b80cedb3/lun2vol/c2p1 affinity=1

# phys/recovery
/dev/dsk/200500a0b80cedb3/lun3vol/c2p1 affinity=0 preferred
/dev/dsk/200400a0b80cedb3/lun3vol/c3p1 affinity=1
```

Following is a parallel example for SGI ProPack for Linux:

```
# phys/d9400_0
/dev/xscsi/pci01.02.1/node200500a0b80cedb3/port1/lun0/disc affinity=0 preferred
/dev/xscsi/pci01.03.1/node200400a0b80cedb3/port1/lun0/disc affinity=1

# phys/d9400_1
/dev/xscsi/pci01.03.1/node200400a0b80cedb3/port1/lun1/disc affinity=0 preferred
/dev/xscsi/pci01.02.1/node200500a0b80cedb3/port1/lun1/disc affinity=1

# phys/wally
/dev/xscsi/pci01.03.1/node200400a0b80cedb3/port1/lun2/disc affinity=0 preferred
/dev/xscsi/pci01.02.1/node200500a0b80cedb3/port1/lun2/disc affinity=1

# phys/recovery
/dev/xscsi/pci01.02.1/node200500a0b80cedb3/port1/lun3/disc affinity=0 preferred
/dev/xscsi/pci01.03.1/node200400a0b80cedb3/port1/lun3/disc affinity=1
```

For other platform-specific examples of `/etc/failover2.conf`, see the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

The easiest method to generate a default `failover2.conf` file is to run the following command:

```
xvm show -v phys | fgrep affinity
```

For more information, see *XVM Volume Manager Administrator's Guide*

Overview of the Installation and Configuration Steps

This section provides an overview of the installation, verification, and configuration steps for IRIX and for SGI ProPack for Linux.

CXFS Packages Installed

Different packages are installed on CXFS administration nodes and client-only nodes.

Client-Only Packages Installed

The following packages are installed on a client-only node:

- Application binaries, documentation, and support tools:

  ```
  cxfs_client
  cxfs_util
  ```

- Kernel libraries:

  ```
  cxfs
  eoe.sw.xvm
  ```

Administration Packages Installed

The following packages are installed on an administration node:

- Application binaries, documentation, and support tools:

  ```
  cluster_admin
  cluster_control
  cluster_services
  cxfs_cluster
  cxfs_util
  ```

- Kernel libraries:

  ```
  cxfs
  eoe.sw.xvm
  ```

- GUI tools:

  ```
  sysadm_base
  sysadm_cluster
  sysadm_cxfs
  sysadm_xvm
  ```

CXFS Commands Installed

Different commands are installed on CXFS administration nodes and client-only nodes.

Client-Only Commands Installed

The following commands are shipped as part of the CXFS client-only package:

```
/usr/cluster/bin/cxfs_client (the CXFS client service)
/usr/cluster/bin/cxfs-config
/usr/cluster/bin/cxfsdump
/usr/cluster/bin/cxfslicense
```

These commands provide all of the services needed to include an IRIX or a Linux client-only node.

For more information, see the `cxfs_client` and `cxfsdump` man pages.

Administration Commands Installed

The following commands are shipped as part of the CXFS administration package:

```
/usr/cluster/bin/clconf_info
/usr/cluster/bin/clconf_stats
/usr/cluster/bin/clconf_status
/usr/cluster/bin/clconfd
/usr/cluster/bin/cxfs-config
/usr/cluster/bin/cxfs_shutdown
/usr/cluster/bin/cxfsdump
/usr/cluster/bin/hafence
```

```
/usr/cluster/bin/cxfslicense
```

IRIX Overview

Following is the order of installation and configuration steps for an IRIX node:

1. Install IRIX 6.5.*x* according to the *IRIX 6.5 Installation Instructions* (if not already done). See the CXFS IRIX release notes for supported levels.
2. Install and verify the RAID. See the release notes.
3. Install and verify the switch. See the release notes.
4. Obtain and install the CXFS license. If you want to access an XVM mirrored volume from a given node in the cluster, you must purchase a mirroring software option and obtain and install a FLEXlm license. Only those nodes that will access the mirrored volume must be licensed. For information about purchasing this license, see your sales representative. See Chapter 2, "CXFS and XVM FLEXlm Licenses" on page 51.
5. Prepare the node, including adding a private network. See "Adding a Private Network" on page 58.
6. Install the CXFS software. See Chapter 4, "IRIX CXFS Installation" on page 65.
7. Configure the cluster to define the new node in the pool, add it to the cluster, start CXFS services, and mount filesystems. See "Guided Configuration Tasks" on page 169.

SGI ProPack for Linux Overview

Following is the order of installation and configuration steps for a Linux node:

1. Read the release notes `README` file for the Linux platform to learn about any late-breaking changes in the installation procedure.
2. Install the SGI ProPack for Linux release, according to the directions in the SGI ProPack documentation. Ensure that you select the `SGI Licensed` package group for installation. See the CXFS SGI ProPack release notes for supported levels.
3. Install and verify the RAID. See the release notes.

4. Install and verify the switch. See the release notes.
5. Obtain and install the CXFS license. If you want to access an XVM mirrored volume from a given node in the cluster, you must purchase a mirroring software option and obtain and install a FLEXlm license. Only those nodes that will access the mirrored volume must be licensed. For information about purchasing this license, see your sales representative. See Chapter 2, "CXFS and XVM FLEXlm Licenses" on page 51.
6. Prepare the node, including adding a private network. See "Adding a Private Network" on page 58.
7. Install the CXFS software. See Chapter 5, "Linux CXFS Installation" on page 83.
8. Configure the cluster to define the new node in the pool, add it to the cluster, start CXFS services, and mount filesystems. See "Guided Configuration Tasks" on page 169.

CXFS and XVM FLEXlm Licenses

The software licensing used by CXFS is based on the FLEXlm product from Macrovision Corporation. For all nodes in the cluster, a FLEXlm license is required to use CXFS. Perform the procedures in this chapter to satisfy this requirement.

XVM provides a mirroring feature. If you want to access a mirrored volume from a given node in the cluster, you must install a FLEXlm mirroring license on that node. Only those nodes that will access the mirrored volume must be licensed. For information about purchasing this license, see your sales representative.

This section discusses the following:

- "Gathering the Host Information Required for the License"
- "Obtaining the Licenses from SGI" on page 54
- "Installing the Licenses" on page 54
- "Verifying the Licenses" on page 54
- "For More Information About Licensing" on page 56

Gathering the Host Information Required for the License

When you order CXFS, you will receive an entitlement ID. You must submit the system host ID, host name, and entitlement ID when requesting your permanent CXFS license. The method used to obtain this information is platform-specific.

IRIX Host Information

To obtain the host identifier and hostname of the system on which you will run CXFS, see the information in the CXFS release notes.

Linux Host Information

You must obtain the host identifier, hostname, serial number, and CPU count of the system on which you will run CXFS.

ProPack 3 Host Information

Execute the following commands to obtain the host information for ProPack 3:

```
cat /proc/sgi_sn/system_serial_number
/bin/hostname
lmhostid
hinv -c processor
```

The system serial number allows SGI to generate multiple host ID keys for Altix partitioning.

For example:

```
[root@linux64 root]# cat /proc/sgi_sn/system_serial_number
N0000002

[root@linux64 root]# /bin/hostname
cxfslinux

[root@linux64 root]# lmhostid
lmhostid - Copyright (c) 1989-2004 by Macrovision Corporation. All
rights reserved.
The FLEXlm host ID of this machine is "e0000010"

root@linux64 root]# hinv -c processor
1 Ix-Brick
1 C-Brick
4   1300 MHz Itanium 2 Rev. 5 Processor
```

In this case, the host identifier is `e0000010` and there are 4 processors.

When you are asked for the license manager host identifier, provide this information. You must have a separate license for each host on which CXFS is installed.

ProPack 4 Host Information

Execute the following commands to obtain the host information for ProPack 4:

```
cat /proc/sgi_sn/system_serial_number
/bin/hostname
lmhostid
hwinfo --cpu
```

You can also use the `hwinfo --short --cpu` command to get an abbreviated listing and then count the number of processors displayed.

For example:

```
[root@linux64 root]# cat /proc/sgi_sn/system_serial_number
N0000002

[root@linux64 root]# /bin/hostname
cxfslinux

[root@linux64 root]# lmhostid
lmhostid - Copyright (c) 1989-2004 by Macrovision Corporation. All
rights reserved.
The FLEXlm host ID of this machine is "e0000010"

[root@linux64 root]# /usr/sbin/hwinfo --short --cpu
cpu:
                        Itanium 2, 1300 MHz
                        Itanium 2, 1300 MHz
                        Itanium 2, 1300 MHz
                        Itanium 2, 1300 MHz
                        Itanium 2, 1300 MHz
                        Itanium 2, 1300 MHz
                        Itanium 2, 1300 MHz
                        Itanium 2, 1300 MHz


[root@linux64 root]# /usr/sbin/hwinfo --short --cpu | grep Itanium | wc -l
8
```

In this case, the host identifier is `e0000010` and there are 8 processors.

The system serial number allows SGI to generate multiple host ID keys for Altix partitioning.

When you are asked for the license manager host identifier, provide this information. You must have a separate license for each host on which CXFS is installed.

Obtaining the Licenses from SGI

Along with your entitlement number, you will receive a URL to a key generation page. To obtain your permanent CXFS and XVM licenses, follow the instructions on the key generation page. After the required information is provided, a key will be generated and displayed on the webpage along with installation instructions.

Installing the Licenses

You will install the licenses in the following location, according to platform:

- IRIX: `/var/flexlm/license.dat`
- Linux: `/etc/flexlm/license.dat`

Do the following:

1. Create the license directory if necessary.

   For example, on Linux:

   ```
   [root@linux cdrom]# mkdir -p /etc/flexlm
   ```

2. Copy the key to the `license.dat` file.

Note: If you increase the number of CPUs in your system, you may need a new license. Partitioned Origin 3000 and Onyx 3000 systems upgrading to IRIX 6.5.15f or later will require replacement licenses. Prior to IRIX 6.5.15f, these partitioned systems used the same `lmhostID` to license all the partitions in the system. For more information, see the 6.5.15 *Start Here/Welcome* and the following web page: http://www.sgi.com/support/licensing.

Verifying the Licenses

To verify that the licenses have been installed properly, use the `cxfslicense` command after installing the CXFS software.

If the license is installed properly, there will be no output. If it is not installed, there will be errors.

For more verbose output, use the `-d` option. For example:

- License installed properly:

  ```
  irix# /usr/cluster/bin/cxfslicense
  irix# /usr/cluster/bin/cxfslicense -d
  no xvm user license required.
  XLV license granted.
  CXFS license granted.
  irix#
  ```

- No license (error condition):

  ```
  irix# /usr/cluster/bin/cxfslicense
  cxfs license check failed - use ’cxfslicense -d’ for details
  irix# /usr/cluster/bin/cxfslicense -d
  no xvm user license required.
  XLV license granted.
  Error on CXFS license check out.
  checkout failed: No such feature exists
  Feature:       CXFS
  License path:  /etc/flexlm/license.dat:/var/flexlm/license.dat
  FLEXlm error:  -5,357.  System Error: 2 "No such file or directory"
  For further information, refer to the FLEXlm End User Manual,
  available at "www.globetrotter.com".
  ```

If you do not have the CXFS license properly installed, you will see the following error on the console when trying to run CXFS:

```
Starting CXFS services> ....
CXFS not properly licensed for this host.  Run
   ’/usr/cxfs_cluster/bin/cxfslicense -d’
for detailed failure information. After fixing the
license, please run ’/usr/cxfs_cluster/bin/cxfs_cluster restart’.
```

An error such as the following example will appear in the `SYSLOG` file (line breaks added here for readability):

```
Jan 25 10:24:03 ncc1701:Jan 25 10:24:03 cxfs_client:
cis_main FATAL: cxfs_client failed the CXFS license check.
Use the cxfslicense command to diagnose the license problem
```

On an administration node, the error will appear in the `clconfd` log.

For More Information About Licensing

To request software keys or information about software licensing, see the following web page:

http://www.sgi.com/support/licensing

If you do not have access to the web, please contact your local Customer Support Center.

For more information about installing IRIX software licenses, see the *IRIX 6.5 Installation Instructions* booklet.

For more information on FLEXlm, you may order the *Flexible License Manager End User Manual* from Macrovision Corporation.

Chapter 3

Preinstallation Steps

When you install the CXFS software, you must modify certain system files. The network configuration is critical. Each node in the cluster must be able to communicate with every other node in the cluster by both logical name and IP address without going through any other network routing; proper name resolution is key. SGI recommends static routing.

This section provides an overview of the steps that you should perform on your nodes prior to installing the CXFS software. It contains the following sections:

- "Hostname Resolution and Network Configuration Rules"
- "Configuring Network Interfaces"
- "Verifying the Private and Public Networks" on page 62
- "Configuring the Serial Ports for IRIX Administration Nodes" on page 63

Hostname Resolution and Network Configuration Rules

Caution: It is critical that you understand these rules before attempting to configure a CXFS cluster.

Use the following hostname resolution rules and recommendations when defining a node:

- The first node you define in the pool must be an administration node.
- Hostnames cannot begin with an underscore (_) or include any white-space characters.
- The private network IP addresses on a running node in the cluster cannot be changed while CXFS services are active.
- You must be able to communicate directly between every node in the cluster (including client-only nodes) using IP addresses and logical names, without routing.

- A private network must be dedicated to be the heartbeat and control network. No other load is supported on this network.
- The heartbeat and control network must be connected to all nodes, and all nodes must be configured to use the same subnet.

If you change hostname resolution settings in the `/etc/nsswitch.conf` file after you have defined the first administration node (which creates the cluster database), you must re-create the cluster database.

Configuring Network Interfaces

When configuring your network, remember the following:

- You must be able to communicate between every node in the cluster directly using IP address and logical name, without routing.
- Dedicate a private network to be your heartbeat and control network. No other load is supported on this network.
- The heartbeat and control network must be connected to all nodes, and all nodes must be configured to use the same subnet for that network.

Adding a Private Network

The following procedure provides an overview of the steps required to add a private network.

Note: A private network is required for use with CXFS.

You may skip some steps, depending upon the starting conditions at your site.

1. Edit the `/etc/hosts` file so that it contains entries for every node in the cluster and their private interfaces as well.

   The `/etc/hosts` file has the following format, where *primary_hostname* can be the simple hostname or the fully qualified domain name:

   IP_address　　*primary_hostname*　　*aliases*

You should be consistent when using fully qualified domain names in the `/etc/hosts` file. If you use fully qualified domain names on a particular node, then all of the nodes in the cluster should use the fully qualified name of that node when defining the IP/hostname information for that node in their `/etc/hosts` file.

The decision to use fully qualified domain names is usually a matter of how the clients are going to resolve names for their client/server programs (such as NFS), how their default resolution is done, and so on.

Even if you are using the domain name service (DNS) or the network information service (NIS), you must add every IP address and hostname for the nodes to `/etc/hosts` on all nodes. For example:

```
190.0.2.1 server1-company.com server1
190.0.2.3 stocks
190.0.3.1 priv-server1
190.0.2.2 server2-company.com server2
190.0.2.4 bonds
190.0.3.2 priv-server2
```

You should then add all of these IP addresses to `/etc/hosts` on the other nodes in the cluster.

For more information, see the `hosts` and `resolve.conf` man pages.

Note: Exclusive use of NIS or DNS for IP address lookup for the nodes will reduce availability in situations where the NIS or DNS service becomes unreliable.

---

2. Edit the `/etc/nsswitch.conf` file so that local files are accessed before either NIS or DNS. That is, the `hosts` line in `/etc/nsswitch.conf` must list `files` first.

   For example:

   ```
   hosts:      files nis dns
   ```

   (The order of `nis` and `dns` is not significant to CXFS, but `files` must be first.)

3. Configure your private interface according to the instructions in the Network Configuration section of your Linux distribution manual. To verify that the private interface is operational, use the `ifconfig -a` command. For example:

```
[root@linux64 root]# ifconfig -a

eth0      Link encap:Ethernet  HWaddr 00:50:81:A4:75:6A
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:13782788 errors:0 dropped:0 overruns:0 frame:0
          TX packets:60846 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:826016878 (787.7 Mb)  TX bytes:5745933 (5.4 Mb)
          Interrupt:19 Base address:0xb880 Memory:fe0fe000-fe0fe038

eth1      Link encap:Ethernet  HWaddr 00:81:8A:10:5C:34
          inet addr:10.0.0.10  Bcast:10.0.0.255  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:19 Base address:0xef00 Memory:febfd000-febfd038

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:162 errors:0 dropped:0 overruns:0 frame:0
          TX packets:162 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:11692 (11.4 Kb)  TX bytes:11692 (11.4 Kb)
```

This example shows that two Ethernet interfaces, `eth0` and `eth1`, are present and running (as indicated by `UP` in the third line of each interface description).

If the second network does not appear, it may be that a network interface card must be installed in order to provide a second network, or it may be that the network is not yet initialized.

4. *(Optional)* Make the modifications required to use CXFS connectivity diagnostics. See "IRIX Modifications for CXFS Connectivity Diagnostics" on page 80, and "Linux Modifications for CXFS Connectivity Diagnostics" on page 93.

Configuring IRIX Interfaces

To configure IRIX network interfaces, do the following:

1. Ensure that name services are available. See step 1 in "Adding a Private Network" on page 58.

2. On one node, add that node's interfaces and their IP addresses to the `/etc/config/netif.options` file.

   For the example:

   ```
   if1name=ec0
   if1addr=$HOSTNAME
   ```

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

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

   ```
   if2name=
   if2addr=
   ```

   In the example, the control network name and IP address are as follows:

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

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

3. If there are more than eight interfaces on the node, change the value of `if_num` in `/etc/config/netif.options` to the number of interfaces. For fewer than eight interfaces, the line is as follows:

   ```
   if_num=8
   ```

4. Repeat steps 1 through 3 for the other nodes.

5. Edit the `/etc/config/routed.options` file on each IRIX node so that the routes are not advertised over the control network. See the `routed`(1M) man page for a list of options.

   For example:

   ```
   -q -h -Prdisc_interval=45
   ```

The options do the following:

- Turn off the advertising of routes
- Cause host or point-to-point routes to not be advertised (provided there is a network route going the same direction)
- Set the nominal interval with which Router Discovery Advertisements are transmitted to 45 seconds (and their lifetime to 135 seconds)

Verifying the Private and Public Networks

For each private network on each node in the pool, verify access with the `ping` command. Enter the following, where *nodeIPaddress* is the IP address of the node:

```
ping nodeIPaddress
```

For example:

```
[root@linux64 root]# ping 10.0.0.1
PING 10.0.0.1 (10.0.0.1) from 128.162.240.141 : 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=0.310 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.122 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.127 ms
```

Also execute a `ping` on the public networks. If `ping` fails, follow these steps:

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

   ```
   [root@linux64 root]# ifconfig eth1
   eth1      Link encap:Ethernet  HWaddr 00:81:8A:10:5C:34
             inet addr:10.0.0.10  Bcast:10.0.0.255  Mask:255.255.255.0
             UP BROADCAST MULTICAST  MTU:1500  Metric:1
             RX packets:0 errors:0 dropped:0 overruns:0 frame:0
             TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:100
             RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
             Interrupt:19 Base address:0xef00 Memory:febfd000-febfd038
   ```

   In the third output line above, `UP` indicates that the interface was configured up.

2. Verify that the cables are correctly seated.

Repeat this procedure on each node.

Configuring the Serial Ports for IRIX Administration Nodes

If one IRIX administration node is configured to reset another IRIX administration node, you must turn off the `getty` process for the tty ports to which the reset serial cables are connected. You must do this on the IRIX administration node performing the reset (not the node receiving the reset). To do this, perform the following steps on each IRIX administration node; if you have a cluster with nodes running other operating systems, see the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

1. Determine which port is used for the reset line. `ttyd2` is the most commonly used port, except on Origin 300 and Origin 350 system, where `ttyd4` is commonly used.
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, for an Origin 3000:

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

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

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

IRIX CXFS Installation

Caution: CXFS is a complex product. To ensure that it is installed and configured in an optimal manner, you **must** purchase initial setup services from SGI. This chapter is not intended to be used directly by the customer, but is provided for reference.

On IRIX nodes, CXFS supports either an *administration node* containing the cluster administration daemons (`fs2d`, `crsd`, `cad`, and `cmond`), the CXFS control daemon (`clconfd`), and the cluster database or a *client-only node* containing the `cxfs_client` daemon. The software you install on a node determines the node type.

Nodes that you intend to run as metadata servers must be installed as administration nodes; all other nodes should be client-only nodes.

You should read through this entire book, especially Chapter 17, "Troubleshooting" on page 381, before attempting to install and configure a CXFS cluster. If you are using coexecution with FailSafe, see the *FailSafe Administrator's Guide for SGI InfiniteStorage*. If you are using a multiOS cluster, see the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

This chapter discusses the following:

- "IRIX Administration Software Installation" on page 65
- "IRIX Client-only Software Installation" on page 76
- "IRIX Modifications for CXFS Connectivity Diagnostics" on page 80

IRIX Administration Software Installation

Any node that may be a CXFS metadata server must be installed as a CXFS administration node. All other nodes should be client-only nodes.

Note: An IRIX node can be either be a CXFS administration node (for which you install `cluster_admin`) or a client-only node (for which you install `cxfs_client`). You cannot install both `cluster_admin` and `cxfs_client` on the same node. This procedure installs an administration node; to install a client-only node, see "IRIX Client-only Software Installation" on page 76.

Installing the CXFS base CD for a CXFS administration node requires approximately 30.3 MB of space.

IRIX Administration Node Installation Procedure

To install the required IRIX software for a CXFS administration node, do the following:

1. On each CXFS administration node in the pool, upgrade to IRIX 6.5.*x* according to the *IRIX 6.5 Installation Instructions*.

   To verify that a given node has been upgraded, use the following command to display the currently installed system:

   ```
   # uname -aR
   ```

2. (*For sites with a serial port server*) On each CXFS administration node, install the version of the serial port server driver that is appropriate to the operating system. Use the CD that accompanies the serial port server. Reboot the system after installation.

   For more information, see the documentation provided with the serial port server.

3. On each CXFS administration node in the pool, do the following:

   a. Insert the *CXFS 3.4 IRIX Server and Client for IRIX 6.5.x* CD into the CD drive.

   b. Read the release notes for the CXFS IRIX platform to learn about any late-breaking changes in the installation procedure.

      To view the release notes before they are installed, choose the following from the desktop Toolchest to bring up the **Software Manager** window:

System
　　> Software Manager

Choose **Customize Installation** by typing `/CDROM/dist` into the **Available Software** box. A list of products available for installation will come up. If the product name is highlighted (similar to an HTML link), then there are release notes available. Simply click on the link to bring up the **Release Notes** window.

If you do not have access to a graphics terminal, you must install the release notes and then use the `relnotes` command to view the CXFS relnotes. For example:

```
# inst
Inst> from /CDROM/dist
Inst> keep *
Inst> install cxfs.man.relnotes
Inst> go
Inst> sh
# /usr/sbin/relnotes cxfs ChapterNumber
```

CXFS has the following chapters:

1	Introduction
2	Installation Information
3	Changes and Additions
4	Bug Fixes
5	Known Problems and Workarounds
6	Documentation Errors
7	Activating Your CXFS *x.x* and Cluster XVM for 6.5.*x* License With FLEXlm

c. Insert IRIX CD-ROM #1 into the CD drive.

d. Start up `inst`:

```
# inst
```

e. Instruct `inst` to read the already inserted CD-ROM as follows:

```
Inst> from /CDROM/dist
```

Caution: Do not install to an alternate root using the `inst -r` option. Some of the exit operations (exitops) do not use pathnames relative to the alternate root, which can result in problems on both the main and alternate root filesystem if you use the `-r` option. For more information, see the `inst` man page.

f. When you see the following message, press the `Enter` key to read the CD-ROM:

```
Install software from : [/CDROM/dist]
```

g. Install the XVM software:

```
Inst> keep *
Inst> install eoe.sw.xvm
Inst> install eoe.books.xvm
Inst> go
```

The following subsystems will be installed:

```
eoe.sw.xvm
eoe.books.xvm
```

h. If you want to use Guaranteed-rate I/O version 2 (GRIOv2), install `eoe.sw.grio2`:

```
Inst> keep *
Inst> install eoe.sw.grio2
Inst> go
```

i. If you want to use Performance Co-Pilot to run XVM statistics, install the default `pcp_eoe` subsystems and also select `pcp_eoe.sw.xvm`. This installs the Performance Co-Pilot PMDA (the agent to export XVM statistics) as an exit operation (exitop).

```
Inst> keep *
Inst> install pcp_eoe default
Inst> install pcp_eoe.sw.xvm
Inst> go
```

j. Insert IRIX CD-ROM #3 into the CD drive.

k. When you see the following message, press the `Enter` key to read the CD-ROM:

```
Install software from : [/CDROM/dist]
```

l. Install the base cluster software:

```
Inst> keep *
Inst> install cluster_admin
Inst> install cluster_control
Inst> install cluster_services
Inst> install sysadm_base
Inst> install sysadm_cluster
Inst> install sysadm_xvm
Inst> go
```

The following subsystems will be installed:

```
cluster_admin.man.man
cluster_admin.sw.base
cluster_control.man.man
cluster_control.sw.base
cluster_control.sw.cli
cluster_services.man.man
cluster_services.sw.base
cluster_services.sw.cli
sysadm_base.man.priv
sysadm_base.man.relnotes
sysadm_base.man.server
sysadm_base.sw.client
sysadm_base.sw.dso
sysadm_base.sw.priv
sysadm_base.sw.server
sysadm_cluster.man.relnotes
sysadm_cluster.sw.client
sysadm_cluster.sw.server
sysadm_xvm.man.pages
sysadm_xvm.man.relnotes
sysadm_xvm.sw.client
sysadm_xvm.sw.desktop
sysadm_xvm.sw.server
sysadm_xvm.sw.web
```

m. Insert the *CXFS 3.4 IRIX Server and Client for IRIX 6.5.x* CD into the CD drive.

Note: If you have a system running an earlier version of IRIX with CXFS installed and try to upgrade IRIX without also installing the required CXFS CD, you will get a conflict. You must either install the CXFS CD or remove CXFS.

n. Instruct `inst` to read the already inserted CD as follows:

```
Inst> from /CDROM/dist
```

o. When you see the following message, press the `Enter` key to read the CD-ROM:

```
Install software from : [/CDROM/dist]
```

p. Install the CXFS software:

```
Inst> keep *
Inst> install cxfs
Inst> install sysadm_cxfs
Inst> keep cxfs.sw.grio2_cell
Inst> install cxfs_cluster
Inst> install cxfs_util
Inst> go
```

Or, if you are installing GRIOv2:

```
Inst> keep *
Inst> install cxfs
Inst> install sysadm_cxfs
Inst> install cxfs_cluster
Inst> install cxfs_util
Inst> go
```

Caution: If you do not install `cxfs_cluster`, the `inst` utility will not detect a conflict, but the CXFS cluster will not work. You **must** install the `cxfs_cluster` subsystem.

The following subsystems will be installed:

```
cxfs.books.CXFS_AG
cxfs_cluster.man.man
cxfs_cluster.sw.base
cxfs_cluster.sw.cli
cxfs.sw.cxfs
cxfs.sw.xvm_cell
cxfs_util.man.man
cxfs_util.sw.base
sysadm_cxfs.man.pages
sysadm_cxfs.man.relnotes
sysadm_cxfs.sw.client
sysadm_cxfs.sw.desktop
sysadm_cxfs.sw.server
sysadm_cxfs.sw.web
```

When `sysadm_base` is installed, `tcpmux` service is added to the `/etc/inetd.conf` file.

Note: If you want to run the CXFS Manager graphical user interface (GUI) from a login other than `root`, you will also want to install `sysadmdesktop`. This action provides commands that allow you to give users privileges, including the privileges required to run the CXFS commands. If you install `sysadmdesktop`, you will install the following subsystems from the *Applications CD 1 of 2 for 6.5.x*:

```
sysadmdesktop.man.base
sysadmdesktop.man.relnotes
sysadmdesktop.sw.base
sysadmdesktop.sw.data
sysadmdesktop.sw.sysadm
```

---

4. If you want to use a web-based version of the GUI, the following subsystems must be installed on the CXFS administration nodes that you will connect to (by means of a Java-enabled web browser running on any platform) for performing administrative operations:

   ```
   sysadm_base.sw.client
   sysadm_cxfs.sw.client
   sysadm_cxfs.sw.web
   sysadm_xvm.sw.client
   ```

   These subsystems are part of the default software that was installed in step 3.

   If you want to use a web-based version of the GUI, you must also have one of the following installed:

   - `sgi_apache.sw.server`
   - `nss_enterprise.sw.server` (from the Netscape CD-ROM)

   If one of these subsystems is not already installed, you must load the appropriate CD-ROM and install the subsystem.

5. If you want to run the GUI client from an IRIX desktop (which can be a node in the cluster or outside of the cluster), install the following subsystems:

   ```
   Inst> keep *
   Inst> install java2_eoe.sw
   Inst> install java2_eoe.sw32
   Inst> install sysadm_base.man
   ```

```
Inst> install sysadm_base.sw.client
Inst> install sysadm_cluster.sw.client
Inst> install sysadm_cxfs.man
Inst> install sysadm_cxfs.sw.client
Inst> install sysadm_cxfs.sw.desktop
Inst> install sysadm_xvm.sw.client
Inst> install sysadm_xvm.sw.desktop
Inst> go
```

Caution: The GUI on IRIX only operates with Java2 v1.4.1 Execution Environment (Sun JRE v1.4.1). This is the version of Java that is provided with the supported IRIX 6.5.*x* release.

The SGI website also contains Java1. However, you cannot use this version of Java with the GUI. Using a Java version other than 1.4.1 will cause the GUI to fail.

6. If the workstation is an IRIX machine that launches the GUI client from a web browser that supports Java, install the `java_plugin` subsystem from the IRIX 6.5.*x* CD. This is the Runtime Plug-in for IRIX, Java Edition 1.4.1, which supports JRE 1.4.1. (However, launching the GUI from a web browser is not the recommended method on IRIX. Running the GUI client from an IRIX desktop, as in step 5 above, is preferred.)

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

7. Exit from `inst`:

   ```
   Inst> quit
   ```

   The process may take a few minutes to complete.

   After you have installed the software and quit the `inst` interface, you are prompted to reboot the system to apply the changes. However, you will reboot in step 9.

8. Use the `cxfslicense` command to verify that the license is installed in `/var/flexlm/license.dat`. If the license is installed properly, there will be no output. If it is not installed, there will be errors.

For more verbose output, use the `-d` option. For example, the following shows a properly installed license:

```
irix# /usr/cluster/bin/cxfslicense -d
no xvm user license required.
XLV license granted.
CXFS license granted.
irix#
```

If there are errors, verify that you have obtained and installed the CXFS licenses. For more information, see "Verifying the Licenses" on page 54.

9. Reboot the system.

Differences When Performing a Remote or Miniroot Install

If you perform a remote or miniroot install, the `exitop` commands will be deferred until the `fs2d`, `crsd`, `cad`, and `cmond` daemons are started, at which time the `exitop` commands will be run. While performing the installation, you will see informational "`can't run remotely`" messages among the normal set of messages. For example:

```
Removing orphaned directories
Installing/removing files ..  94%
Running exit-commands ..  94%
cluster_admin.sw.base: ( $rbase/usr/cluster/bin/cdb-exitop )
cdb-exitop: can't run remotely - scheduling to run later
Running exit-commands ..  95%
cluster_control.sw.base: ( $rbase/usr/cluster/bin/cluster_control-exitop )
cluster_control-exitop: can't run remotely - scheduling to run later
Running exit-commands ..  96%
cluster_services.sw.base: ( $rbase/usr/cluster/bin/cluster_ha-exitop )
cluster_ha-exitop: can't run remotely - scheduling to run later
Running exit-commands ..  97%
cxfs_cluster.sw.base: ( $rbase/usr/cluster/bin/cluster_cx-exitop )
cluster_cx-exitop: can't run remotely - scheduling to run later
Running exit-commands ..  99%
Checking dependencies .. 100% Done.
Installations and removals were successful.
You may continue with installations or quit now.
```

If you see the above informational messages during the install process, you will see the following messages when the `fs2d`, `crsd`, `cad`, and `cmond` daemons start up:

```
running /usr/cluster/bin/cdb-exitop
cdb-exitop: initializing CDB
cdb-exitop: success
running /usr/cluster/bin/cluster_control-exitop
cluster_control-exitop: success
running /usr/cluster/bin/cluster_ha-exitop
cluster_ha-exitop: Added HA keys to /var/cluster/cdb/cdb.db

         * * * * * * * * * * I M P O R T A N T * * * * * * * * * * * * *

         "sgi-cmsd" service MUST be added to /etc/services.
         Restart cluster processes after adding the entry.  Failure to do so
         will cause cluster and FailSafe services to function incorrectly.
         Please refer to the SGI FailSafe Administrator's Guide for more
         information.


         * * * * * * * * * * I M P O R T A N T * * * * * * * * * * * * *

         "sgi-gcd" service MUST be added to /etc/services.
         Restart cluster processes after adding the entry.  Failure to do so
         will cause cluster and FailSafe services to function incorrectly.
         Please refer to the SGI FailSafe Administrator's Guide for more
         information.

cluster_ha-exitop: success
running /usr/cluster/bin/cluster_cx-exitop
cluster_cx-exitop: Added CXFS keys to /var/cluster/cdb/cdb.db
cluster_cx-exitop: success
```

Note: The messages that say `sgi-cmsd` and `sgi-gcd` must be added to `/etc/services` are true only for coexecution with FailSafe; if you are running just CXFS, you do not need `sgi-cmsd`. CXFS does not require `sgi-cmsd`. For more information, see "`/etc/services` on CXFS Administration Nodes" on page 96

IRIX Client-only Software Installation

An IRIX node can be either be a CXFS administration node (for which you install `cluster_admin`) or a client-only node (for which you install `cxfs_client`). You cannot install both `cluster_admin` and `cxfs_client` on the same node. This procedure installs a client-only node; to install an administration node, see "IRIX Administration Software Installation" on page 65.

To install the required IRIX software, do the following:

1. On each IRIX client-only node in the pool, upgrade to IRIX 6.5.*x* according to the *IRIX 6.5 Installation Instructions*.

   To verify that a given node has been upgraded, use the following command to display the currently installed system:

   ```
   # uname -aR
   ```

2. (*For sites with a serial port server*) On each node, install the version of the serial port server driver that is appropriate to the operating system. Use the CD that accompanies the serial port server. Reboot the system after installation.

   For more information, see the documentation provided with the serial port server.

3. On each IRIX client-only node in the pool, do the following:

   a. Insert the *CXFS 3.4 IRIX Server and Client for IRIX 6.5.x* CD into the CD drive.

   b. Read the release notes for the CXFS IRIX platform to learn about any late-breaking changes in the installation procedure.

      To view the release notes before they are installed, choose the following from the desktop Toolchest to bring up the **Software Manager** window:

      System
          > Software Manager

      Choose **Customize Installation** by typing `/CDROM/dist` into the **Available Software** box. A list of products available for installation will come up. If the product name is highlighted (similar to an HTML link), then there are release notes available. Simply click on the link to bring up the **Release Notes** window.

If you do not have access to a graphics terminal, you must install the release notes and then use the `relnotes` command to view the CXFS relnotes. For example:

```
# inst
Inst> from /CDROM/dist
Inst> keep *
Inst> install cxfs.man.relnotes
Inst> go
Inst> sh
# /usr/sbin/relnotes cxfs ChapterNumber
```

CXFS has the following chapters:

1	Introduction
2	Installation Information
3	Changes and Additions
4	Bug Fixes
5	Known Problems and Workarounds
6	Documentation Errors
7	Activating Your CXFS *x.x* and Cluster XVM for 6.5.*x* License With FLEXlm

c. Start up `inst`:

```
# inst
```

d. Insert IRIX CD-ROM #1 into the CD drive.

e. Instruct `inst` to read the already inserted CD-ROM as follows:

```
Inst> from /CDROM/dist
```

Caution: Do not install to an alternate root using the `inst -r` option. Some of the exit operations (exitops) do not use pathnames relative to the alternate root, which can result in problems on both the main and alternate root filesystem if you use the `-r` option. For more information, see the `inst` man page.

f. When you see the following message, press the Enter key to read the CD-ROM:

```
Install software from : [/CDROM/dist]
```

g. Install the XVM software:

```
Inst> keep *
Inst> install eoe.sw.xvm
Inst> install eoe.books.xvm
Inst> go
```

The following subsystem will be installed:

```
   eoe.sw.xvm
   eoe.books.xvm
```

h. If you want to use Performance Co-Pilot to run XVM statistics, install the default `pcp_eoe` subsystems and also select `pcp_eoe.sw.xvm`. This installs the Performance Co-Pilot PMDA (the agent to export XVM statistics) as an exit operation (exitop).

i. Insert the *CXFS 3.4 IRIX Server and Client for IRIX 6.5.x* CD into the CD drive.

j. Instruct `inst` to read the already inserted CD-ROM as follows:

```
Inst> from /CDROM/dist
```

Note: If you have a system running an earlier version of IRIX with CXFS installed and try to upgrade IRIX without also installing the required CXFS CD, you will get a conflict. You must either install the CXFS CD or remove CXFS.

Caution: Do not install to an alternate root using the `inst -r` option. Some of the exit operations (exitops) do not use pathnames relative to the alternate root, which can result in problems on both the main and alternate root filesystem if you use the `-r` option. For more information, see the `inst` man page.

k. When you see the following message, press the Enter key to read the CD-ROM:

```
Install software from : [/CDROM/dist]
```

l. Install the CXFS software:

```
Inst> keep *
Inst> install cxfs
Inst> install cxfs_client
Inst> install cxfs_util
Inst> go
```

Caution: If you do not install cxfs_client, the inst utility will not detect a conflict, but the CXFS cluster will not work. You **must** install the cxfs_client subsystem.

The following subsystems will be installed:

```
cxfs.books.CXFS_AG
cxfs_client.man.man
cxfs_client.sw.base
cxfs.sw.cxfs
cxfs.sw.xvm_cell
cxfs_util.man.man
cxfs_util.sw.base
```

4. Exit from inst:

```
Inst> quit
```

The process may take a few minutes to complete.

After you have installed the software and quit the inst interface, you are prompted to reboot the system to apply the changes. However, you will reboot in step 6.

5. Use the cxfslicense command to verify that the license is installed in /var/flexlm/license.dat. If the license is installed properly, there will be no output. If it is not installed, there will be errors.

For more verbose output, use the `-d` option. For example, the following shows a properly installed license:

```
irix# /usr/cluster/bin/cxfslicense -d
no xvm user license required.
XLV license granted.
CXFS license granted.
irix#
```

If there are errors, verify that you have obtained and installed the CXFS licenses. For more information, see "Verifying the Licenses" on page 54.

6. Reboot the system.

IRIX Modifications for CXFS Connectivity Diagnostics

If you want to use the connectivity diagnostics provided with CXFS, ensure that the `/.rhosts` file on each administration node allows all the nodes in the cluster to have access to each other in order to run remote commands such as `rsh`. The connectivity tests execute a `ping` command from the local node to all nodes and from all nodes to the local node. To execute `ping` on a remote node, CXFS uses `rsh` (user `root`). For example, suppose you have a cluster with three nodes: `cxfs0`, `cxfs1`, and `cxfs2`. The `/.rhosts` file on each administration node will be as follows (prompt denotes node name):

```
cxfs0# cat /.rhosts
cxfs1 root
cxfs1-priv root
cxfs2 root
cxfs2-priv root

cxfs1# cat /.rhosts
cxfs0 root
cxfs0-priv root
cxfs2 root
cxfs2-priv root

cxfs2# cat /.rhosts
cxfs0 root
cxfs0-priv root
cxfs1 root
```

```
cxfs1-priv root
```

Make sure that the mode of the `.rhosts` file is set to `600` (read and write access for the owner only).

After you have completed running the connectivity tests, you may wish to disable `rsh` on all cluster nodes.

Linux CXFS Installation

Caution: CXFS is a complex product. To ensure that it is installed and configured in an optimal manner, you **must** purchase initial setup services from SGI. This chapter is not intended to be used directly by the customer, but is provided for reference.

On SGI ProPack for Linux 3 nodes, CXFS supports either an *administration node* containing the cluster administration daemons (`fs2d`, `crsd`, `cad`, and `cmond`), the CXFS control daemon (`clconfd`), and the cluster database or a *client-only node* containing the `cxfs_client` daemon. The software you install on a node determines the node type.

On ProPack 4 nodes, CXFS supports only a client-only node.

Note: SGI ProPack for Linux is an overlay product that adds or enhances features in the supported Linux base distributions.

Nodes that you intend to run as metadata servers must be installed as administration nodes; all other nodes should be client-only nodes.

This chapter discusses the following:

- "Linux Limitations and Considerations"
- "ProPack 3 Administration Software Installation" on page 86
- "Linux Client-Only Software Installation" on page 89
- "Linux Modifications for CXFS Connectivity Diagnostics" on page 93

After completing these steps, see Chapter 8, "Initial Configuration of the Cluster" on page 125. For details about specific configuration tasks, see Chapter 9, "Reference to GUI Tasks for CXFS" on page 149, and Chapter 10, "Reference to `cmgr` Tasks for CXFS" on page 217.

You should read through this entire book, especially Chapter 17, "Troubleshooting" on page 381, before attempting to install and configure a CXFS cluster. If you are using coexecution with IRIS FailSafe, see the *FailSafe Administrator's Guide for SGI InfiniteStorage*. If you are using a multiOS cluster, see the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

Linux Limitations and Considerations

The following limitations and considerations apply to any Linux node (client-only or administration):

- On Linux systems, the `mkfs.xfs` command does not discover log or realtime subvolumes. You must specify the log or realtime subvolumes on the command line. For more information, see the `mkfs.xfs`(8) man page.
- If you want to use quotas on CXFS filesystems in a cluster with Linux metadata servers, you must use the following options rather than the standard Linux options:
  - `uquota` for user quotas (rather than `usrquota`)
  - `gquota` for group quotas (rather than `grpquota`)

  For example, to enforce both user and group quotas, you could use the following:

  ```
  uquota,gquota
  ```
- GPT partition tables, often created by operating system installers or the `parted` partitioning tool, store labels in two locations. If you reuse a disk that previously had a GPT label, you must be careful; using tools such as `fdisk` to repartition the drive will not eliminate the backup GPT label. When you reboot, EFI scans the disks before the operating system is started. It assumes any backup labels it finds are valid and restores them. This can corrupt or destroy filesystems. You can use the `parted` tool to detect this situation and fix it.
- CXFS filesystems with XFS version 1 directory format cannot be mounted on Linux nodes.
- Whenever you install a new kernel patch, you must also install the corresponding CXFS package. This is required because the kernel patch causes the kernel version number to be increased. Failure to install the corresponding CXFS package will result in the inability to run CXFS. To obtain the required CXFS package, see your SGI support contact.

- After CXFS installation, you must reboot the system before starting any of the cluster administration, CXFS administration, or CXFS client daemons (fs2d, crsd, cad, cmond, clconfd, and cxfs_client). If you start CXFS services without first rebooting the system, the system may panic.
- The implementation of file creates using O_EXCL is not complete. Multiple applications running on the same node using O_EXCL creates as a synchronization mechanism will see the expected behavior (only one of the creates will succeed). However, applications running between nodes will not get the O_EXCL behavior they requested (creates of the same file from two or more separate nodes will all succeed).

CXFS administration Linux nodes have the following limitation: recovery between Altix metadata servers may not complete when using XVM mirrored filesystems.

Client-only Linux nodes have the following limitations and considerations:

- Client-only nodes cannot view or edit user and group quotas. However, user and group quotas are enforced correctly by the metadata server.
- To view or edit your quota information, you must log in to an administration node and make any necessary changes. If you would like to provide a viewing command such as repquota, you could construct shell script similar to the following on the Linux node:

```
#! /bin/sh
#

# Where repquota lives on administration node
repquota=/usr/etc/repquota

# The name of an administration node in the cluster
adminnode=cain

rsh $adminnode "$repquota $*"
exit
```

See also Appendix E, "Filesystem Specifications" on page 485.

ProPack 3 Administration Software Installation

The CXFS administration software will be initially installed and configured by SGI personnel. This section provides an overview of those procedures. You can use the information in this section to verify the installation.

Note: Version numbers shown here are examples; your installed system may differ.

ProPack 3 Administration Installation Overview

Any node that may be a CXFS metadata server must be installed as a CXFS administration node. All other nodes should be client-only nodes.

Note: A ProPack 3 node can be either be a CXFS administration node (for which you install `cluster_admin`) or a client-only node (for which you install `cxfs_client`). You cannot install both `cluster_admin` and `cxfs_client` on the same node. This procedure installs an administration node; to install a client-only node, see "Linux Client-Only Software Installation" on page 89.

Installing the CXFS software for a CXFS administration node requires approximately 65 MB of space.

Do the following to install the software required for a ProPack 3 administration node:

1. Read the CXFS `README` file for the Linux platform to learn about any late-breaking changes in the installation procedure.
2. Install ProPack 3 release, according to the directions in the SGI ProPack documentation. Ensure that you select the `SGI Licensed` package group.
3. Install any required patches. See the SGI ProPack `releasenotes/README` file for more information.
4. Install the CXFS kernel modules:

```
[root@linux CXFS_CDROM]# rpm -ivh cxfs-modules-kernel*
Preparing...                ########################################### [100%]
   1:cxfs-modules           ########################################### [100%]
```

5. Install the CXFS application binaries, documentation, and support tools from the *CXFS 3.4 Altix Server/Client and XVM Plexing for SGI ProPack* CD:

```
[root@linux CXFS_CDROM]# rpm -Uvh xvm-cmds* cluster_admin* cluster_control* \
cluster_services* cxfs_util* cxfs_cluster* cxfs-doc*
Preparing...                ###########################################[100%]
   1:cluster_admin          ########################################### [14%]
   2:cluster_control        ########################################### [29%]
   3:cluster_services       ########################################### [43%]
Started cluster control processes
   4:cxfs_util              ########################################### [57%]
   5:xvm-cmds               ########################################### [71%]
   6:cxfs_cluster           ########################################### [86%]
   7:cxfs-doc               ###########################################[100%]
```

Note: If you have not yet installed the FLEXlm license file, you may get a warning at this point.

6. Remove any remaining `xvm-standalone-module` packages:

```
[root@linux CXFS_CDROM]# rpm -e xvm-standalone-module
```

7. If you want to use quotas on a CXFS filesystem, you must install the quota package:

```
[root@linux cdrom]# rpm -Uvh quota-3.09-1sgi.ia64.rpm
Preparing...########################################### [100%]
   1:quota  ########################################### [100%]
```

8. Install the CXFS graphical user interface (GUI) and XVM GUI packages from the *CXFS 3.4 Altix Server/Client and XVM Plexing for SGI ProPack* CD:

```
[root@linux CXFS_CDROM]# rpm -Uvh sysadm_*
Preparing...                ########################################### [100%]
   1:sysadm_base-lib        ########################################### [  9%]
Adding /usr/lib/sysadm/lib to ld.so.conf
Running ldconfig
   2:sysadm_base-client     ########################################### [ 18%]
   3:sysadm_cluster_base-ser########################################### [ 27%]
Adding /usr/cluster/lib to ld.so.conf
Running ldconfig
```

```
   4:sysadm_cxfs-client     ########################################### [ 36%]
   5:sysadm_cxfs-server     ########################################### [ 45%]
   6:sysadm_cxfs-web        ########################################### [ 55%]
   7:sysadm_xvm-client      ########################################### [ 64%]
   8:sysadm_xvm-server      ########################################### [ 73%]
   9:sysadm_xvm-web         ########################################### [ 82%]
  10:sysadm_base-server     ########################################### [ 91%]
sysadm_base-server rpm post-install: Adding tcpmux file to /etc/xinetd.d
Reloading configuration: [  OK  ]
  11:sysadm_cluster_base-cli########################################### [100%]
Not removing /usr/lib/sysadm/lib from ld.so.conf
```

For more information about XVM, see *XVM Volume Manager Administrator's Guide*.

9. Install the latest Java2 software to support the GUI:

   a. Install the J2SE 1.4.2 (latest patch) SDK for Linux 64-bit from http://java.sun.com.

   b. Do one of the following:

      - (Preferred method) Set the JAVA_HOME environment variable to the java installation directory (such as /usr/lib/j2sdk1.4.2_04) before launching the GUI.

      - Create a symbolic link in /usr/java/ with the following commands so that the GUI can find the new J2SE SDK software. For example, if you installed j2sdk1.4.2_03:

        ```
        [root@linux] root#  cd /usr/java/
        [root@linux ] root # ln -s j2sdk1.4.2_03 j2sdk1.4.2
        ```

10. Use the cxfslicense command to verify that the license is installed in /etc/flexlm/license.dat. If the license is installed properly, there will be no output. If it is not installed, there will be errors.

    For more verbose output, use the -d option. For example, the following shows a properly installed license:

    ```
    [root@linux flexlm]# /usr/cluster/bin/cxfslicense -d
    XVM_STD_IPF license granted.
    XVM_PLEX_IPF license granted.
    CXFS_IPF license granted.
    ```

If there are errors, verify that you have obtained and installed the CXFS licenses. For more information, see "Verifying the Licenses" on page 54.

11. Reboot the system.

⚠ **Caution:** If XVM standalone was in use prior to CXFS installation, you should reboot the system before starting any of the cluster administration daemons, CXFS control daemon, or CXFS client daemon (`fs2d`, `crsd`, `cad`, `cmond`, `clconfd`, and `cxfs_client`) to ensure that the new `xvm` modules are loaded.

Verifying the Linux Administration Installation

To verify that the CXFS software has been installed properly, use the `rpm -q` command to query the packages.

Linux Client-Only Software Installation

The CXFS client-only software will be initially installed and configured by SGI personnel. This section provides an overview of those procedures. You can use the information in this section to verify the installation.

Note: Package version numbers shown here are examples; your installed system may differ.

ProPack 3 Client-Only Installation Overview

Installing the ProPack 3 client requires approximately 60 MB of space, depending upon the packages installed at your site.

To install the required software on a ProPack 3 node, SGI personnel will do the following:

1. Read the release notes to learn about any late-breaking changes in the installation procedure.
2. Install the ProPack 3 release, according to the directions in the SGI ProPack documentation. Ensure that you select the `SGI Licensed` package group.

3. Install any required patches. See the SGI ProPack `releasenotes/README` file for more information.

4. Install the CXFS kernel modules from the *CXFS 3.4 Altix Server/Client and XVM Plexing for SGI ProPack* CD:

```
[root@linux CXFS_CDROM]# rpm -ivh cxfs-modules-kernel*
Preparing...                ########################################### [100%]
   1:cxfs-modules           ########################################### [100%]
```

5. Install the CXFS application binaries, documentation, and support tools from the *CXFS 3.4 Altix Server/Client and XVM Plexing for SGI ProPack* CD:

```
[root@linux CXFS_CDROM]# rpm -Uvh xvm-cmds* cxfs_util* cxfs_client* cxfs-doc*
Preparing...                ###########################################[100%]
   1:cxfs_util              ########################################### [25%]
   2:xvm-cmds               ########################################### [50%]
   3:cxfs_client            ########################################### [75%]
   4:cxfs-doc               ###########################################[100%]
```

6. If you want to use quotas on a CXFS filesystem, you must install the quota package:

```
[root@linux cdrom]# rpm -Uvh quota-3.09-1sgi.ia64.rpm
Preparing...########################################### [100%]
   1:quota  ########################################### [100%]
```

7. Remove any remaining `xvm-standalone-module` packages:

```
[root@linux CXFS_CDROM]# rpm -e xvm-standalone-module
```

8. Use the `cxfslicense` command to verify that the license is installed in `/etc/flexlm/license.dat`. If the license is installed properly, there will be no output. If it is not installed, there will be errors.

   For more verbose output, use the `-d` option. For example, the following shows a properly installed license:

```
[root@linux flexlm]# /usr/cluster/bin/cxfslicense -d
XVM_STD_IPF license granted.
XVM_PLEX_IPF license granted.
CXFS_IPF license granted.
```

If there are errors, verify that you have obtained and installed the CXFS licenses. For more information, see "Verifying the Licenses" on page 54.

9. Reboot the system.

Caution: If XVM standalone was in use prior to CXFS installation, you should reboot the system before starting CXFS services to ensure that the new `xvm` modules are loaded.

ProPack 4 Client-Only Installation Overview

Note: Specific packages listed here are examples and may not match the released product.

Installing the CXFS client CD for ProPack 4 requires approximately 50–200 MB of space, depending upon the packages installed at your site.

To install the required software on a Linux node, SGI personnel will do the following:

1. Read the release notes to learn about any late-breaking changes in the installation procedure.
2. Install the ProPack 4 release, according to the directions in the SGI ProPack documentation. Ensure that you select the `SGI Licensed` package group.
3. Install any required patches. See the SGI ProPack `releasenotes/README` file for more information.
4. Verify that the node is running a supported Linux distribution, according to the CXFS for Linux release notes. For example:

```
[root@linux root]# cat /etc/SuSE-release
SUSE LINUX Enterprise Server 9 (i586)
VERSION = 9
```

Caution: You **must** update the operating system with all security fixes, bug fixes, and enhancements available from the operating system vendor.

5. Insert and mount the *CXFS MultiOS Client 3.4* CD-ROM.
6. Install the CXFS kernel modules:

```
[root@linux cdrom]# rpm -ivh kernel-module-cxfs-kernelrelease-version.ia64.rpm
Preparing...                ########################################### [100%]
   1:kernel-module-cxfs-kern########################################### [100%]
```

Where:

- *kernelrelease* is the kernel release level as output by the `uname -r` command
- *version* is the version number

Note: The *kernelrelease* must match the stock kernel release provided by SUSE.

7. Install the user-space packages:

```
[root@linux cdrom]# rpm -Uvh cxfs_client* cxfs_util* xvm-cmds* cxfs-doc*
Preparing...                ###########################################[100%]
   1:cxfs_util              ########################################### [25%]
   2:xvm-cmds               ########################################### [50%]
   3:cxfs_client            ########################################### [75%]
   4:cxfs-doc               ###########################################[100%]
```

8. Edit the `/etc/cluster/config/cxfs_client.options` file as necessary. See "Client-only Node System Files" on page 101 and the `cxfs_client`(1M) man page.
9. Use the `cxfslicense` command to verify that the license is installed in `/etc/flexlm/license.dat`. If the license is installed properly, there will be no output. If it is not installed, there will be errors.

   For more verbose output, use the `-d` option. For example, the following shows a properly installed license:

   ```
   [root@linux flexlm]# /usr/cluster/bin/cxfslicense -d
   XVM_STD_IPF license granted.
   XVM_PLEX_IPF license granted.
   CXFS_IPF license granted.
   ```

   If there are errors, verify that you have obtained and installed the CXFS licenses. For more information, see "Verifying the Licenses" on page 54.

10. Reboot the system with the newly installed kernel:

```
[root@linux root]# reboot
```

Verifying the Linux Client-Only Installation

To verify that the CXFS software has been installed properly, use the `rpm -q` command to query the packages.

Linux Modifications for CXFS Connectivity Diagnostics

If you want to use the cluster diagnostics to test node connectivity, the `root` user on the node running the CXFS diagnostics must be able to access a remote shell using the `rsh` command (as `root`) on all other nodes in the cluster. There are several ways of accomplishing this, depending on the existing settings in the pluggable authentication modules (PAM) and other security configuration files.

Following is one possible method that works with default settings. Do the following on all administration nodes in the cluster:

1. Install the `rsh-server` RPM.
2. Enable `rsh`.
3. Restart `xinetd`.
4. Add `rsh` to the `/etc/securetty` file.
5. Add the hostname of the node from which you will be running the diagnostics into the `/root/.rhosts` file. Make sure that the mode of the `.rhosts` file is set to `600` (read and write access for the owner only).

After you have completed running the connectivity tests, you may wish to disable `rsh` on all cluster nodes.

For more information, see the Red Hat documentation and the `hosts.equiv` man page.

Postinstallation Steps

This chapter discusses the following:

- "Configuring System Files"
- "Testing the System" on page 101
- "Manual CXFS Start/Stop" on page 104
- "Rolling Upgrades" on page 106
- "IRIX: Configuring for Automatic Restart" on page 109
- "IRIX: Converting Filesystem Definitions for Upgrades" on page 109
- "Linux: Using `cxfs-reprobe` on Client-Only Nodes" on page 110

After completing these step discussed in this chapter, see Chapter 8, "Initial Configuration of the Cluster" on page 125. For details about specific configuration tasks, see Chapter 9, "Reference to GUI Tasks for CXFS" on page 149, and Chapter 10, "Reference to `cmgr` Tasks for CXFS" on page 217. For information about installing CXFS and Trusted IRIX, see Chapter 14, "Trusted IRIX and CXFS" on page 357. For information about upgrades, see "Rolling Upgrades" on page 106.

Configuring System Files

When you install the CXFS software, there are some system file considerations you must take into account. **The network configuration is critical.** Each node in the cluster must be able to communicate with every other node in the cluster by both logical name and IP address without going through any other network routing; proper name resolution is key. SGI recommends static routing.

`/etc/exports` on All Nodes

The optional `/etc/exports` file on each node describes the filesystems that are being exported to NFS clients.

If the `/etc/exports` file contains a CXFS mount point, then when the system is booted NFS will export the empty mount point because the exports are done before

CXFS is running. When CXFS on the node joins membership and starts mounting filesystems, the `clconfd-pre-mount` script searches the `/etc/exports` file looking for the mountpoint that is being mounted. If found, the script unexports the mountpoint directory because if it did not the CXFS mount would fail. After successfully mounting the filesystem, the `clconfd-post-mount` script will search the `/etc/exports` file and export the mount point if it is found in the `/etc/exports` file.

For more information, see "CXFS Mount Scripts" on page 293.

Administration Node System Files

This section discusses system files on administration nodes.

`/etc/services` on CXFS Administration Nodes

Edit the `/etc/services` file on each CXFS administration node so that it contains entries for `sgi-cad` and `sgi-crsd` before you install the `cluster_admin` product on each CXFS administration node in the pool. The port numbers assigned for these processes must be the same in all nodes in the pool.

Note: You will see an `inst` message that says `sgi-cmsd` and `sgi-gcd` must be added to `/etc/services`. This is true only for coexecution with FailSafe, or when running only FailSafe; if you are running just CXFS, you do not need `sgi-cmsd`. CXFS does not require `sgi-cmsd`.

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

```
sgi-crsd        7500/udp          # Cluster reset services daemon
sgi-cad         9000/tcp          # Cluster Admin daemon
```

`cad.options` on CXFS Administration Nodes

The `cad.options` file on each CXFS administration node contains the list of parameters that the cluster administration daemon reads when the `cad` process is started. The files are located as follows:

- IRIX: `/etc/config/cad.options`
- Linux: `/etc/cluster/config/cad.options`

`cad` provides cluster information.

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

`--append_log`	Append `cad` logging information to the `cad` log file instead of overwriting it.
`--log_file` *filename*	`cad` log filename. Alternately, this can be specified as `-lf` *filename.*
`-vvvv`	Verbosity level. The number of `v` characters indicates the level of logging. Setting `-v` logs the fewest messages; setting `-vvvv` logs the highest number of messages.

The default file has the following options:

```
-lf /var/cluster/ha/log/cad_log --append_log
```

The following example shows an `/etc/config/cad.options` file that uses a medium-level of verbosity:

```
-vv -lf /var/cluster/ha/log/cad_nodename --append_log
```

The default log file is `/var/cluster/ha/log/cad_log`. Error and warning messages are appended to the log file if log file is already present.

The contents of the `/etc/config/cad.options` file cannot be modified using the `cmgr` command or the GUI.

If you make a change to the `cad.options` file at any time other than initial configuration, you must restart the `cad` processes in order for these changes to take effect. You can do this by rebooting the nodes or by entering the following command:

- IRIX:

  ```
  # /etc/init.d/cluster restart
  ```

- Linux:

  ```
  # /etc/init.d/cxfs_cluster restart
  ```

If you execute this command on a running cluster, it will remain up and running. However, the GUI will lose connection with the `cad` daemon; the GUI will prompt you to reconnect.

`fs2d.options` on CXFS Administration Nodes

The `fs2d.options` file on each CXFS administration node contains the list of parameters that the `fs2d` daemon reads when the process is started. (The `fs2d` daemon manages the distribution of the cluster database (CDB) across the CXFS administration nodes in the pool.) The files are located as follows:

- IRIX: `/etc/config/fs2d.options`
- Linux: `/etc/cluster/config/fs2d.options`

Table 6-1 shows the options can that can be set in the `fs2d.options` file.

Table 6-1 `fs2d.options` File Options

Option	Description
`-logevents` *event name*	Log selected events. The following event names may be used: `all`, `internal`, `args`, `attach`, `chandle`, `node`, `tree`, `lock`, `datacon`, `trap`, `notify`, `access`, `storage`. The default is `all`.
`-logdest` *log destination*	Set log destination. The following log destinations may be used: `all`, `stdout`, `stderr`, `syslog`, `logfile`. If multiple destinations are specified, the log messages are written to all of them. If `logfile` is specified, it has no effect unless the `-logfile` option is also specified. The default is `logfile`.
`-logfile` *filename*	Set log filename. The default is `/var/cluster/ha/log/fs2d_log`.
`-logfilemax` *maximum size*	Set log file maximum size (in bytes). If the file exceeds the maximum size, any preexisting `filename.old` will be deleted, the current file will be renamed to `filename.old`, and a new file will be created. A single message will not be split across files. If `-logfile` is set, the default is `10000000`.
`-loglevel` *loglevel*	Set log level. The following log levels may be used: `always`, `critical`, `error`, `warning`, `info`, `moreinfo`, `freq`, `morefreq`, `trace`, `busy`. The default is `info`.

Option	Description
`-trace` *trace_class*	Trace selected events. The following trace classes may be used: `all`, `rpcs`, `updates`, `transactions`, `monitor`. If you specify this option, you must also specify `-tracefile` and/or `-tracelog`. No tracing is done, even if it is requested for one or more classes of events, unless either or both of `-tracefile` or `-tracelog` is specified. The default is `transactions`.
`-tracefile` *filename*	Set trace filename. There is no default.
`-tracefilemax` *maximum_size*	Set trace file maximum size (in bytes). If the file exceeds the maximum size, any preexisting `filename.old` will be deleted, the current file will be renamed to `filename.old`, and a new file will be created.
`-[no]tracelog`	[Do not] trace to log destination. When this option is set, tracing messages are directed to the log destination or destinations. If there is also a trace file, the tracing messages are written there as well. The default is `-tracelog.`
`-[no]parent_timer`	[Do not] exit when the parent exits. The default is `-noparent_timer`.
`-[no]daemonize`	[Do not] run as a daemon. The default is `-daemonize`.
`-l`	Do not run as a daemon.
`-h`	Print usage message.
`-o help`	Print usage message.

If you use the default values for these options, the system will be configured so that all log messages of level `info` or less, and all trace messages for transaction events, are sent to the `/var/cluster/ha/log/fs2d_log` file. When the file size reaches 10 MB, this file will be moved to its namesake with the `.old` extension and logging will roll over to a new file of the same name. A single message will not be split across files.

If you make a change to the `fs2d.options` file at any time other than the initial configuration time, you must restart the `fs2d` processes in order for those changes to take effect. You can do this by rebooting the CXFS administration nodes or by entering the following command:

- IRIX:

```
# /etc/init.d/cluster restart
```

- Linux:

  ```
  # /etc/init.d/cxfs_cluster restart
  ```

If you execute this command on a running cluster, it should remain up and running. However, the GUI will lose connection with the `cad` daemon; the GUI will prompt you to reconnect.

Example 1

The following example shows an `/etc/config/fs2d.options` file that directs logging and tracing information as follows:

- All log events are sent to:
  - IRIX: `/var/adm/SYSLOG`
  - Linux: `/var/log/messages`
- Tracing information for RPCs, updates, and transactions are sent to `/var/cluster/ha/log/fs2d_ops1`.

  When the size of this file exceeds 100,000,000 bytes, this file is renamed to `/var/cluster/ha/log/fs2d_ops1.old` and a new file `/var/cluster/ha/log/fs2d_ops1` is created. A single message is not split across files.

(Line breaks added for readability.)

```
-logevents all -loglevel trace -logdest syslog -trace rpcs
-trace updates -trace transactions -tracefile /var/cluster/ha/log/fs2d_ops1
-tracefilemax 100000000
```

Example 2

The following example shows an `/etc/config/fs2d.options` file that directs all log and trace messages into one file, `/var/cluster/ha/log/fs2d_chaos6`, for which a maximum size of 100,000,000 bytes is specified. `-tracelog` directs the tracing to the log file.

(Line breaks added for readability.)

```
-logevents all -loglevel trace -trace rpcs -trace updates
-trace transactions -tracelog -logfile /var/cluster/ha/log/fs2d_chaos6
-logfilemax 100000000 -logdest logfile.
```

Client-only Node System Files

This section discusses the `cxfs_client.options` file for IRIX and Linux client-only nodes. For client-only nodes running other operating systems, see the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

On client-only nodes, you can modify the CXFS client daemon (`/usr/cluster/bin/cxfs_client`) by placing options in the `cxfs_client.options` file:

- IRIX: `/etc/config/cxfs_client.options`
- Linux: `/etc/cluster/config/cxfs_client.options`

The available options are documented in the `cxfs_client` man page.

Caution: Some of the options are intended to be used internally by SGI only for testing purposes and do not represent supported configurations. Consult your SGI service representative before making any changes.

The first line in the `cxfs_client.options` file must contain the options you want `cxfs_client` to process; you cannot include a comment as the first line.

For example, to see if `cxfs_client` is using the options in `cxfs_client.options`, enter the following:

```
irix# ps -ax | grep cxfs_client
 3612 ?        S      0:00 /usr/cluster/bin/cxfs_client -i cxfs3-5
 3841 pts/0    S      0:00 grep cxfs_client
```

Testing the System

This section discusses the following:

- "Private Network Interface"
- "System Reset Connection for CXFS Administration Nodes" on page 102

Private Network Interface

For each private network on each node in the pool, enter the following, where *nodeIPaddress* is the IP address of the node:

```
# ping -c 3 nodeIPaddress
```

Typical `ping` output should appear, such as the following:

```
PING IPaddress (190.x.x.x: 56 data bytes
64 bytes from 190.x.x.x: icmp_seq=0 tt1=254 time=3 ms
64 bytes from 190.x.x.x: icmp_seq=1 tt1=254 time=2 ms
64 bytes from 190.x.x.x: icmp_seq=2 tt1=254 time=2 ms
```

If `ping` fails, follow these steps:

1. Verify that the network interface was configured up by using `ifconfig`. For example:

```
# ifconfig ec3
ec3: flags=c63<UP,BROADCAST,NOTRAILERS,RUNNING,FILTMULTI,MULTICAST>
inet 190.x.x.x netmask 0xffffff00 broadcast 190.x.x.x
```

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

2. Verify that the cables are correctly seated.

Repeat this procedure on each node.

System Reset Connection for CXFS Administration Nodes

To test the system reset connections, do the following:

1. Ensure that the nodes and the serial port multiplexer are powered on.
2. Start the `cmgr` command on one of the CXFS administration nodes in the pool:

   ```
   # cmgr
   ```

3. Stop CXFS services on the entire cluster:

   ```
   stop cx_services for cluster clustername
   ```

For example:

```
cmgr> stop cx_services for cluster cxfs6-8
```

Wait until the node has successfully transitioned to inactive state and the CXFS processes have exited. This process can take a few minutes.

4. Test the serial connections by entering one of the following:

   - To test the whole cluster, enter the following:

     `test serial in cluster` *clustername*

     For example:

```
cmgr> test serial in cluster cxfs6-8
Status: Testing serial lines ...
Status: Checking serial lines using crsd (cluster reset services) from node cxfs8
Success: Serial ping command OK.

Status: Checking serial lines using crsd (cluster reset services) from node cxfs6
Success: Serial ping command OK.

Status: Checking serial lines using crsd (cluster reset services) from node cxfs7
Success: Serial ping command OK.

Notice: overall exit status:success, tests failed:0, total tests executed:1
```

   - To test an individual node, enter the following:

     `test serial in cluster` *clustername* `node` *machinename*

     For example:

```
cmgr> test serial in cluster cxfs6-8 node cxfs7
Status: Testing serial lines ...
Status: Checking serial lines using crsd (cluster reset services) from node cxfs6
Success: Serial ping command OK.

Notice: overall exit status:success, tests failed:0, total tests executed:1
```

   - To test an individual node using just a `ping`, enter the following:

     `admin ping node` *nodename*

For example:

```
cmgr> admin ping node cxfs7

ping operation successful
```

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

6. Repeat the process on other nodes in the cluster.

Manual CXFS Start/Stop

On administration nodes, the `/etc/init.d/cluster` (IRIX) or `/etc/init.d/cxfs_cluster` (Linux) script will be invoked automatically during normal system startup and shutdown procedures; on client-only nodes, the script is `/etc/init.d/cxfs_client`. This script starts and stops the processes required to run CXFS.

To start up CXFS processes manually, enter the following commands:

- On an administration node:
  - IRIX:

    ```
    # /etc/init.d/cluster start
    Starting cluster services: fs2d cmond cad crsd
    # /etc/init.d/cxfs start
    Starting CXFS Cluster services:
    Starting clconfd:
    ```

  - Linux:

    ```
    # /etc/init.d/cxfs_cluster start
    Starting cluster services: fs2d cmond cad crsd             [  OK  ]
    # /etc/init.d/cxfs start
    ```

- On an IRIX client-only node:

  ```
  # /etc/init.d/cxfs_client start
  cxfs_client daemon started
  ```

- On a Linux client-only node:

  ```
  # /etc/init.d/cxfs_client start
  Loading cxfs modules:                                       [  OK  ]
  Mounting devfs filesystems:                                 [  OK  ]
  Starting cxfs client:                                       [  OK  ]
  ```

To stop CXFS processes manually , enter the following command:

- On an administration node:
  - IRIX:

    ```
    # /etc/init.d/cxfs stop
    # /etc/init.d/cluster stop
    ```

  - Linux:

    ```
    # /etc/init.d/cxfs stop
    # /etc/init.d/cxfs_cluster stop
    ```

- On an IRIX client-only node:

  ```
  # /etc/init.d/cxfs_client stop
  Shutting down CXFS client
  ```

- On a Linux client-only node:

  ```
  # /etc/init.d/cxfs_client stop
  Stopping cxfs client:                                       [  OK  ]
  ```

Note: There is also a `restart` option that performs a stop and then a start.

To see the current status of the CXFS processes, use the `status` argument. For example, the following output shows that `cxfs_client` is running:

```
# /etc/init.d/cxfs_client status
cxfs_client (pid 3226) is running...
```

The output in the following example shows that the CXFS client is stopped on a client-only node:

```
# /etc/init.d/cxfs_client status
cxfs_client is stopped
```

Rolling Upgrades

Beginning with CXFS 3.2, SGI supports a policy for CXFS that permits a rolling upgrade of a subset of nodes from 3.2 to 3.*anything*. This policy lets you to keep your cluster running and filesystems available during the upgrade process. For example, you could upgrade from 3.2 to 3.3, 3.4.*X*, or 3.6.

Note: When performing upgrades, you should not make any other configuration changes to the cluster (such as adding new nodes or filesystems) until the upgrade of all nodes is complete and the cluster is running normally.

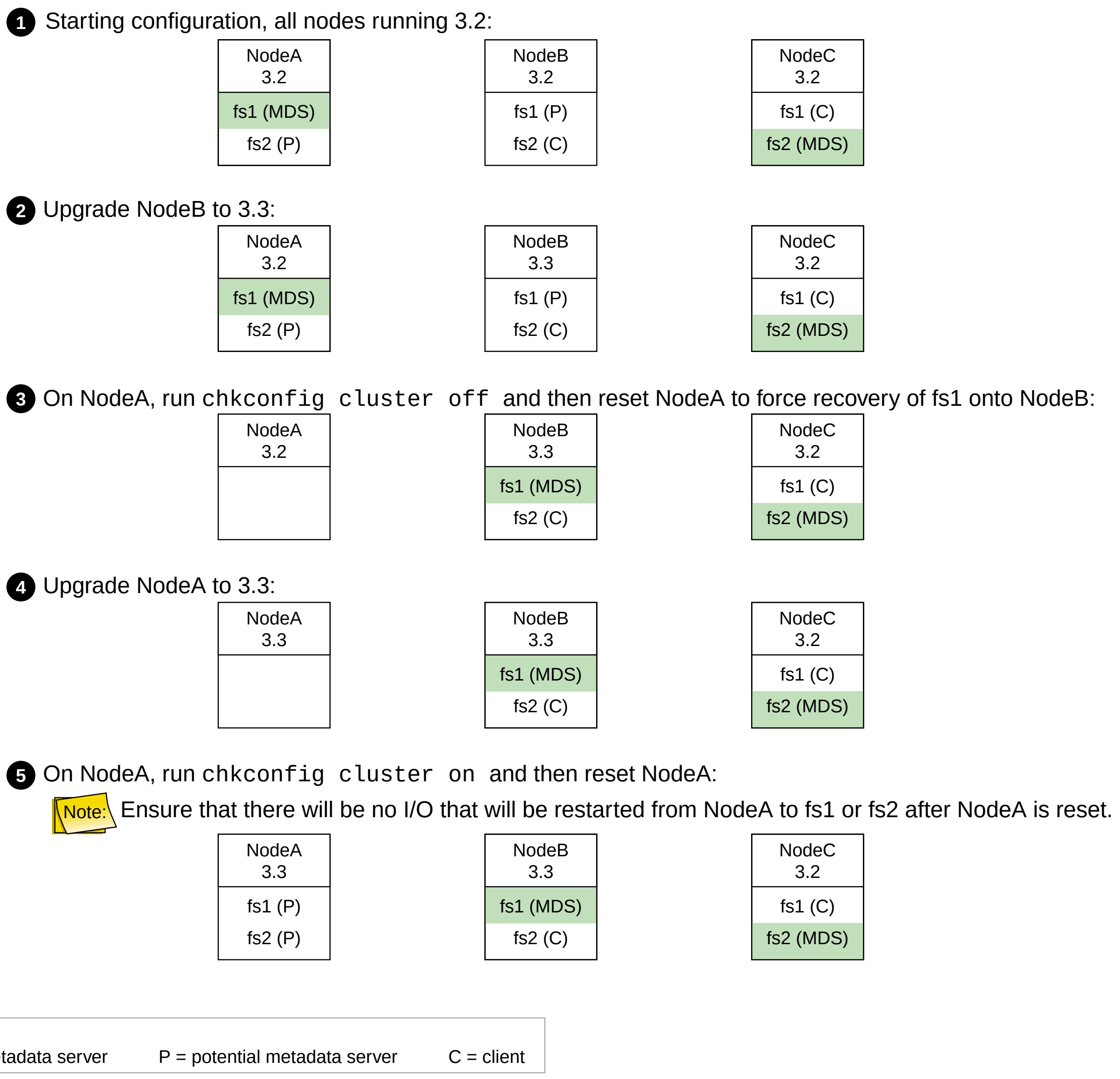
The upgrade procedure makes use of a *standby node*, which is a server-capable administration node that is configured as a potential metadata server for a given filesystem, but does not currently run any applications that will use that filesystem.

Note: If you are running multiple CXFS releases and run into problems, you may have to bring all administration nodes to a single release before the problem can be addressed.

After the upgrade process is complete, all nodes in a production cluster should be running the same major-level release, such as 3.4, or any minor-level release with the same major level (3.4.*anything*). SGI strongly recommends that all server-capable nodes run the same minor-level release as well (such as 3.4.3). For example, a production cluster could contain server-capable nodes running 3.4.1 and client-only nodes running 3.4, 3.4.1, and 3.4.5; however, it should not contain any nodes running 3.3.

Each CXFS release is paired with a given operating system release. For more information, see the product release notes.

The following figures show an example upgrade procedure for a three-node cluster with two filesystems (`fs1` and `fs2`), in which all nodes are running CXFS 3.2 at the beginning.


1 Starting configuration, all nodes running 3.2:

2 Upgrade NodeB to 3.3:

3 On NodeA, run `chkconfig cluster off` and then reset NodeA to force recovery of fs1 onto NodeB:

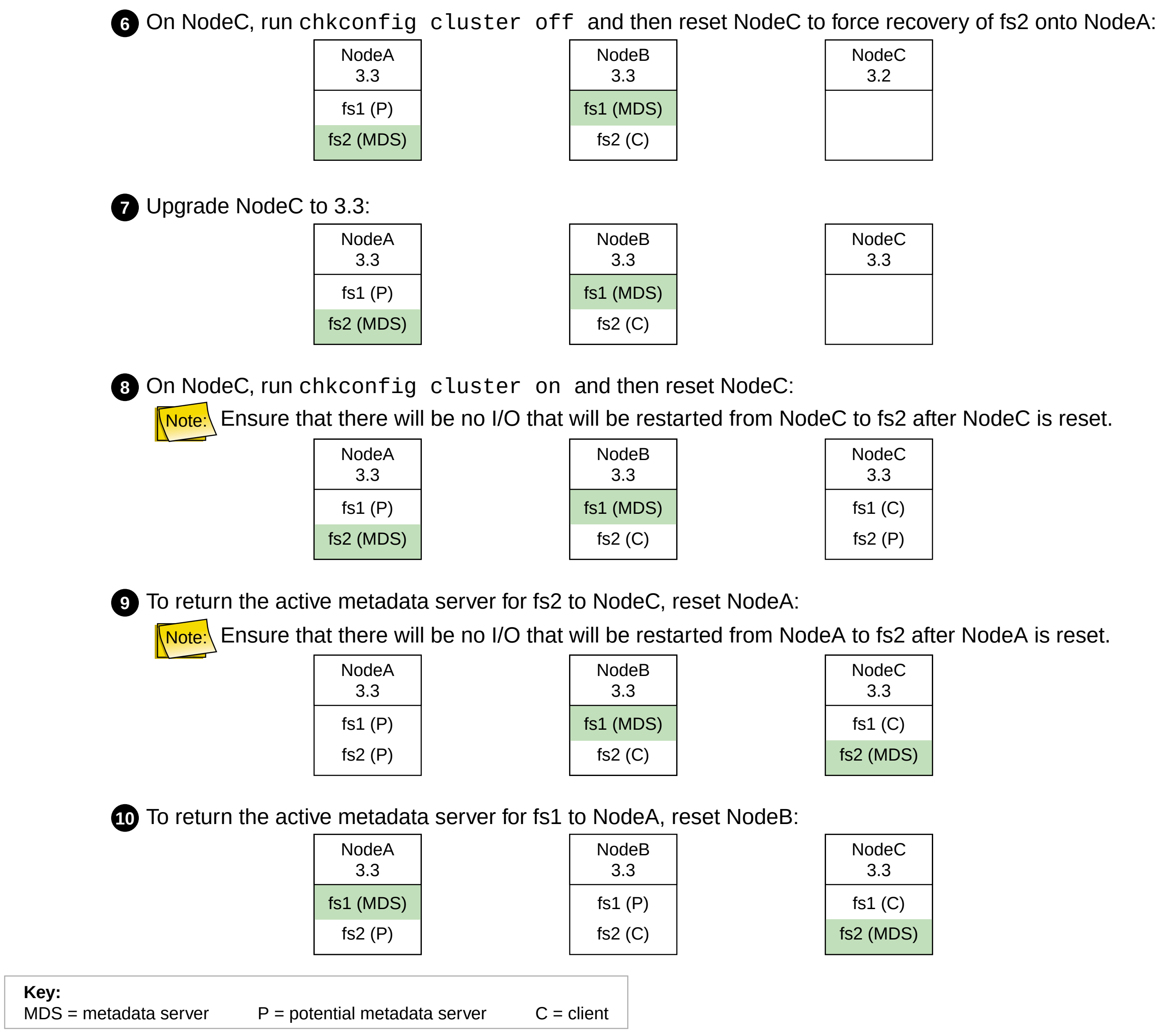
4 Upgrade NodeA to 3.3:

5 On NodeA, run `chkconfig cluster on` and then reset NodeA:

Note: Ensure that there will be no I/O that will be restarted from NodeA to fs1 or fs2 after NodeA is reset.

Key:
MDS = metadata server
P = potential metadata server
C = client

Figure 6-1 Example Rolling Upgrade Procedure (steps 1-5)



6 On NodeC, run `chkconfig cluster off` and then reset NodeC to force recovery of fs2 onto NodeA:

7 Upgrade NodeC to 3.3:

8 On NodeC, run `chkconfig cluster on` and then reset NodeC:

Note: Ensure that there will be no I/O that will be restarted from NodeC to fs2 after NodeC is reset.

9 To return the active metadata server for fs2 to NodeC, reset NodeA:

Note: Ensure that there will be no I/O that will be restarted from NodeA to fs2 after NodeA is reset.

10 To return the active metadata server for fs1 to NodeA, reset NodeB:

Key:
MDS = metadata server P = potential metadata server C = client

NodeA 3.3 | fs1 (P) | fs2 (MDS)
NodeB 3.3 | fs1 (MDS) | fs2 (C)
NodeC 3.2
NodeA 3.3 | fs1 (P) | fs2 (MDS)
NodeB 3.3 | fs1 (MDS) | fs2 (C)
NodeC 3.3
NodeA 3.3 | fs1 (P) | fs2 (MDS)
NodeB 3.3 | fs1 (MDS) | fs2 (C)
NodeC 3.3 | fs1 (C) | fs2 (P)
NodeA 3.3 | fs1 (P) | fs2 (P)
NodeB 3.3 | fs1 (MDS) | fs2 (C)
NodeC 3.3 | fs1 (C) | fs2 (MDS)
NodeA 3.3 | fs1 (MDS) | fs2 (P)
NodeB 3.3 | fs1 (P) | fs2 (C)
NodeC 3.3 | fs1 (C) | fs2 (MDS)

Figure 6-2 Example Rolling Upgrade Procedure (steps 6-10)

IRIX: Configuring for Automatic Restart

If you want nodes to restart automatically when they are reset or when the node is powered on, you must set the boot parameter `AutoLoad` variable on each IRIX node to `yes` as follows:

```
# nvram AutoLoad yes
```

This setting is recommended, but is not required for CXFS.

You can check the setting of this variable with the following command:

```
# nvram AutoLoad
```

IRIX: Converting Filesystem Definitions for Upgrades

The structure of the CXFS filesystem configuration was changed with the release of IRIX 6.5.13f. Upgrading to the 6.5.13f release provided an automatic conversion from the old structure to the new structure. However, if you are upgrading directly from 6.5.12f or earlier, (without first installing and running 6.5.13f), you must convert your CXFS filesystem definitions manually.

Upgrading from 6.5.12f or Earlier

Note: If you are upgrading from 6.5.13f or later, you do not need to follow the instructions in this section. Filesystems definitions are automatically and transparently converted when running 6.5.13f.

After upgrading from 6.5.12f or earlier, you will notice that the CXFS filesystems are no longer mounted, and that they do not appear in the GUI or `cmgr` queries. To convert all of the old CXFS filesystem definitions to the new format, simply run the following command from one of the 6.5.14f or later nodes in the CXFS cluster:

```
 # /usr/sysadm/privbin/cxfsfilesystemUpgrade
```

After running this command, the CXFS filesystems should appear in the GUI and `cmgr` output, and they should be mounted if their status was `enabled` and CXFS services are active.

Caution: This conversion is a one-time operation and **should not** be run a second time. If you make changes to the filesystem and then run `cxfsfilesystemUpgrade` for a second time, all of your changes will be lost.

Running with All IRIX Nodes Upgraded to 6.5.14f or Later

After all of the IRIX nodes in the cluster have been upgraded to 6.5.14f or later, it is recommended that you destroy the old CXFS filesystem definitions, in order to prevent these stale definitions from overwriting the new definitions if the `cxfsfilesystemUpgrade` command were to be run again accidentally. To destroy the old CXFS filesystem definitions, enter the following:

```
# /usr/cluster/bin/cdbutil -c "delete #cluster#clustername#Cellular#FileSystems"
```

Linux: Using `cxfs-reprobe` on Client-Only Nodes

When `cxfs_client` needs to rescan disk buses, it executes the `/var/cluster/cxfs_client-scripts/cxfs-reprobe` script. This requires the use of parameters in Linux due to limitations in the Linux SCSI layer. You can export these parameters from the `/etc/cluster/config/cxfs_client.options` file.

The script detects the presence of the SCSI and/or XSCSI layers on the system and defaults to probing whichever layers are detected. This decision can be overridden by setting `CXFS_PROBE_SCSI` and/or `CXFS_PROBE_XSCSI` to either `0` (to disable the probe) or `1` (to force the probe) on the appropriate bus.

When an XSCSI scan is performed, all buses are scanned by default. This can be overridden by specifying a space-separated list of buses in `CXFS_PROBE_XSCSI_BUSES`. (If you include space, you must enclose the list within single quotation marks.) For example:

```
export CXFS_PROBE_XSCSI_BUSES='/dev/xscsi/pci01.03.0-1/bus /dev/xscsi/pci02.01.0-2/bus'
```

When a SCSI scan is performed, a fixed range of buses/channels/IDs and LUNs are scanned; these ranges may need to be changed to ensure that all devices are found. The ranges can also be reduced to increase scanning speed if a smaller space is sufficient.

The following summarizes the environment variables (separate multiple values by white space and enclose withing single quotation marks):

`CXFS_PROBE_SCSI=`*0|1*

Stops (`0`) or forces (`1`) a SCSI probe. Default: `1` if SCSI

`CXFS_PROBE_SCSI_BUSES=`*BusList*

Scans the buses listed. Default: `0 1 2`

`CXFS_PROBE_SCSI_CHANNELS=`*ChannelList*

Scans the channels listed. Default: `0`

`CXFS_PROBE_SCSI_IDS=`*IDList*

Scans the IDS listed. Default: `0 1 2 3`

`CXFS_PROBE_SCSI_LUNS=`*LunList*

Scans the LUNs listed. Default: `0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15`

`CXFS_PROBE_XSCSI=`*0|1*

Stops (`1`) or forces (`1`) an XSCSI probe. Default: `1` if XSCSI

`CXFS_PROBE_XSCSI_BUSES=`*BusList*

Scans the buses listed. Default: all XSCSI buses

For example, the following would only scan the first two SCSI buses:

```
export CXFS_PROBE_SCSI_BUSES='0 1'
```

The following would scan 16 LUNs on each bus, channel, and ID combination (all on one line):

```
export CXFS_PROBE_SCSI_LUNS='0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15'
```

Other options within the `/etc/cluster/config/cxfs_client.options` file begin with a `-` character. Following is an example `cxfs_client.options` file:

```
# Example cxfs_client.options file
#
-Dnormal -serror
export CXFS_PROBE_SCSI_BUSSES=1
export CXFS_PROBE_SCSI_LUNS='0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20'
```

Note: The `-` character or the term `export` must start in the first position of each line in the `cxfs_client.options` file; otherwise, they are ignored by the `/etc/init.d/cxfs_client` script.

Best Practices

This chapter summarizes configuration and administration best-practices information for CXFS.

For the latest information and a matrix of supported CXFS and operating system software, see http://support.sgi.com/content_request/838562/index.html on Supportfolio.

Configuration Best Practices

This section discusses the following configuration topics:

- "Fix Network Issues First" on page 114
- "Use a Private Network" on page 114
- "Provide Enough Memory" on page 114
- "Do Not Mix Metadata Operating System Flavors" on page 115
- "Use the Correct Mix of Software Releases" on page 115
- "Form a Small Functional Cluster First" on page 115
- "Choose a Metadata Server that is Dedicated to CXFS Work" on page 116
- "Use an Odd Number of Server-Capable Nodes" on page 116
- "Make Most Nodes Client-Only" on page 116
- "Use a Client-Only Tiebreaker" on page 116
- "Protect Data Integrity on All Nodes" on page 117
- "Configure Filesystems Properly" on page 120
- "Verify the Configuration" on page 121
- "Starting the Cluster" on page 121

Fix Network Issues First

If there are any network issues on the private network, fix them before trying to use CXFS. Ensure that you understand the information in "Hostname Resolution and Network Configuration Rules" on page 57.

Use a Private Network

You must use a private network for CXFS metadata traffic:

- A private network is a requirement.
- The private network is used for metadata traffic and should not be used for other kinds of traffic.
- A stable private network is important for a stable CXFS cluster environment.
- Two or more clusters should not share the same private network. A separate private network switch is required for each cluster.
- The private network should contain at least a 100-Mbit network switch. A network hub is not supported and should not be used.
- All cluster nodes should be on the same physical network segment (that is, no routers between hosts and the switch).
- The private network must be configured as the highest priority network for the cluster. The public network may be configured as a lower priority network to be used by CXFS network failover in case of a failure in the private network.
- A virtual local area network (VLAN) is not supported for a private network.
- Use private (10.*x.x.x*, 176.16.*x.x*, or 192.168.*x.x*) network addresses (RFC 1918).
- When administering more than one CXFS cluster, use unique private network addresses for each cluster.

Provide Enough Memory

There should be at least 2 GB of RAM on the system. A metadata server must have at least 1 processor and 1 GB of memory more that what it would need for its normal workload (work other than CXFS). In general, this means that the minimum configuration would be 2 processors and 2 GB of memory. If the metadata server is also doing NFS or Samba serving, then more memory is recommended (and the `nbuf` and `ncsize` kernel parameters should be increased from their defaults). CXFS makes heavy use of memory for caching.

If a very large number of files (tens of thousands) are expected to be open at any one time, additional memory over the minimum is recommended. Use the following general rule to determine the amount of memory required when the number of open files at any one time may be this large:

`2 KB  x` *#inodes* `=` *metadata_server_memory*

To avoid problems during metadata server recovery/relocation, all potential metadata servers should have as much memory as the active metadata server

Do Not Mix Metadata Operating System Flavors

Mixing Linux and IRIX metadata servers in one cluster is not supported. All server-capable administration nodes in a cluster must be either all Linux or all IRIX.

Use the Correct Mix of Software Releases

Create a new cluster using server-capable nodes that have the same version of the OS release and the same version of CXFS installed.

All nodes should run the same level of CXFS and the same level of operating system software, according to platform type. To support upgrading without having to take the whole cluster down, nodes can run different CXFS releases during the upgrade process.

For details, see the platform-specific release notes and "Rolling Upgrades" on page 106.

Form a Small Functional Cluster First

For large clusters, SGI recommends that you first form a functional cluster with just server-capable nodes and then build up the large cluster in small groups of client-only nodes. This method make it easier to locate and fix problems, should any occur. See "Configuring a Large Cluster" on page 145.

Choose a Metadata Server that is Dedicated to CXFS Work

The nodes you choose as potential metadata servers should be dedicated to CXFS and filesystems work. You should not choose a graphics system or one that is running other applications that could cause conflicts with CXFS. Therefore, you should not use Fuel, Onyx, Octane, or Prism systems as metadata servers.

SGI recommends that all potential metadata servers be configured with system reset in order to to protect data integrity. See "Protect Data Integrity on All Nodes" on page 117.

Use an Odd Number of Server-Capable Nodes

Use an odd number of server-capable nodes. If you have an even number of server-capable administration nodes, define a CXFS tiebreaker node. See "Use a Client-Only Tiebreaker" on page 116.

Make Most Nodes Client-Only

You should define most nodes as client-only nodes and define just the nodes that may be used for CXFS metadata as server-capable administration nodes. Use of client administration nodes should be an exception, such as for a Failsafe co-execution node that cannot be a metadata server (Failsafe requires that a node be either a server-capable administration node or a client administration node).

The advantage to using client-only nodes is that they do not keep a copy of the cluster database; they contact an administration node to get configuration information. It is easier and faster to keep the database synchronized on a small set of nodes, rather than on every node in the cluster. In addition, if there are issues, there will be a smaller set of nodes on which you must look for problem.

Use a Client-Only Tiebreaker

SGI recommends that you always define a client-only node CXFS tiebreaker. (Server-capable nodes are not recommended as tiebreaker nodes.) This is most important when there are an even number of server-capable administration nodes.

The tiebreaker is of benefit in a cluster with an odd number of server-capable admin nodes when one of the server-capable administration nodes is removed from the cluster (via a stop of CXFS services) for maintenance.

The following rules apply:

- If exactly two server-capable nodes are configured and there are no client-only nodes, **neither** server-capable node should be set as the tiebreaker. (If one node was set as the tiebreaker and it failed, the other node would also shut down.)
- If exactly two server-capable nodes are configured and there is at least one client-only node, you should specify the client-only node as a tiebreaker.

  If one of the server-capable nodes is the CXFS tiebreaker in a two server-capable cluster, failure of that node or stopping the CXFS services on that node will result in a cluster-wide forced shutdown. Therefore SGI recommends that you use client-only nodes as tiebreakers so that either server could fail but the cluster would remain operational via the other server.

  Setting a client-only node as the tiebreaker is recommended. This avoids the problem of multiple-clusters being formed (also known as *split-brain syndrome*) while still allowing the cluster to continue if the one if the metadata servers fails.
- Setting a server-capable node as tiebreaker is recommended when there are four or more server-capable nodes and no client-only nodes.
- If there are an even number of servers and there is no tiebreaker set, the failure action hierarchy should not contain the `shutdown` option because there is no notification that a shutdown has occurred. See "Isolating Failed Nodes" on page 27.

SGI recommends that you start CXFS services on the tie-breaker client after the metadata servers are all up and running, and before CXFS services are started on any other clients.

Protect Data Integrity on All Nodes

All nodes must protect data integrity in case of failure. Configured system reset or I/O fencing is required to ensure data integrity for all nodes.

Note: No matter what the cluster components are, SGI recommends that you use a system reset configuration on potential metadata servers to protect data integrity and improve server reliability. I/O fencing (or system reset when available) must be used on client-only nodes.

System Reset

You should configure system reset for any potential metadata servers in order to protect data integrity. (I/O fencing is appropriate for client-only nodes.) This means that nodes without system reset capability, such as Fuel systems, should not be potential metadata servers.

System reset is recommended because if a server hangs, it must be rebooted as quickly as possbile to get it back in service, which is not available with I/O fencing. In addition, data corruption is more likely to occur with a rouge metadata server, not a rouge client. (If fencing were to be used on a metadata server and fail, the cluster would have to either shutdown or hang. A fencing failure can occur if an administrator is logged into the switch.)

I/O Fencing

Nodes without system reset capability (such as Fuel, Octane, AIX, Linux 32-bit, Mac OS X, Solaris, and Windows nodes) require I/O fencing. I/O fencing is also appropriate for nodes with system controllers if they are client-only nodes.

You should use the `admin` account when configuring I/O fencing. On a Brocade switch running 4.*x.x.x* or later firmware, modify the `admin` account to restrict it to a single `telnet` session. For details, see the release notes.

If you use I/O fencing, you must keep the `telnet` port on the switch free at all times; **do not** perform a `telnet` to the switch and leave the session connected.

If you use I/O fencing, SGI recommends that you use a switched network of at least 100baseT.

You should isolate the power supply for the switch from the power supply for a node and its system controller. You should avoid any possible situation in which a node can continue running while both the switch and the system controller lose power. Avoiding this situation will prevent the possibility a split-brain scenario.

You must put switches used for I/O fencing on a network other than the primary CXFS private network so that problems on the CXFS private network can be dealt with by the fencing process and thereby avoid data corruption issues. The network to which the switch is connected must be accessible by all administration nodes in the cluster.

For details, see the release notes.

Avoid Network Partition

The worst scenario is one in which the node does not detect the loss of communication but still allows access to the shared disks, leading to data corruption. For example, it is possible that one node in the cluster could be unable to communicate with other nodes in the cluster (due to a software or hardware failure) but still be able to access shared disks, despite the fact that the cluster does not see this node as an active member.

In this case, the reset will allow one of the other nodes to forcibly prevent the failing node from accessing the disk at the instant the error is detected and prior to recovery from the node's departure from the cluster, ensuring no further activity from this node.

In a case of a true network partition, where an existing CXFS kernel membership splits into two halves (each with half the total number of server-capable nodes), the following will happen:

- If the CXFS tiebreaker and system reset or I/O fencing are configured, the half with the tiebreaker node will reset or fence the other half. The side without the tiebreaker will attempt to forcibly shut down CXFS services.
- If there is no CXFS tiebreaker node but system reset or I/O fencing is configured, each half will attempt to reset or fence the other half using a delay heuristic. One half will succeed and continue. The other will lose the reset/fence race and be rebooted/fenced.
- If there is no CXFS tiebreaker node and system reset or I/O fencing is not configured, then both halves will delay, each assuming that one will win the race and reset the other. Both halves will then continue running, because neither will have been reset or fenced, leading to likely data corruption.

  To avoid this situation, you should configure a tiebreaker node, and you must use system reset or I/O fencing. However, if the tiebreaker node (in a cluster with only two server-capable nodes) fails, or if the administrator stops CXFS services, the other node will do a forced shutdown, which unmounts all CXFS filesystems.

If the network partition persists when the losing half attempts to form a CXFS kernel membership, it will have only half the number of server-capable nodes and be unable to form an initial CXFS kernel membership, preventing two CXFS kernel memberships in a single cluster.

For more information, contact SGI professional or managed services.

Configure Filesystems Properly

Configure filesystems properly:

- Perform reconfiguration (including but not limited to adding and deleting filesystems or nodes) during a scheduled cluster maintenance shift and not during production hours.

- Determine whether or not to have all filesystems served off of one metadata server or to use multiple metadata servers to balance the load, depending upon how filesystems will be accessed. The more often a file is accessed, the greater the stress; a filesystem containing many small files that are accessed often causes greater stress than a filesystem with a few large files that are not accessed often.

- Use CXFS on filesystems that contain small files that are frequently accessed. CXFS performs best when data I/O operations are greater than 16 KB and large files are being accessed. (A lot of activity on small files will result in slower performance.)

- Enable the forced unmount feature for CXFS filesystems, which is off by default. Many sites have found that enabling this feature improves the stability of their CXFS clusters, particularly in situations where the filesystem must be unmounted.

  On IRIX nodes, this feature uses the `umount -k` option. The `-k` option attempts to kill processes that have open files or current directories in the appropriate filesystems and then unmount them. That is, it attempts to terminate any I/O going to the filesystem, so that it can unmount it promptly, rather than having to wait for the I/O to finish on its own, causing the unmount to possibly fail.

  On Linux nodes, a similar function is performed with the `fuser -m -k` command and the `umount` command

  This feature is available through the following CXFS GUI menu:

  Tasks
  > Filesystems
  > Unmount a CXFS Filesystem

  You can also specify this feature using the `cmgr` commands to define the filesystem.

  See "Unmount CXFS Filesystems with the GUI" on page 208, and "Define a CXFS Filesystem with `cmgr`" on page 260.

- If you are using NFS or Samba, you should have the NFS or Samba server run on the active metadata server.

- IRIX nodes do not permit nested mount points on CXFS filesystems; that is, you cannot mount an IRIX XFS or CXFS filesystem on top of an existing CXFS filesystem. Although it is possible to mount other filesystems on top of a Linux CXFS filesystem, this is not recommended.

Verify the Configuration

You should always run the following command after any significant configuration change, or whenever problems, warnings or errors occur:

```
/usr/cluster/bin/cxfs-config -xfs -xvm
```

The `cmgr` and CXFS GUI do not always prevent poor configurations. The `cxfs-config` tool can detect a large number of potential problems.

Starting the Cluster

SGI recommends that you do the following:

1. Start CXFS services (using the CXFS GUI or `cmgr`) on the potential metadata servers.
2. Start CXFS services on the client-only tiebreaker node.
3. Start CXFS services on the remaining client-only nodes.

Administration Best Practices

This section discusses the following administration topics:

- "Do Not Run User Jobs on Metadata Servers" on page 122
- "Do Not Run Backups on a Client Node" on page 122
- "Use `cron` Jobs Properly" on page 122
- "Repair Filesystems with Care" on page 123
- "Use Relocation and Recovery Properly" on page 123
- "Shut Down Nodes Properly" on page 123

- "Use `fam` Properly" on page 124
- "Use Trusted IRIX Consistently" on page 124
- "Upgrade the Software Properly" on page 124

Do Not Run User Jobs on Metadata Servers

Do not run user jobs on the CXFS metadata server node.

Do Not Run Backups on a Client Node

SGI recommends that backups are done on the metadata server.

Do not run backups on a client node, because it causes heavy use of non-swappable kernel memory on the metadata server. During a backup, every inode on the filesystem is visited, and if done from a client, it imposes a huge load on the metadata server. The metadata server may experience typical out-of-memory symptoms, and in the worst case can even become unresponsive or crash.

Use `cron` Jobs Properly

Because CXFS filesystems are considered as local on all nodes in the cluster, the nodes may generate excessive filesystem activity if they try to access the same filesystems simultaneously while running commands such as `find`, `ls`, or Linux `slocate`. You should build databases for `rfind` and GNU `locate` only on the metadata server.

On IRIX systems, the default `root` `crontab` on some platforms has the following `find` job that should be removed or disabled on all nodes (line breaks added here for readability):

```
    0       5       *       *       *       /sbin/suattr -m -C CAP_MAC_READ,
CAP_MAC_WRITE,CAP_DAC_WRITE,CAP_DAC_READ_SEARCH,CAP_DAC_EXECUTE=eip
-c "find / -local -type f ’(’ -name core -o -name dead.letter ’)’ -atime +7
-mtime +7 -exec rm -f ’{}’ ’;’"
```

Edit the nodes’ `crontab` file to only execute this `find` command on one metadata server of the cluster.

On Linux systems, there is often a `cron` job to execute `updatedb`, which can be problematic. You should remove this `cron` job or modify it to exclude CXFS directories. (On SGI ProPack for Linux systems on which you are using local XFS, you cannot add `xfs` to the `PRUNEFS` configuration variable to exclude all CXFS filesystems because this would also exclude local XFS filesystems.)

Repair Filesystems with Care

Do not use any filesystem defragmenter software. You can use the IRIX `fsr` command or the Linux `xfs_fsr` command **only** on a metadata server for the filesystem it acts upon.

Always contact SGI technical support before using `xfs_repair` on CXFS filesystems. Only use `xfs_repair` on metadata servers and only when you have verified that all other cluster nodes have unmounted the filesystem.

When using `xfs_repair`, make sure it is run only on a cleanly unmounted filesystem. If your filesystem has not been cleanly unmounted, there will be un-committed metadata transactions in the log, which `xfs_repair` will erase. This usually causes loss of some data and messages from `xfs_repair` that make the filesystem appear to be corrupted.

If you are running `xfs_repair` right after a system crash or a filesystem shutdown, your filesystem is likely to have a dirty log. To avoid data loss, you **MUST** mount and unmount the filesystem before running `xfs_repair`. It does not hurt anything to mount and unmount the filesystem locally, after CXFS has unmounted it, before `xfs_repair` is run.

Use Relocation and Recovery Properly

Use relocation and recovery only on standby nodes. A *standby node* is a server-capable administration node that is configured as a potential metadata server for a given filesystem, but does not currently run any applications (including NFS and Samba) that will use that filesystem. The node can run applications that use other filesystems.

Shut Down Nodes Properly

Use the proper procedures for shutting down nodes. See "Removing and Restoring Cluster Members" on page 329.

When shutting down, resetting, or restarting a CXFS client-only node, do not stop CXFS services on the node. (Stopping CXFS services is more intrusive on other nodes in the cluster because it updates the cluster database. Stopping CXFS services is appropriate only for a CXFS administration node.) Rather, let the CXFS shutdown scripts on the node stop CXFS when the client-only node is shut down or restarted.

If you are going to perform maintenance on a potential metadata server, you should first shut down CXFS services on it. Disabled nodes are not used in CXFS kernel membership calculations, so this action may prevent a loss of quorum.

Use `fam` Properly

If you want to use the file alteration monitor (`fam`), you must remove the `/dev/imon` file from CXFS nodes. Removing this file forces `fam` to poll the filesystem. For more information about the monitor, see the `fam` man page.

Use Trusted IRIX Consistently

If you want to run CXFS and Trusted IRIX, all server-capable nodes in the cluster must run Trusted IRIX. The client-only nodes can run IRIX. Linux and the multiOS platforms are not supported in a cluster with Trusted IRIX. You should configure your system such that all nodes in the cluster have the same user IDs, access control lists (ACLs), and capabilities.

Upgrade the Software Properly

Do the following when upgrading the software:

- Read the release notes when installing and/or upgrading CXFS. These notes contain useful information and caveats needed for a stable install/upgrade.
- Do not make any other configuration changes to the cluster (such as adding new nodes or filesystems) until the upgrade of all nodes is complete and the cluster is running normally.

Chapter 8

Initial Configuration of the Cluster

This chapter provides recommendations and a summary of the steps required to initially configure a cluster using either the graphical user interface (GUI) or the `cmgr` command. You should refer to the information in "Configuration Best Practices" on page 113. You may also wish to use the worksheet provided in Appendix H, "Initial Configuration Checklist" on page 503. If you are converting from an existing FailSafe cluster, see "Set Up an Existing FailSafe Cluster for CXFS with the GUI" on page 170.

This chapter points to detailed descriptions in the task reference chapters and in the *XVM Volume Manager Administrator's Guide*.

For the initial installation, SGI **highly recommends that you use the GUI guided configuration tasks;** see "Configuring with the GUI" on page 129. You should also read through the entire book, including Chapter 17, "Troubleshooting" on page 381, before configuring the cluster.

CXFS requires a license to be installed on each node. If you increase the number of CPUs in your system, you may need a new license; see Chapter 4, "IRIX CXFS Installation" on page 65. For information about other operating systems, see the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

Preliminary Cluster Configuration Steps

Note: Administration must be performed using the GUI connected to a CXFS administration node (one that has the `cluster_admin` software package installed) or using the `cmgr` command on a CXFS administration node.

Complete the following steps to ensure that you are ready to configure the initial cluster:

- "Verify the License"
- "Verify that the Cluster Daemons are Running" on page 126
- "Determine the Hostname of the CXFS Administration Node" on page 127
- "Verify that the `chkconfig` Arguments are On" on page 128

During the course of configuration, you will see various information-only messages in the log files. See "Normal Messages" on page 413.

Verify the License

Verify that you have a CXFS license by using the `-d` option to the `cxfslicense` command. For example:

```
# /usr/cluster/bin/cxfslicense -d
CXFS license granted.
```

If you have a properly installed license, you will also see a `FEATURE CXFS` line in the `license.dat` file on all nodes:

- IRIX: `/var/flexlm/license.dat`
- Linux: `/etc/flexlm/license.dat`

Note: The `license.dat` file cannot simply be copied between nodes because it is unique to each node.

For Linux, you also need a license for XVM.

For more information about installing software licenses, see the *IRIX 6.5 Installation Instructions* booklet.

Verify that the Cluster Daemons are Running

When you **first install** the software, the following daemons should be running on an administration node:

- `fs2d`
- `cmond`
- `cad`
- `crsd`

To determine which daemons are running, enter the following:

```
# ps -ef | grep cluster
```

The following shows an example of the output when just the initial daemons are running; for readability, whitespace has been removed and the daemon names are highlighted:

```
cxfs6 # ps -ef | grep cluster
root 31431     1 0 12:51:36 ?     0:14 /usr/lib32/cluster/cbe/fs2d /var/cluster/cdb/cdb.db #
root 31456 31478 0 12:53:01 ?     0:03 /usr/cluster/bin/crsd -l
root 31475 31478 0 12:53:00 ?     0:08 /usr/cluster/bin/cad -l -lf /var/cluster/ha/log/cad_log --append_log
root 31478     1 0 12:53:00 ?     0:00 /usr/cluster/bin/cmond -L info -f /var/cluster/ha/log/cmond_log
root 31570 31408 0 14:01:52 pts/0 0:00 grep cluster
```

If you do not see these processes, go to the logs to see what the problem might be. If you must restart the daemons, enter the following:

- IRIX:

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

- Linux:

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

To start the `clconfd` daemon on an administration node, enter the following:

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

To start the `cxfs_client` daemon on a client-only node, enter the following:

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

For more information, see "Stopping and Restarting Cluster Administration Daemons" on page 439 and "Daemons" on page 445.

Determine the Hostname of the CXFS Administration Node

When you are initially configuring the cluster with `cmgr`, you must use fully qualified hostname when defining the first node in the pool. (This information is automatically supplied for you in the GUI.)

Also, if you use `nsd`, you must configure your system so that local files are accessed before the network information service (NIS) or the domain name service (DNS).

Caution: It is critical that these files are configured properly and that you enter the primary name for the first node defined in the pool; aliases may be used for subsequent node definitions. See Chapter 4, "IRIX CXFS Installation" on page 65.

Verify that the `chkconfig` Arguments are On

Ensure that the appropriate `chkconfig` arguments are `on`. For more information, see "CXFS `chkconfig` Arguments" on page 289.

IRIX `chkconfig` Verification

For an IRIX node, ensure that `chkconfig` displays the following

```
irix# chkconfig | grep cluster
        cluster              on
        cxfs_cluster         on
```

If it does not, set the flags to `on` and reboot. For example:

```
irix# /etc/chkconfig cluster on
irix# /etc/chkconfig cxfs_cluster on
irix# init 6
```

Or:

```
irix# init 1
irix# /etc/chkconfig cluster on
irix# /etc/chkconfig cxfs_cluster on
irix# init 2
```

Linux `chkconfig` Verification

For a Linux node, use the following commands to verify the `chkconfig` names are set to `on`:

```
[root@linux root]# chkconfig --list | grep cxfs
cxfs_cluster    0:off   1:off   2:on    3:on    4:on    5:on    6:off
cxfs            0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

If they are not, set them to `on` and reboot. For example:

```
[root@linux root]# chkconfig cxfs_cluster on
[root@linux root]# chkconfig cxfs on
[root@linux root]# reboot
```

Configuring with the GUI

To initially configure the cluster with GUI, do the following:

- "Start the GUI" on page 129
- "Set Up a New Cluster with the GUI" on page 131
- "Set Up a New CXFS Filesystem with the GUI" on page 132

The CXFS administration node to which you connect the GUI affects your view of the cluster. You should wait for a change to appear in the view area before making another change; the change is not guaranteed to be propagated across the cluster until it appears in the view area. You should only make changes from one instance of the GUI at any given time; changes made by a second GUI instance may overwrite changes made by the first instance.

Start the GUI

Start the CXFS Manager by entering the following:

```
# /usr/sbin/cxfsmgr
```

You can also start the GUI from your web browser on a Microsoft Windows, Linux, or other platform. To do this, enter `http://`*server*`/CXFSManager/` (where *server* is the name of a CXFS administration node in the pool) and press **Enter**. At the resulting webpage, click the CXFS Manager icon. This method of launching CXFS Manager requires you to have enabled Java in your browser's preferences and have installed the appropriate Java plug-in. (After installing the plug-in, you must close any existing Java windows and restart your browser.) The CXFS administration node must be running a web server, such as Apache, and have the following software installed:

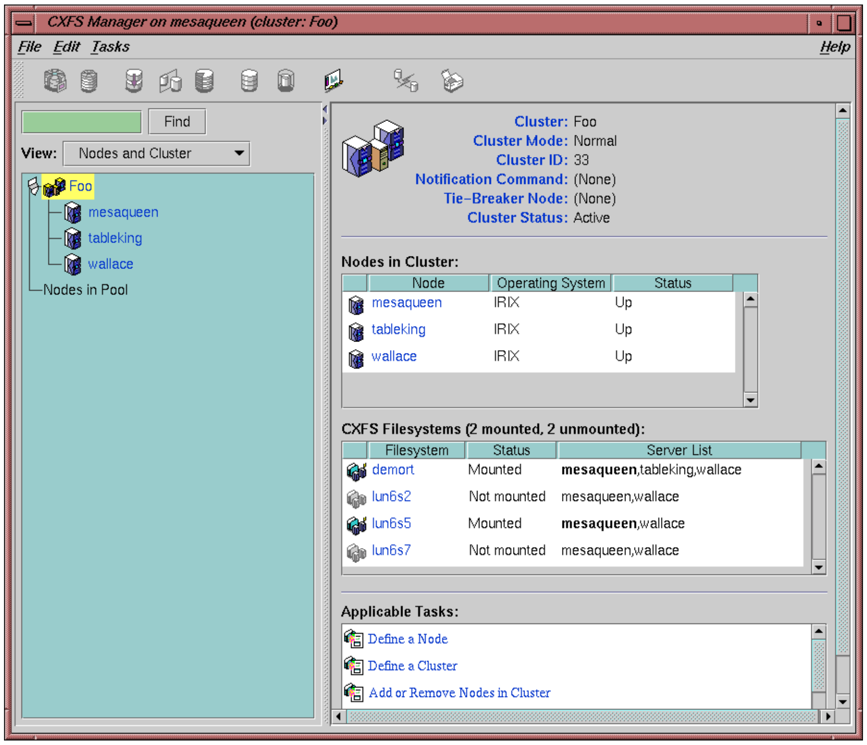
- IRIX: `sysadm_cxfs.sw.web`
- Linux: `sysadm_cxfs-web`

Note: If you load the GUI using Netscape on IRIX and then switch to another page in Netscape, CXFS Manager GUI will not operate correctly. To avoid this problem, leave the CXFS Manager GUI web page up and open a new Netscape window if you want to view another page.

There are other methods of starting the GUI. For more information, see "Starting the GUI" on page 150.

Supply the name of the CXFS administration node you wish to connect to and the `root` password.

Figure 8-1 shows an example of the CXFS Manager window.



Cluster: Foo
Cluster Mode: Normal
Cluster ID: 33
Notification Command: (None)
Tie-Breaker Node: (None)
Cluster Status: Active

Nodes in Cluster:

Node	Operating System	Status
mesaqueen	IRIX	Up
tableking	IRIX	Up
wallace	IRIX	Up

CXFS Filesystems (2 mounted, 2 unmounted):

Filesystem	Status	Server List
demort	Mounted	**mesaqueen**,tableking,wallace
lun6s2	Not mounted	mesaqueen,wallace
lun6s5	Mounted	**mesaqueen**,wallace
lun6s7	Not mounted	mesaqueen,wallace

Applicable Tasks:

Define a Node
Define a Cluster
Add or Remove Nodes in Cluster

Figure 8-1 CXFS Manager

Set Up a New Cluster with the GUI

Note: Within the CXFS tasks, you can click any **blue** text to get more information about that concept or input field. In every task, the cluster configuration will not update until you click on **OK**.

The **Set Up a New Cluster** task in the **Guided Configuration** menu leads you through the steps required to create a new cluster. It encompasses tasks that are detailed elsewhere.

Do the following:

1. Click **Define a Node** to define the CXFS administration node to which you are connected. See "Define a Node with the GUI" on page 172.

   Note: If you attempt to define a cluster or other object before the local node has been defined, you will get an error message that says:

   ```
   No nodes are registered on servername. You cannot define a cluster
   until you define the node to which the GUI is connected. To do so,
   click "Continue" to launch the "Set Up a New Cluster" task.
   ```

   ---

2. (*Optional*) **After** the first node icon appears in the view area on the left, click step 2, **Define a Node**, to define the other nodes in the cluster. To use private network failover, you must use the `cmgr` command's `add net` subcommand to group the NICs into networks; by default, only the priority `1` NICs form a network. See "Define a Node with `cmgr`" on page 224. See "Define a Node with the GUI" on page 172.

   Note: Do not define another node until this node appears in the view area. If you add nodes too quickly (before the database can include the node), errors will occur.

   Repeat this step for each node. For large clusters, define only the administration nodes first; see "Configuring a Large Cluster" on page 145.

3. Click **Define a Cluster** to create the cluster definition. See "Define a Cluster with the GUI" on page 188. Verify that the cluster appears in the view area. Choose **View: Nodes and Cluster**.

4. After the cluster icon appears in the view area, click **Add/Remove Nodes in Cluster** to add the nodes to the new cluster. See "Add or Remove Nodes in the Cluster with the GUI" on page 181.

   Click **Next** to move to the second screen of tasks.

5. (*Optional*) Click on **Test Connectivity** to verify that the nodes are physically connected. See "Test Node Connectivity with the GUI" on page 187. (This test requires the proper configuration; see "IRIX Modifications for CXFS Connectivity Diagnostics" on page 80, "Linux Modifications for CXFS Connectivity Diagnostics" on page 93.)

6. If you are using I/O fencing, define the switch in the cluster; see the release notes for supported switches. I/O fencing is required for nodes without system controllers; see "Requirements" on page 37.

7. Click **Start CXFS Services**. See "Start CXFS Services with the GUI" on page 191.

8. Click **Close**. Clicking on **Close** exits the task; it does not undo the task.

Set Up a New CXFS Filesystem with the GUI

Note: Within the CXFS tasks, you can click any blue text to get more information about that concept or input field. In every task, the cluster configuration will not update until you click **OK**.

The **Set Up a New CXFS Filesystem** task leads you through the steps required to create a new filesystem and mount it on all nodes in your cluster. It encompasses tasks that are detailed elsewhere.

Do the following:

1. Click **Start CXFS Services** if the services have not been started already. (The current status is displayed beneath the task link.) See "Start CXFS Services with the GUI" on page 191.

2. Click **Label Disks**.

   Note: The disk must be initialized before being labeled. If your disk has not been initialized during factory set-up, use the IRIX `fx` command or Linux `fdisk` command to initialize the disk.

For information about XVM tasks, see the *XVM Volume Manager Administrator's Guide*.

3. Create slices, which define the physical storage, on the labeled disk. Click **Slice Disks**.
4. Create the type of filesystem you want: stripe, mirror, or concat.
5. Click **Make the Filesystem**. If you do not want to use the default options, click **Specify Sizes** and go to the next page. For more information, see the `mkfs` man page, the *IRIX Admin: Disks and Filesystems* guide, and the *XVM Volume Manager Administrator's Guide*.
6. Click **Define a CXFS Filesystem**. This task lets you define a new filesystem, set the ordered list of potential metadata servers, and set the list of client nodes for the filesystem. See "Define CXFS Filesystems with the GUI" on page 204.
7. Click **Mount a CXFS Filesystem**. This task lets you mount the filesystem on all nodes in the cluster. See "Mount CXFS Filesystems with the GUI" on page 208.

Repeat these steps for each filesystem.

Configuring with the `cmgr` Command

Note: For the initial installation, SGI highly recommends that you use the GUI guided configuration tasks. See "Configuring with the GUI" on page 129.

For details about `cmgr` commands, see the man page and Chapter 10, "Reference to `cmgr` Tasks for CXFS" on page 217.

To initially configure the cluster with the `cmgr` command, do the following:

1. Follow the directions in "Preliminary Cluster Configuration Steps" on page 125.
2. Define the nodes that are eligible to be part of the cluster. The hostname/IP-address pairings and priorities of the networks must be the same for each node in the cluster. See "Define a Node with `cmgr`" on page 224.

   For large clusters, SGI recommends that you define only the first three CXFS administration nodes and then continue on to the next step; add the remaining nodes after you have a successful small cluster.

The following example sequence defines three nodes. (To use the default value for a prompt, press the `Enter` key. The `Enter` key is not shown in the examples in this guide.)

To define the first node, named `cxfs6`, enter the following:

```
cxfs6 # /usr/cluster/bin/cmgr -p
Welcome to SGI Cluster Manager Command-Line Interface

cmgr> define node cxfs6
Enter commands, you may enter "done" or "cancel" at any time to exit

Hostname[optional] ?
Is this a FailSafe node <true|false> ? false
Is this a CXFS node <true|false> ? true
Operating System  <IRIX|Linux32|Linux64|AIX|HPUX|Solaris|MacOSX|Windows> ? irix
Node Function <server_admin|client_admin|client_only> ? server_admin
Node ID[optional]?
Partition ID[optional] ? (0)
Do you wish to define failure hierarchy[y/n]:n
Reset type <powerCycle|reset|nmi>  ? (powerCycle)
Do you wish to define system controller info[y/n]:y
Sysctrl Type <msc|mmsc|l1|l2> ? (msc)
Sysctrl Password[optional] ? ( )
Sysctrl Status <enabled|disabled> ? enabled
Sysctrl Owner ? cxfs8
Sysctrl Device ? /dev/ttyd2
Sysctrl Owner Type <tty|network> ? (tty)
Number of Network Interfaces ? (1)
NIC 1 - IP Address ? cxfs6
NIC 1 - Heartbeat HB (use network for heartbeats) <true|false> ? true
NIC 1 - (use network for control messages) <true|false> ? true

NIC 1 - Priority <1,2,...> 1


Successfully defined node cxfs6
```

To define the second node, named `cxfs7`, enter the following:

```
cmgr> define node cxfs7
Enter commands, you may enter "done" or "cancel" at any time to exit
```

```
Hostname[optional] ?
Is this a FailSafe node <true|false> ? false
Is this a CXFS node <true|false> ? true
Node Function <server_admin|client_admin|client_only> ? server_admin
Operating System <IRIX|Linux32|Linux64|AIX|HPUX|Solaris|MacOSX|Windows> ? irix
Node ID[optional] ?
Partition ID[optional] ? (0)
Do you wish to define failure hierarchy[y/n]:n
Reset type <powerCycle|reset|nmi>  ? (powerCycle)
Do you wish to define system controller info[y/n]:y
Sysctrl Type <msc|mmsc|l2> ? (msc)
Sysctrl Password[optional] ? ( )
Sysctrl Status <enabled|disabled> ? enabled
Sysctrl Owner ? cxfs6
Sysctrl Device ? /dev/ttyd2
Sysctrl Owner Type <tty|network> ? (tty)
Number of Network Interfaces ? (1)
NIC 1 - IP Address ? cxfs7
NIC 1 - Heartbeat HB (use network for heartbeats) <true|false> ? true
NIC 1 - (use network for control messages) <true|false> ? true

NIC 1 - Priority <1,2,...> 1

Successfully defined node cxfs7
```

To define the third node, named `cxfs8`, enter the following:

```
cmgr> define node cxfs8
Enter commands, you may enter "done" or "cancel" at any time to exit

Hostname[optional] ?
Is this a FailSafe node <true|false> ? false
Is this a CXFS node <true|false> ? true
Node Function <server_admin|client_admin|client_only> ? server_admin
Operating System <IRIX|Linux32|Linux64|AIX|HPUX|Solaris|MacOSX|Windows> ? irix
Node ID[optional] ?
Partition ID[optional] ? (0)
Do you wish to define failure hierarchy[y/n]:n
Reset type <powerCycle|reset|nmi>  ? (powerCycle)
Do you wish to define system controller info[y/n]:y
Sysctrl Type <msc|mmsc|l2> ? (msc)
Sysctrl Password[optional] ? ( )
Sysctrl Status <enabled|disabled> ? **enabled**
Sysctrl Owner ? **cxfs7**
Sysctrl Device ? **/dev/ttyd2**
Sysctrl Owner Type <tty|network> ? (tty)
Number of Network Interfaces ? (1)
NIC 1 - IP Address ? **cxfs8**
NIC 1 - Heartbeat HB (use network for heartbeats) <true|false> ? **true**
NIC 1 - (use network for control messages) <true|false> ? **true**

NIC 1 - Priority <1,2,...> **1**

Successfully defined node cxfs8
```

You now have three nodes defined in the pool. To verify this, enter the following:

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

3 Machine(s) defined
        cxfs6
        cxfs7
        cxfs8
```

To show the contents of node `cxfs6`, enter the following:

```
cmgr> **show node cxfs6**

Logical Machine Name: cxfs6
Hostname: cxfs6.americas.sgi.com
Operating System: irix
Node Is FailSafe: false
Node Is CXFS: true
Node Function: server_admin
Nodeid: 13203
Partition id: 0
Reset type: powerCycle
System Controller: msc
System Controller status: enabled
System Controller owner: cxfs8
System Controller owner device: /dev/ttyd2
System Controller owner type: tty
ControlNet Ipaddr: cxfs6
ControlNet HB: true
ControlNet Control: true
ControlNet Priority: 1
```

3. Define the cluster and add the nodes to it. See "Define a Cluster with `cmgr`" on page 246.

   For example, to define a cluster named `cxfs6-8` and add the nodes that are already defined, enter the following:

```
cmgr> **define cluster cxfs6-8**
Enter commands, you may enter "done" or "cancel" at any time to exit

Is this a FailSafe cluster <true|false> **false** ?
Is this a CXFS cluster  <true|false> **true** ?
Cluster Notify Cmd [optional] ?
Cluster Notify Address [optional] ?
Cluster mode <normal|experimental>[optional]
Cluster ID ? **22**

No nodes in cluster cxfs6-8

Add nodes to or remove nodes from cluster cxfs6-8
Enter "done" when completed or "cancel" to abort

cxfs6-8 ? **add node cxfs6**
cxfs6-8 ? **add node cxfs7**
cxfs6-8 ? **add node cxfs8**
cxfs6-8 ? **done**
Successfully defined cluster cxfs6-8

Added node <cxfs6> to cluster <cxfs6-8>
Added node <cxfs7> to cluster <cxfs6-8>
Added node <cxfs8> to cluster <cxfs6-8>
```

   The fail action hierarchy is the set of instructions that determines which method is used in case of failure. If you set a hierarchy including fencing, you could define the switch at this point. For more information, see "Switches and I/O Fencing Tasks with `cmgr`" on page 273.

   To define a list of private networks that can be used in case the highest priority network (consisting by default of the priority `1` NICs) fails, use the `add net` command; see "Define a Node with `cmgr`" on page 224.

For more information, see "Define a Cluster with `cmgr`" on page 246.

To verify the cluster and its contents, enter the following:

```
cmgr> show clusters

1 Cluster(s) defined
        cxfs6-8

cmgr> show cluster cxfs6-8
Cluster Name: cxfs6-8
Cluster Is FailSafe: false
Cluster Is CXFS: true
Cluster ID: 22
Cluster CX mode: normal

Cluster cxfs6-8 has following 3 machine(s)
        cxfs6
        cxfs7
        cxfs8

CXFS Failover Networks:
     default network 0.0.0.0, mask 0.0.0.0
```

For an example of this step using a script, see "Script Example" on page 277.

4. Start CXFS services for each node in the cluster by entering the following:

   ```
   start cx_services for cluster clustername
   ```

   For example:

   ```
   cmgr> start cx_services for cluster cxfs6-8

   CXFS services have been activated in cluster cxfs6-8
   ```

   This action starts CXFS services and sets the configuration so that CXFS services will be restarted automatically whenever a node reboots.

   Note: If you stop CXFS services using either the GUI or `cmgr`, the automatic restart capability is turned off. You must start CXFS services again to reinstate the automatic restart capability.

To verify that CXFS services have been started in the cluster and there is a membership formed, you can use the following `cmgr` command:

`show status of cluster` *clustername*

For example:

```
cmgr> show status of cluster cxfs6-8

Cluster (cxfs6-8) is not configured for FailSafe


CXFS cluster state is ACTIVE.
```

You can also use the `clconf_info` command. For example:

```
cxfs6 # /usr/cluster/bin/clconf_info

Event at [2004-04-16 09:20:59]

Membership since Fri Apr 16 09:20:56 2004

_____________ ______ _________ ______ ______
Node          NodeID Status    Age    CellID
_____________ ______ _________ ______ ______
cxfs7          12812 up         0          1
cxfs6          13203 up         0          0
cxfs8          14033 up         0          2
_____________ ______ _________ ______ ______
0 CXFS FileSystems
```

For more information, see "Display a Cluster with `cmgr`" on page 253.

5. Obtain a shell window for one of the CXFS administration nodes in the cluster and use the `fx` command to create a volume header on the disk drive. For information, see *IRIX Admin: Disks and Filesystems*.

6. Create the XVM logical volumes. In the shell window, use the `xvm` command line interface. For information, see the *XVM Volume Manager Administrator's Guide*.

7. Make the filesystems. In the shell window, use the `mkfs` command. For information, see the *XVM Volume Manager Administrator's Guide* and *IRIX Admin: Disks and Filesystems*.

8. Define the filesystems by using the `define cxfs_filesystem` subcommand to `cmgr`. See "CXFS Filesystem Tasks with `cmgr`" on page 260.

   The following example shows two potential metadata servers for the `fs1` filesystem; if `cxfs6` (the preferred server, with rank 0) is not up when the cluster starts or later fails or is removed from the cluster, then `cxfs7` (rank 1) will be used. It also shows the filesystem being mounted by default on all nodes in the cluster (`Default Local Status enabled`) but explicitly not mounted on `cxfs8`.

   Note: Although the list of metadata servers for a given filesystem is ordered, it is impossible to predict which server will become the server during the boot-up cycle because of network latencies and other unpredictable delays.

   Do the following:

```
cmgr> define cxfs_filesystem fs1 in cluster cxfs6-8

(Enter "cancel" at any time to abort)

Device ? /dev/cxvm/d76lun0s0
Mount Point ? /mnts/fs1
Mount Options[optional] ?
Use Forced Unmount ? <true|false> ? false
Default Local Status <enabled|disabled> ? (enabled)

DEFINE CXFS FILESYSTEM OPTIONS

        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:1
```

```
No current servers

Server Node ? cxfs6
Server Rank ? 0

        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:1
Server Node ? cxfs7
Server Rank ? 1


        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:5

No disabled clients

Disabled Node ? cxfs8

        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:7

Current settings for filesystem (fs1)

CXFS servers:
        Rank 0          Node cxfs6
        Rank 1          Node cxfs7

Default local status: enabled

No explicitly enabled clients

Explicitly disabled clients:
        Disabled Node: cxfs8

        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:9
Successfully defined cxfs_filesystem fs1

cmgr> define cxfs_filesystem fs2 in cluster cxfs6-8

(Enter "cancel" at any time to abort)
```

```
Device ? /dev/cxvm/d77lun0s0
Mount Point ? /mnts/fs2
Mount Options[optional] ?
Use Forced Unmount ? <true|false> ? false
Default Local Status <enabled|disabled> ? (enabled)

DEFINE CXFS FILESYSTEM OPTIONS

        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:1

Server Node ? cxfs8
Server Rank ? 0

        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:7

Current settings for filesystem (fs2)

CXFS servers:
        Rank 0          Node cxfs8
```

```
Default local status: enabled

No explicitly enabled clients

No explicitly disabled clients

        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:9
Successfully defined cxfs_filesystem fs2
```

To see the modified contents of cluster `cxfs6-8`, enter the following:

```
cmgr> show cxfs_filesystems in cluster cxfs6-8

fs1
fs2
```

9. Mount the filesystems on all nodes in the cluster by using the `admin cxfs_mount cxfs_filesystem` subcommand to `cmgr`. See "Mount a CXFS Filesystem with `cmgr`" on page 266. For example:

```
cmgr> admin cxfs_mount cxfs_filesystem fs1 in cluster cxfs6-8
cxfs_mount operation successful

cmgr> admin cxfs_mount cxfs_filesystem fs2 in cluster cxfs6-8
cxfs_mount operation successful
```

10. To quit out of `cmgr`, enter the following:

```
cmgr> quit
```

Configuring a Large Cluster

When configuring a large cluster, you should ensure that a small cluster containing just the server-capable administration nodes is fully functional before adding client-only nodes. By building up the cluster with client-only nodes in small groups, you will minimize concurrent operational issues and use the database most efficiently.

Do the following:

1. Create the initial cluster with just the server-capable nodes and test it:
   a. Define all of the server-capable administration nodes.
   b. Define the cluster.
   c. Add all of the server-capable administration nodes to the cluster.
   d. Create the filesystems as described in "Set Up a New CXFS Filesystem with the GUI" on page 132.
   e. Verify that the nodes are all part of the cluster membership and that the filesystems are mounted and fully functional.
2. Add the client-only nodes to the database:
   a. Define **all** client-only nodes.
   b. Add **all** client-only nodes to the cluster.
3. Gradually build up the functional cluster with subsets of client-only nodes:
   a. Start CXFS services on a **subset** of four client-only nodes.
   b. Ensure that the nodes are part of the cluster membership and that the filesystems are fully functional.
4. Repeat step 3 as needed to complete the cluster membership.

Following is an example script for configuring a one-node cluster that can be copied and repeated for the number of nodes required:

```
#!/usr/cluster/bin/cmgr -f
# Node nodename definition
define node nodename
        set hostname to nodename
        set operating_system to OS
        set node_function to server_admin|client_admin|client_only
        set is_failsafe to false
        set is_cxfs to true
        set nodeid to nodeID#
        set hierarchy to [system][fence][reset][fencereset][shutdown]
        set reset_type to powerCycle|reset|nmi
        add nic IP address  or nodename
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 1
        done
done
# Define cluster and add nodes to the cluster
define cluster clustername
        set is_failsafe to false
        set is_cxfs to true
        set cx_mode to normal
        set clusterid to clusterID#
done
modify cluster clustername
        add node nodename
done
set cluster clustername
define cxfs_filesystem filesystemname
        set device_name to /dev/cxvm/volumename
        set mount_point to /mountpoint
        set force to false
        set dflt_local_status to enabled
        add cxfs_server server1, server2, etc
                set rank to 0
        done
done
# Setting CXFS parameters
modify cx_parameters
        set tie_breaker to none
done
start cx_services for cluster clustername
quit
```

After adding one client node to the cluster, use the `build_cmgr_script` command to generate a script that can be used as a template for adding more client-only nodes to the cluster. The `build_cmgr_script` will generate a script for defining the entire cluster. The commands for generating the single client-only node can be easily extracted, replicated, and modified in a new script to define the remaining client-only nodes. Using scripts to define the client-only nodes in a large cluster is highly recommended.

For more information about using scripts and the `cmgr` command, see Chapter 10, "Reference to `cmgr` Tasks for CXFS" on page 217

Chapter 9

Reference to GUI Tasks for CXFS

This chapter discusses the CXFS Manager graphical user interface (GUI). It contains detailed information about CXFS tasks and an overview of XVM tasks. (For details about XVM tasks, see the *XVM Volume Manager Administrator's Guide*.)

This chapter contains the following sections:

Note: CXFS requires a license to be installed on each node. If you install the software without properly installing the license, you will get an error and will not be able to use the CXFS Manager GUI. For more information about licensing, see Chapter 2, "CXFS and XVM FLEXlm Licenses" on page 51. For information about licensing on nodes running operating systems other than IRIX or Linux, see the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

GUI Overview

The GUI lets you set up and administer CXFS filesystems and XVM logical volumes. It also provides icons representing status and structure.

This section provides an overview of the GUI:

- "GUI Features" on page 156
- "Key to Icons and States" on page 166

Note: CXFS is incompatible with the Red Hat cluster manager available in the Red Hat Advanced Server product.

Starting the GUI

There are several methods to start the GUI and connect to a node.

Starting the GUI on IRIX

To start the GUI, use one of the following methods:

- On an IRIX system where the CXFS GUI-client software (`sysadm_cxfs.sw.client`) and desktop support software (`sysadm_cxfs.sw.desktop`) are installed, do one of the following:

  Note: SGI does not recommend this method across a wide-area network (WAN) or virtual private network (VPN), or if the IRIX system has an R5000 or earlier CPU and less than 128-MB memory.

  – Enter the following command line:

  ```
  # /usr/sbin/cxfsmgr
  ```

  (The `cxdetail` and `cxtask` commands perform the identical function as `cxfsmgr`; these command names are kept for historical purposes.)

  – Choose the following from the Toolchest:

  System
      > CXFS Manager

  You must restart the Toolchest after installing CXFS in order to see the **CXFS** entry on the Toolchest display. Enter the following commands to restart the Toolchest:

  ```
  # killall toolchest
  # /usr/bin/X11/toolchest &
  ```

If you are using WAN or VPN, see "Starting the GUI on a PC" on page 151.

Starting the GUI on Linux

To start the GUI on a Linux system where the CXFS GUI-client software (`sysadm_cxfs-client`) is installed, do the following:

1. Obtain and install the J2SE 1.4.2 (latest patch) software available from http://java.sun.com
2. Enter the following command line:

```
# /usr/sbin/cxfsmgr
```

Starting the GUI on a PC

To start the GUI on a PC or if you want to perform administration from a remote location via VPN or WAN, do the following:

- Install a web server (such as Apache) and the following packages on one of the powerful CXFS administration nodes:
  - IRIX: `sysadm_cxfs.sw.web` and `sysadm_xvm.sw.web`
  - Linux: `sysadm_cxfs-web` and `sysadm_xvm-web`
- Install the Java2 v1.4.1 or v1.3.2 plug-in on your PC.
- Add the following to your `httpd.conf` file:

```
<Location "/CXFSManager">
    Options Includes ExecCGI FollowSymLinks
    DirectoryIndex index.html index.shtml
</Location>
```

- Close any existing Java windows and restart the Web browser on the PC.
- Enter the URL `http://`*server*`/CXFSManager/` where *server* is the name of a CXFS administration node in the pool
- At the resulting webpage, click the CXFS Manager icon.

Note: This method can be used on IRIX systems, but it is not the preferred method unless you are using WAN or VPN. If you load the GUI using Netscape on IRIX and then switch to another page in Netscape, CXFS Manager GUI will not operate correctly. To avoid this problem, leave the CXFS Manager GUI web page up and open a new Netscape window if you want to view another web page.

Summary of GUI Platforms

Table 9-1 describes the platforms where the GUI may be started, connected to, and displayed.

Table 9-1 GUI Platforms

GUI Mode	Where You Start the GUI	Where You Connect the GUI	Where the GUI Displays
`cxfsmgr`	Any IRIX system (such as an SGI 2000 series or SGI O2 workstation) with `sysadm_cxfs.sw.client` and `sysadm_cxfs.sw.desktop` software installed A Linux system with `sysadm_cxfs-client` installed	The CXFS administration node in the pool that you want to use for cluster administration	The system where the GUI was invoked
Toolchest	Any IRIX system (such as an SGI 2000 series or SGI O2 workstation) with `sysadm_cxfs.sw.client` and `sysadm_cxfs.sw.desktop` software installed	The CXFS administration node in the pool that you want to use for cluster administration	The system where the GUI was invoked
Web	Any system with a web browser and Java2 1.4.1 or 1.4.2 plug-in installed and enabled	The CXFS administration node in the pool that you want to use for cluster administration	The same system with the web browser

Logging In

To ensure that the required GUI privileges are available for performing all of the tasks, you should log in to the GUI as `root`. However, some or all privileges can be granted to any other user using the GUI privilege tasks; see "Privileges Tasks with the GUI" on page 211. (Under IRIX, this functionality is also available with the Privilege Manager, part of the IRIX Interactive Desktop System Administration `sysadmdesktop` product. For more information, see the *Personal System Administration Guide*.)

A dialog box will appear prompting you to log in to a CXFS host. You can choose one of the following connection types:

- **Local** runs the server-side process on the local host instead of going over the network
- **Direct** creates a direct socket connection using the `tcpmux` TCP protocol (`tcpmux` must be enabled)
- **Remote Shell** connects to the server via a user-specified command shell, such as `rsh` or `ssh`. For example:

  ```
  ssh -l root servername
  ```

  Note: For secure connection, choose **Remote Shell** and type a secure connection command using a utility such as `ssh`. Otherwise, the GUI will not encrypt communication and transferred passwords will be visible to users of the network.

- **Proxy** connects to the server through a firewall via a proxy server

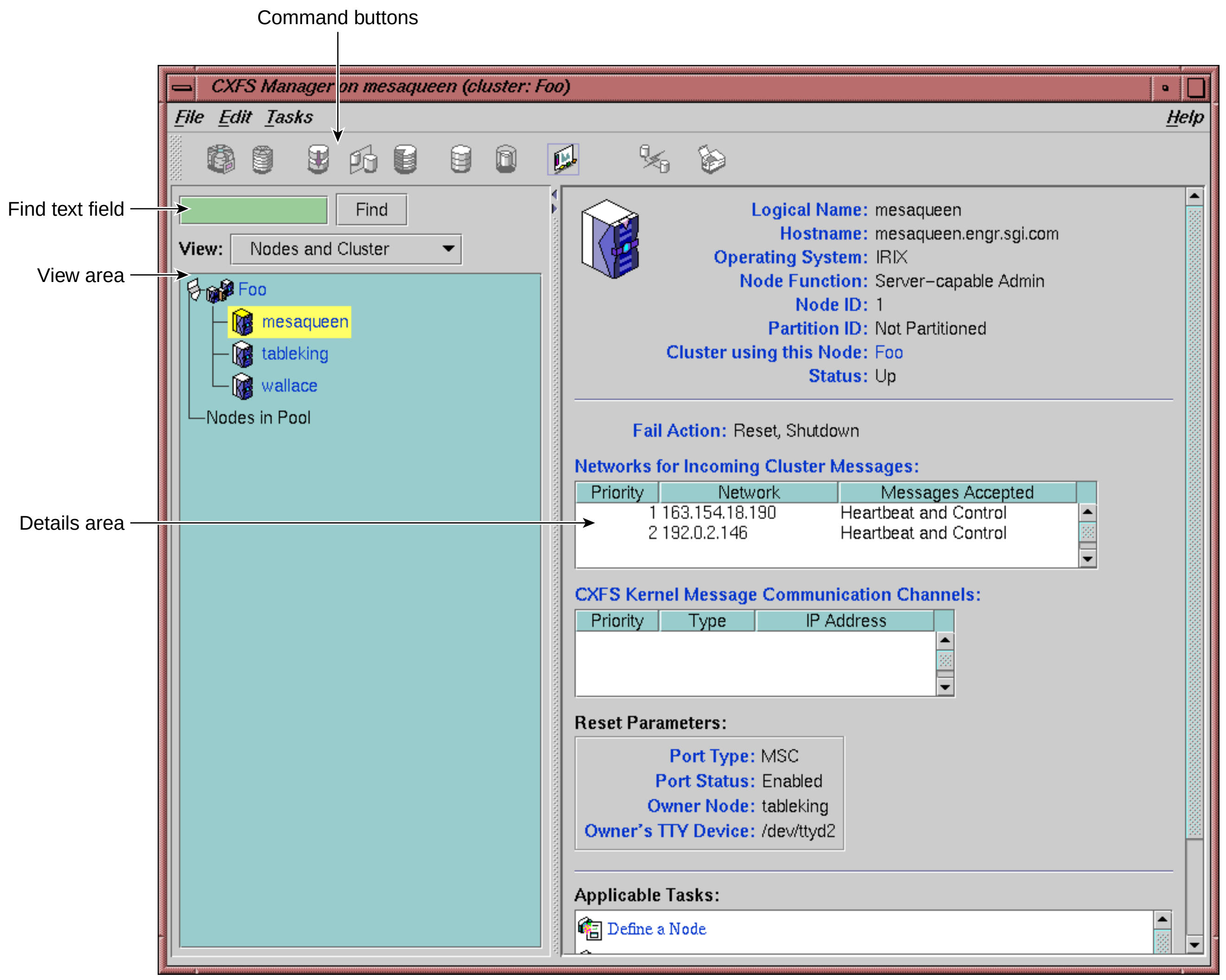
Making Changes Safely

Do not make configuration changes on two different administration nodes in the pool simultaneously, or use the CXFS GUI, `cmgr`, and `xvm` commands simultaneously to make changes. You should run one instance of the `cmgr` command or the CXFS GUI on a single administration node in the pool when making changes at any given time. However, you can use any node in the pool when requesting status or configuration information. Multiple CXFS Manager windows accessed via the **File** menu are all part of the same application process; you can make changes from any of these windows.

The CXFS administration node to which you connect the GUI affects your view of the cluster. You should wait for a change to appear in the *view area* before making another change; the change is not guaranteed to be propagated across the cluster until it appears in the view area. (To see the location of the view area, see Figure 9-1 on page 155.) The entire cluster status information is sent to every CXFS administration node each time a change is made to the cluster database.

GUI Windows

Figure 9-1 shows the **CXFS Manager** window displaying information for a specific component in the *details area*. For information about using the *view area* to monitor status and an explanation of the icons and colors, see "Cluster Status" on page 362.



Logical Name: mesaqueen
Hostname: mesaqueen.engr.sgi.com
Operating System: IRIX
Node Function: Server−capable Admin
Node ID: 1
Partition ID: Not Partitioned
Cluster using this Node: Foo
Status: Up

Fail Action: Reset, Shutdown

Networks for Incoming Cluster Messages:

Priority	Network	Messages Accepted
1	163.154.18.190	Heartbeat and Control
2	192.0.2.146	Heartbeat and Control

CXFS Kernel Message Communication Channels:

Priority	Type	IP Address

Reset Parameters:

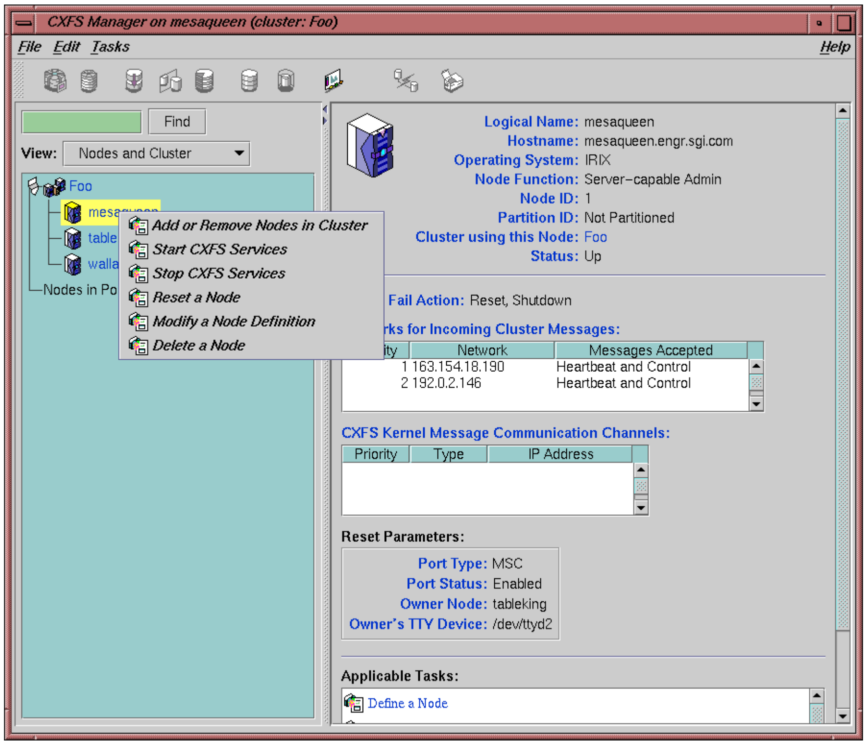
Port Type: MSC
Port Status: Enabled
Owner Node: tableking
Owner's TTY Device: /dev/ttyd2

Applicable Tasks:

Define a Node

Figure 9-1 CXFS Manager GUI Showing Details for a Node

Figure 9-2 shows an example of the pop-up menu of applicable tasks that appears when you click the right mouse button on a selected item; in this example, clicking on the node name `trinity` displays a list of applicable tasks.



Logical Name: mesaqueen
Hostname: mesaqueen.engr.sgi.com
Operating System: IRIX
Node Function: Server–capable Admin
Node ID: 1
Partition ID: Not Partitioned
Cluster using this Node: Foo
Status: Up

Fail Action: Reset, Shutdown

rks for Incoming Cluster Messages:

ity	Network	Messages Accepted
1	163.154.18.190	Heartbeat and Control
2	192.0.2.146	Heartbeat and Control

CXFS Kernel Message Communication Channels:

Priority	Type	IP Address

Reset Parameters:

Port Type: MSC
Port Status: Enabled
Owner Node: tableking
Owner's TTY Device: /dev/ttyd2

Figure 9-2 Pop-up Menu that Appears After Clicking the Right Mouse Button

GUI Features

The **CXFS Manager** GUI allows you to administer the entire CXFS cluster from a single point. It provides access to the tools that help you set up and administer your CXFS cluster:

- *Tasks* let you set up and monitor individual components of a CXFS cluster, including XVM volumes. For details about XVM tasks, see *XVM Volume Manager Administrator's Guide*.
- *Guided configuration tasks* consist of a group of tasks collected together to accomplish a larger goal. For example, **Set Up a New Cluster** steps you through the process for creating a new cluster and allows you to launch the necessary individual tasks by simply clicking their titles.

This section discusses the following:

- "GUI Window Layout" on page 157
- "File Menu" on page 158
- "Edit Menu" on page 158
- "Tasks Menu" on page 158
- "Help Menu" on page 159
- "Shortcuts Using Command Buttons" on page 159
- "View Menu" on page 161
- "Performing Tasks" on page 162
- "Using Drag-and-Drop for XVM Configuration" on page 163
- "Analyzing I/O Performance with Performance Co-Pilot on an IRIX Node" on page 163
- "Structuring Volume Topologies" on page 164
- "Configuring Disks" on page 164
- "Getting More Information" on page 165
- "Important GUI and `xvm` Command Differences" on page 165

GUI Window Layout

By default, the window is divided into two sections: the *view area* and the *details area* (see Figure 9-1 on page 155). The details area shows generic overview text if no item is selected in the view area. You can use the arrows in the middle of the window to shift the display.

File Menu

The **File** menu lets you display the following:

- Multiple windows for this instance of the GUI
- System log file:
  - IRIX: `/var/adm/SYSLOG`
  - Linux: `/var/log/messages`
- System administration log file:
  - IRIX: `/var/sysadm/salog`
  - Linux: `/var/lib/sysadm/salog`

  The `salog` file shows the commands run directly by this instance of the GUI or some other instance of the GUI running commands on the system. (Changes should not be made simultaneously by multiple instances of the GUI or the GUI and `cmgr`.)

The **File** menu also lets you close the current window and exit the GUI completely.

Edit Menu

The **Edit** menu lets you expand and collapse the contents of the view area. You can choose to automatically expand the display to reflect new nodes added to the pool or cluster. You can also use this menu to select all items in the view menu or clear the current selections.

Tasks Menu

The **Tasks** menu contains the following:

- **Guided Configuration**, which contains the tasks to set up your cluster, define filesystems, create volumes, check status, and modify an existing cluster
- **Nodes**, which contains tasks to define and manage the nodes
- **Cluster**, which contains tasks to define and manage the cluster
- **Cluster Services**, which allows you to start and stop CXFS services, set the CXFS tiebreaker node, set the log configuration, and revoke or allow CXFS kernel membership of the local node

- **Switches and I/O Fencing**, which contains tasks to configure switch definitions and manage I/O fencing
- **Disks**, which contains XVM disk administration tasks
- **Volume Elements**, which contains tasks to create, delete, modify, and administer XVM volume elements
- **Filesystems**, which contains tasks to define and manage filesystems and relocate a metadata server
- **Privileges**, which lets you grant or revoke access to a specific task for one or more users
- **Find Tasks**, which lets you use keywords to search for a specific task

Help Menu

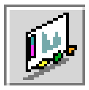
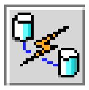
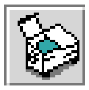
The **Help** menu provides an overview of the GUI and a key to the icons. You can also get help for certain items in blue text by clicking on them.

Shortcuts Using Command Buttons

The command buttons along the top of the GUI window provide a method of performing tasks quickly. When you click a button, the corresponding task executes using default values, usually without displaying a task window. To override the defaults, launch the task from the **Tasks** menu. Table 9-2 summarizes the shortcuts available; for details about these tasks, see the *XVM Volume Manager Administrator's Guide*.

Table 9-2 Command Buttons

Button	Task
	Labels selected unlabeled disks. If the selected disks include foreign and/or labeled disks, the **Label Disks** task will be run.
	Brings up the **Slice Disk** task with the selected disks as default inputs
	Creates a concat with a temporary name
	Creates a mirror with a temporary name
	Creates a stripe with a temporary name
	Creates a volume with a temporary name
	Creates a subvolume with a temporary name

Button	Task
	Starts the Performance Co-Pilot XVM I/O monitor `pmgxvm` on the IRIX server, displaying via X Windows to your local administration station
	Detaches the selected volume elements from their current parents
	Deletes the selected non-slice volume elements or unlabels the selected disks directly, or brings up the appropriate delete task for the selected component

View Menu

Choose what you want to view from the **View** menu:

- Nodes and cluster
- Filesystems
- Cluster volume elements
- Local volume elements
- Disks
- Switches
- Users
- Task privileges

Selecting Items to View or Modify

You can use the following methods to select items:

- Click to select one item at a time
- `Shift`+click to select a block of items

- `Ctrl`+click to toggle the selection of any one item

Another way to select one or more items is to type a name into the **Find** text field and then press `Enter` or click the **Find** button.

Viewing Component Details

To view the details on any component, click its name in the view area; see "Selecting Items to View or Modify" on page 161.

The configuration and status details for the component will appear in the details area to the right. At the bottom of the details area will be the **Applicable Tasks** list, which displays tasks you may wish to launch after evaluating the component's configuration details. To launch a task, click the task name; based on the component selected, default values will appear in the task window.

To see more information about an item in the details area, select its name (which will appear in blue); details will appear in a new window. Terms with glossary definitions also appear in blue.

Performing Tasks

To perform an individual task, do the following:

1. Select the task name from the **Task** menu or click the right mouse button within the view area. For example:

   Task
       > Guided Configuration
           > Set Up a New Cluster

   The task window appears.

   As a shortcut, you can right-click an item in the view area to bring up a list of tasks applicable to that item; information will also be displayed in the details area.

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

   ---

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

Note: In every task, the cluster configuration will not update until you click **OK**.

A dialog box appears confirming the successful completion of the task.

3. Continue launching tasks as needed.

Using Drag-and-Drop for XVM Configuration

The GUI allows you to use drag-and-drop to structure volume topologies and to administer XVM disks.

Caution: Always exercise care when restructuring volume elements with drag-and-drop because data that resides on the volume element can be lost. The GUI attempts to warn the user when it can predict that there is a high likelihood of data loss. However, when a volume is not associated with a mounted filesystem, neither the `xvm` command nor the GUI can determine whether that volume holds important data.

You cannot drag and drop between two GUI windows. You cannot drag and drop between the CXFS Manager and the IRIX Interactive Desktop Personal System Administration windows. You cannot drag and drop items onto shortcut command buttons.

See the *XVM Volume Manager Administrator's Guide* for more information about using drag-and-drop to structure volume topologies and configure disks.

Analyzing I/O Performance with Performance Co-Pilot on an IRIX Node

To analyze performance on an IRIX node, click the button to launch Performance Co-Pilot; see "Shortcuts Using Command Buttons" on page 159. The resulting Performance Co-Pilot window shows all volumes, with colored LEDs indicating read and write I/O activity. Position the cursor over any LED and press the spacebar to view a window showing the value-color legend for the LED and the current value of the read or write rate for the corresponding XVM volume or volume element. Middle-mouse-click any LED to get a menu from which you can launch additional tools to show XVM read and write I/O activity charts and a 3D graphical view of disk activity.

Structuring Volume Topologies

To reconfigure a logical volume, do the following:

- Select the view you want:

  View
  > **> Cluster Volume Elements**

  or

  View
  > **> Local Volume Elements**

- Select a volume element icon
- Drag the icon and drop it on another volume element icon

Icons turn blue as you drag to indicate when it is valid to drop upon them. When you drag, if the mouse cursor reaches the top or the bottom of the view area, the display will scroll automatically.

You can use drag-and-drop to operate on multiple volume elements of different types. For example, you can detach several types of volume elements by selecting items and dragging them to any **Unattached** heading, even if no selected item belongs to that category. You can select multiple items of different types and attach them to a parent. For example, you can select two concats and a stripe and use drag-and-drop to attach them to a parent concat.

You can rename volume elements by clicking a selected (highlighted) volume element and typing a new name into the text field.

Configuring Disks

To label or unlabel disks using drag-and-drop, select the following:

View
> **> Disks**

Select an unlabeled disk then drag and drop it on the **Labeled Disks** heading, or select a labeled disk then drag and drop it on the **Unlabeled Disks** heading.

You can give away a disk using the task menu or drag-and-drop. In the **Disks** view, select a disk and then drag and drop it on the **Cluster Disks** heading.

Note: Giving away a disk presents less risk of data loss than stealing a disk.

You can label a disk by clicking a selected (highlighted) disk and typing a name into the resulting name text field.

For more information, see the *XVM Volume Manager Administrator's Guide*.

Getting More Information

Click blue text to launch tasks or display one of the following:

- Term definitions
- Input instructions
- Item details
- The selected task window

Important GUI and `xvm` Command Differences

When volume elements other than volumes are created or detached, the system automatically creates a volume and a subvolume that are associated with the volume element. You can explicitly name this generated volume, in which case the volume name is stored in label space and persists across machine reboots.

The GUI does not display volumes and subvolumes that were not named explicitly. The GUI displays the children of these volumes and subvolumes as available for use or as unattached. In contrast, the `xvm` command shows all volumes and subvolumes.

The GUI displays filesystems that are on volumes that were not named explicitly, but lists the volumes as **None**. Volumes and subvolumes that the system generated automatically with temporary names are mentioned in the full paths of unattached volume elements (for example, `/vol96/datav`), but the GUI ignores them otherwise.

To reduce the risk of data loss, SGI recommends that you name volumes explicitly when using the GUI. If you have created volumes using the `xvm` command that you did not name explicitly, you can use the `xvm` tool to assign these volumes permanent names before proceeding. This can reduce the risk of data loss.

Key to Icons and States

The following tables show keys to the icons and states used in the CXFS Manager GUI.

Table 9-3 Key to Icons

Icon	Entity
	IRIX node (server-capable administration, client administration, or client-only)
	SGI ProPack for Linux node (server-capable administration, client administration, or client-only)
	AIX, Linux third-party, Mac OS X, Solaris, or Windows node (client-only)
	Cluster
	Expanded tree in view area
	Collapsed tree in view area
	Switch

Icon	Entity
	XVM disk
	Unlabeled disk
	Foreign disk
	Slice
	Volume
	Subvolume
	Concat
	Mirror
	Stripe

Icon	Entity
	Slot
	Local filesystem
	CXFS filesystem
	Copy on write
	Repository
	Snapshot
	User account
	GUI task for which execution privilege may be granted or revoked
	Privileged command executed by a given GUI task

Table 9-4 Key to States

Icon	State
	(grey icon) Inactive, unknown, offline — CXFS services may not be active
	(blue icon) Enabled for mount — CXFS services may not be active
	(blue icon) Online, ready for use, up, or mounted without error
	(green swatch) Open, in use
	(blinking orange arrow) Mirror reviving
	(red icon) Error detected, down or mounted with error

Guided Configuration Tasks

This section discusses the following guided configuration tasks:

- "Set Up an Existing FailSafe Cluster for CXFS with the GUI" on page 170
- "Make Changes to Existing Cluster" on page 171
- "Fix or Upgrade Cluster Nodes" on page 171

Also see "Set Up a New Cluster with the GUI" on page 131, "Set Up a New CXFS Filesystem with the GUI" on page 132, and "Check Cluster Status with the GUI" on page 362. For information about XVM guided configuration tasks, see the *XVM Volume Manager Administrator's Guide*.

Set Up an Existing FailSafe Cluster for CXFS with the GUI

Note: Within the CXFS tasks, you can click any blue text to get more information about that concept or input field. In every task, the cluster configuration will not update until you click **OK**.

The **Set Up an Existing FailSafe Cluster for use with CXFS** task leads you through the steps required to convert existing IRIS FailSafe nodes and cluster to CXFS. It encompasses tasks that are detailed elsewhere. This task appears on the CXFS GUI **only if** you also have FailSafe installed.

There is a single database for FailSafe and CXFS. If a given node applies to both products, ensure that any modifications you make are appropriate for both products.

Do the following:

1. Click **Convert a FailSafe Cluster for use with CXFS**. This will change the cluster type to `CXFS and FailSafe`. See "Convert a FailSafe Cluster for use with CXFS with the GUI" on page 189.
2. Stop high availability (HA) services on the nodes to be converted using the FailSafe GUI. See the *FailSafe Administrator's Guide for SGI InfiniteStorage*.
3. Add the second heartbeat and control NIC (for FailSafe use) to the node definitions using the CXFS GUI. See "Modify a Node Definition with the GUI" on page 182.
4. Click **Convert a FailSafe Node for use with CXFS** to convert the local node (the node to which you are connected). A converted node will be of type `CXFS and FailSafe` or `CXFS`. See "Convert a FailSafe Node for use with CXFS with the GUI" on page 185.
5. Click **Convert a FailSafe Node for use with CXFS** to convert another node. Repeat this step for each node you want to convert.
6. Click **Start CXFS Services**.

Make Changes to Existing Cluster

This task lists different ways to edit an existing cluster. You can make changes while the CXFS services are active, such as changing the way the cluster administrator is notified of events; however, your must first stop CXFS services before testing connectivity. You must unmount a file system before making changes to it.

See the following:

- "Modify a Cluster Definition with the GUI" on page 189
- "Set Up a New CXFS Filesystem with the GUI" on page 132
- "Modify a CXFS Filesystem with the GUI" on page 207
- "Define a Node with the GUI" on page 172
- "Test Node Connectivity with the GUI" on page 187
- "Add or Remove Nodes in the Cluster with the GUI" on page 181

Fix or Upgrade Cluster Nodes

This task leads you through the steps required to remove an administration node from a cluster. It covers the following steps:

- "Stop CXFS Services with the GUI" on page 191.
- Perform the necessary maintenance on the node. Only if required, see "Reset a Node with the GUI " on page 181.
- "Start CXFS Services with the GUI" on page 191.
- Monitor the state of the cluster components in the view area. See "Check Cluster Status with the GUI" on page 362.

When shutting down, resetting, or restarting a CXFS client-only node, do not stop CXFS services on the node. (Stopping CXFS services is more intrusive on other nodes in the cluster because it updates the cluster database. Stopping CXFS services is appropriate only for a CXFS administration node.) Rather, let the CXFS shutdown scripts on the node stop CXFS when the client-only node is shut down or restarted.

Node Tasks with the GUI

This section tells you how to define, modify, delete, display, and reset a node using the GUI.

Note: The **Set Up a New Cluster** guided configuration task leads you through the tasks required to set up the cluster and nodes. See "Set Up a New Cluster with the GUI" on page 131.

Define a Node with the GUI

Note: Within the CXFS tasks, you can click any blue text to get more information about that concept or input field. In every task, the cluster configuration will not update until you click **OK**.

To define a node, do the following:

1. **Hostname**: Enter the hostname of the node you are defining. You can use a simple hostname, such as `lilly`, if it can be resolved by the name server or `/etc/hosts` on all nodes in the cluster; otherwise, use a fully qualified domain name such as `lilly.mycompany.com`. Use the `ping` command to display the fully qualified hostname. Do not enter an IP address.

   If you attempt to define a cluster or other object before the local node has been defined, you will get an error message that says:

   ```
   No nodes are registered on servername. You cannot define a cluster
   until you define the node to which the GUI is connected. To do so,
   click "Continue" to launch the "Set Up a New Cluster" task.
   ```

2. **Logical Name**: Enter the simple hostname (such as `lilly`) or an entirely different name (such as `nodeA`). If you entered in the simple hostname for the **Hostname** field, the same name will be entered into the **Logical Name** field by default. Logical names cannot begin with an underscore (_) or include any whitespace characters, and can be at most 255 characters.

   Note: To rename a node, you must delete it and then define a new node.

   ---

3. **Operating System:** Choose the name of the operating system that is running on the node being defined. Choose **Windows** for Windows 2000, Windows 2003, or Windows XP. Choose **Linux 32** when defining an AMD64/EM64T or Itanium 2 node to ensure that you are given the correct follow-on choices and GUI icon.

   An IRIX node or a Linux node can be a server-capable administration node, a client administration node, or a CXFS client-only node, depending upon the node function selected and the software installed. AIX, Linux third-party, Mac OS X, Solaris, and Windows nodes are always CXFS client-only nodes.

   If you select a fail action that includes reset, you will be given an opportunity to provide reset information on a second page. Any potential metadata server should include reset in its fail action hierarchy.

   You cannot later modify the operating system for a defined node. To change the operating system, you would have to delete the node and then define a new node with the new name.

4. **Node Function**: Select one of the following:

   - **Server-capable Admin** is an IRIX or Linux node on which you will execute cluster administration commands and that you also want to be a CXFS metadata server. (You will use the **Define a CXFS Filesystem** task to define the specific filesystem for which this node can be a metadata servers.) Use this node function only if the node will be a metadata servers. You must install the `cluster_admin` product on this node.

   - **Client Admin** is an IRIX or Linux node on which you will execute cluster administration commands but that you do not want to use as a CXFS metadata server. Use this node function only if the node will run FailSafe but you do not want it to be a metadata server. You must install the `cluster_admin` product on this node.

   - **Client-only** is a node that shares CXFS filesystems but on which you will not execute cluster administration commands and that will not be a CXFS metadata server. Use this node function for all nodes other than those that will be metadata servers, or those that will run FailSafe without being a metadata server. You must install the product on this node. This node can run AIX, IRIX, HP-UX, Linux third-party, SGI ProPack for Linux, Mac OS X, Solaris, or Windows. (Nodes other than IRIX and SGI ProPack for Linux are **required** to be client-only nodes.)

5. **Networks for Incoming Cluster Messages**: Do the following:

- **Network**: Enter the IP address or hostname of the NIC. (The hostname must be resolved in the `/etc/hosts` file.) The priorities of the NICs must be the same for each node in the cluster. For information about why a private network is required, see "Private Network" on page 19.

  FailSafe requires at least two NICs.

- **Messages to Accept**: Select **Heartbeat and Control**.

  You can use the **None** setting if you want to temporarily define a NIC but do not want it to accept messages. For more information, see "Cluster Environment" on page 9.

- Click **Add** to add the NIC to the list.

  If you later want to modify the NIC, click the NIC in the list to select it, then click **Modify**.

  To delete a NIC from the list, click the NIC in the list to select it, then click **Delete**.

By default, the priority `1` NICs are used as the private network; they must be on the same subnet. To allow one network to fail over to another, you must group the NICs into networks manually by using the `cmgr` command. See "Define a Cluster with `cmgr`" on page 246 and "Modify a Cluster with `cmgr`" on page 250.

6. **Node ID**: (*Optional for administration nodes*) An integer in the range 1 through 32767 that is unique among the nodes in the pool. If you do not specify a number for an administration node, CXFS will calculate an ID for you.

   For administration nodes, the default ID is a 5-digit number based on the machine's serial number and other machine-specific information; it is not sequential. For client-only nodes, you must supply the node ID.

   You must not change the node ID number after the node has been defined. (There is no default CXFS tiebreaker; for more information, see "CXFS Kernel Membership, Quorum, and Tiebreaker" on page 462.)

7. **Partition ID**: *(Optional)* Uniquely defines a partition in a partitioned Origin 3000 system. If your system is not partitioned, leave this field empty. Use the IRIX `mkpart` command or the Linux `proc` command to determine the partition ID value (see below).

   Click **Next** to move to the next screen.

8. **Fail Action:** The set of actions that determine what happens to a failed node. The second action will be followed only if the first action fails; the third action will be followed only if the first and second fail.

   The available actions depend upon the node's operating system:

   - **Fence:** Disables access to the storage area network (SAN) from the problem node. This action is available for all nodes.
   - **FenceReset:** Disables access to the SAN from the problem node and then, if the node is successfully fenced, **also** performs an asynchronous reset of the node (according to the chosen reset method). Recovery begins without waiting for reset acknowledgment. This action is available only for administration nodes with system controllers; see "Requirements" on page 37.

   Note: A potential metadata server should also include **Reset** in its fail action hierarchy.

   - **Reset:** Performs a system reset (according to the chosen reset method) via a serial connection to the system controller. This action is available only for administration nodes with system controllers. A potential metadata server should include **Reset** in its fail action hierarchy.
   - **Shutdown:** Tells the other nodes in the cluster to wait before reforming the CXFS kernel membership. (Whether this action is set or not, the local node will automatically attempt to shut down CXFS services on the node in response to a loss of CXFS kernel membership quorum.) The surviving cluster delays the beginning of recovery to allow the node time to complete the shutdown. This action is available for all nodes. In the GUI, this action is required for all nodes.

     Caution: If you have a cluster with an even number of server-capable nodes and no tiebreaker: to avoid a split-brain scenario, you should not use the **Shutdown** setting on any server-capable node. To do this, you must define or modify the node with `cmgr`. See "Issues with the Shutdown Fail Action Setting" on page 29 and "Define a Node with `cmgr`" on page 224.

   On nodes without system controllers, your only choice for data integrity protection is I/O fencing.

Note: A switch is mandatory to support I/O fencing; **therefore, nodes without system controllers require a switch**. See the release notes for the supported switches.

On nodes with system controllers, you would want to use I/O fencing for data integrity protection if the node is configured as a client-only node and CXFS is just a part of what the node is doing, and you prefer losing access to CXFS to having the system rebooted; for example, for a big compute server that is also a CXFS client. SGI recommends that you use reset when the node is a potential metadata server.

The default fail action hierarchy for IRIX and Linux nodes is **Reset, Shutdown**. The default for nodes running other supported operating systems is **Shutdown**. SGI recommends that you choose a failure hierarchy other than the default for systems without system controllers; see "Requirements" on page 37.

Click **Next** to move to the next screen.

9. If you have chosen a failure hierarchy that includes **Reset** or **FenceReset**, provide the following information.

   - This node:

     – **Port Type:** select one of the following:

       - **L1** (Origin/Onyx 300/350, Origin/Onyx 3200C and Tezro)
       - **L2** (Altix 350, Altix 3300/3700, Prism, Origin 3400/3800, Onyx 3000, Origin 300 with NUMAlink module)
       - **MSC** (Origin 200, Onyx2 Deskside, SGI 2100/2200 deskside systems)
       - **MMSC** (Rackmount SGI 2400/2800, Onyx2)

       Note: If you have an Altix system that uses an integrated L2 (such as a NUMAlink 4 R-brick) or SGI Altix 3000 Bx2 systems, you must use the `cmgr` command to configure system reset. See "Define a Node with `cmgr`" on page 224.

     – **Reset Method**: The type of reset to be performed:

       - **Power Cycle** shuts off power to the node and then restarts it

    - **Reset** simulates the pressing of the reset button on the front of the machine
    - **NMI** (nonmaskable interrupt) performs a core-dump of the operating system kernel, which may be useful when debugging a faulty machine

  – **Port Password**: The password for the system controller port, **not** the node's `root` password or PROM password. On some nodes, the system administrator may not have set this password. If you wish to set or change the system controller port password, consult the hardware manual for your node.

  – **Temporarily Disable Port**: If you want to provide reset information now but do not want to allow the reset capability at this time, check this box. If this box is checked, CXFS cannot reset the node.

- Owner (node that sends the reset command):

  – **Logical Name**: Name of the node that sends the reset command. Serial cables must physically connect the node being defined and the owner node through the system controller port. At run time, the node must be defined in the CXFS pool.

    You can select a logical name from the pull-down list or enter the logical name of a node that is not yet defined. However, you must define the node in CXFS before you run the node connectivity diagnostics task.

  – **TTY Device**: Name of the terminal port (TTY) on the owner node to which the system controller is connected, such as `/dev/ttyd2`. The other end of the cable connects to this node's system controller port, so the node can be controlled remotely by the other node.

10. Click **OK**.

Note: Do not add a second node until the first node icon appears in the view area. The entire cluster status information is sent to each CXFS administration node each time a change is made to the cluster database; therefore, the more CXFS administration nodes in a configuration, the longer it will take.

You can use the IRIX `mkpart` command to determine the partition ID:

- The `-n` option lists the partition ID (which is 0 if the system is not partitioned).
- The `-l` option lists the bricks in the various partitions (use *rack#.slot#* format in the GUI).

  On Linux, you can find the partition ID by reading the `proc` file. For example:

  ```
  [root@linux64 root]# cat /proc/sgi_sn/partition_id
  0
  ```

  The `0` indicates that the system is not partitioned. If the system is partitioned, the number of partitions (such as `1`, `2`, etc.) is displayed.

For example (output truncated here for readability):

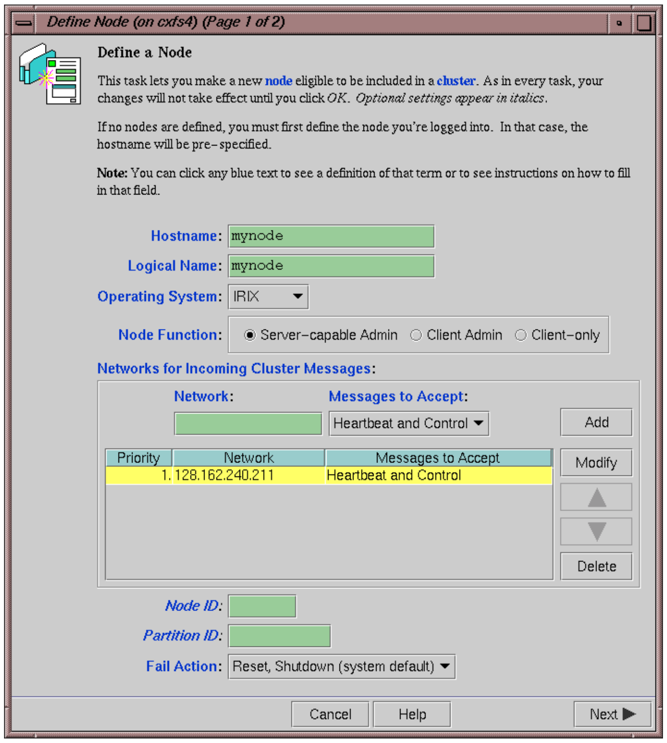
```
# mkpart -n
Partition id = 1
# mkpart -l
partition: 3 = brick: 003c10 003c13 003c16 003c21 ...
partition: 1 = brick: 001c10 001c13 001c16 001c21 ...
```

You could enter one of the following for the **Partition ID** field:

```
1
001.10
```

Examples of Defining a Node with the GUI

The following figures show an example of defining a new node.



Define Node (on cxfs4) (Page 1 of 2)

Define a Node

This task lets you make a new node eligible to be included in a cluster. As in every task, your changes will not take effect until you click *OK*. *Optional settings appear in italics.*

If no nodes are defined, you must first define the node you're logged into. In that case, the hostname will be pre-specified.

Note: You can click any blue text to see a definition of that term or to see instructions on how to fill in that field.

Hostname: mynode

Logical Name: mynode

Operating System: IRIX

Node Function: [✓ Server-capable Admin] Client Admin Client-only

Networks for Incoming Cluster Messages:

Network:

Messages to Accept: Heartbeat and Control

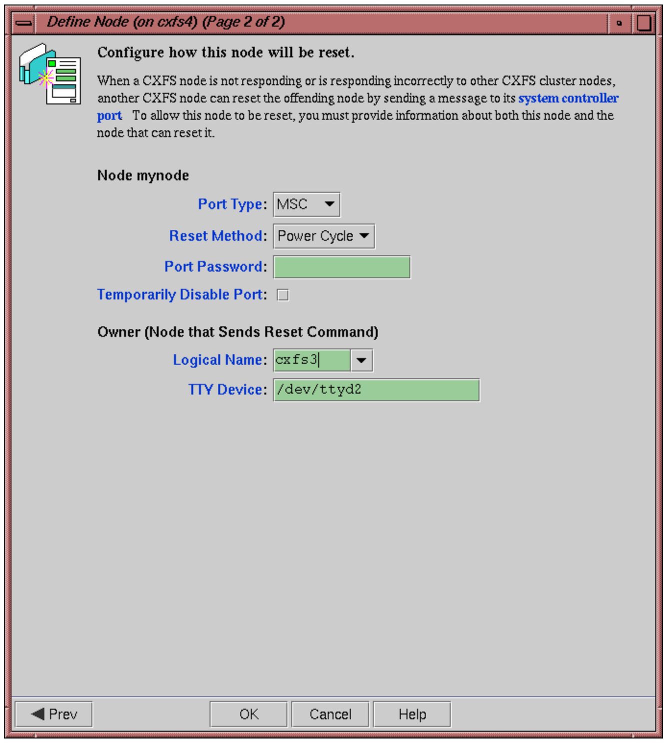
Priority	Network	Messages to Accept
1.	128.162.240.211	Heartbeat and Control

Node ID:

Partition ID:

Fail Action: Reset, Shutdown (system default)

Figure 9-3 Example Node Definition



Configure how this node will be reset.

When a CXFS node is not responding or is responding incorrectly to other CXFS cluster nodes, another CXFS node can reset the offending node by sending a message to its system controller port. To allow this node to be reset, you must provide information about both this node and the node that can reset it.

Node mynode

Port Type: MSC

Reset Method: Power Cycle

Port Password:

Temporarily Disable Port: ☐

Owner (Node that Sends Reset Command)

Logical Name: cxfs3

TTY Device: /dev/ttyd2

Figure 9-4 Example System Reset Settings

Add or Remove Nodes in the Cluster with the GUI

After you have added nodes to the pool and defined the cluster, you can indicate which nodes to include in the cluster.

Note: Do not add or remove nodes until the cluster icon appears in the view area; set the **View** selection to **Nodes and Cluster**.

Do the following:

1. Add or remove the desired nodes:
   - To add a node, select its logical name from the **Available Nodes** pull-down menu and click **Add**. The node name will appear in the **Nodes to Go into Cluster** list. To select all of the available nodes, click **Add All**.
   - To delete a node, click its logical name in the **Nodes to Go into Cluster** screen. (The logical name will be highlighted.) Then click **Remove**.
2. Click **OK**.

Reset a Node with the GUI

You can use the GUI to reset IRIX or Linux nodes in a cluster. This sends a reset command to the system controller port on the specified node. When the node is reset, other nodes in the cluster will detect the change and remove the node from the active cluster. When the node reboots, it will rejoin the CXFS kernel membership.

To reset a node, do the following:

1. **Node to Reset:** Choose the node to be reset from the pull-down list.
2. Click **OK**.

Modify a Node Definition with the GUI

To rename a node or change its operating system, you must delete it and then define a new node.

To modify other information about a node, do the following:

1. **Logical Name**: Choose the logical name of the node from the pull-down list. After you do this, information for this node will be filled into the various fields.

2. **Networks for Incoming Cluster Messages**: The priorities of the NICs must be the same for each node in the cluster.

   - **Network**: To add a NIC for incoming cluster messages, enter the IP address or hostname into the **Network** text field and click **Add**.

   - To modify a NIC that is already in the list, click the network in the list in order to select it. Then click **Modify**. This moves the NIC out of the list and into the text entry area. You can then change it. To add it back into the list, click **Add**.

   - To delete a NIC, click the NIC in the priority list in order to select it. Then click **Delete**.

   - To change the priority of a NIC, click the NIC in the priority list in order to select it. Then click the up and down arrows in order to move it to a different position in the list.

     You can use the **None** setting if you want to temporarily define a NIC but do not want it to accept messages. For more information, see "Cluster Environment" on page 9.

   By default, the priority `1` NICs are used as the private network; they must be on the same subnet. To allow the one network to fail over to another, you must group the NICs into networks manually by using the `cmgr` command. See "Modify a Cluster with `cmgr`" on page 250.

   Click **Next** to move to the next page.

3. **Partition ID**: *(Optional)* Uniquely defines a partition in a partitioned Origin 3000 system. If your system is not partitioned, leave this field empty. You can use the IRIX `mkpart` command or the Linux `proc` command to determine the partition ID value; see below.

4. **Fail Action:** Specify the set of actions that determines what happens to a failed node: the second action will be followed only if the first action fails; the third action will be followed only if the first and second fail.

   The available actions depend upon the node's operating system:

   - **Fence:** Disables access to the storage area network (SAN) from the problem node. Fencing provides faster recovery of the CXFS kernel membership than reset. This action is available for all nodes.
   - **FenceReset:** Disables access to the SAN from the problem node and then, if the node is successfully fenced, **also** performs an asynchronous reset of the node. Recovery begins without waiting for reset acknowledgment. This action is available for nodes with system controllers; see "Requirements" on page 37.

     Note: A potential metadata server should also include **Reset** in its fail action hierarchy.

   - **Reset:** Performs a system reset via a serial connection to the system controller. This action is available for nodes with system controllers. A potential metadata server should include **Reset** in its fail action hierarchy.
   - **Shutdown:** Tells the other nodes in the cluster to wait before reforming the cluster. (Whether this action is set or not, the local node will automatically attempt to stop CXFS services on the node in response to a loss of CXFS kernel membership quorum.) The surviving cluster delays the beginning of recovery to allow the node time to complete the shutdown. This action is available for all nodes. In the GUI, this action is required for all nodes.

     Caution: If you have a cluster with an even number of server-capable nodes and no tiebreaker: to avoid a split-brain scenario, you should not use the **Shutdown** setting on any server-capable node. To do this, you must define or modify the node with `cmgr`. See "Issues with the Shutdown Fail Action Setting" on page 29 and "Modify a Node with `cmgr`" on page 234.

   The default fail action hierarchy for IRIX or Linux nodes is **Reset, Shutdown**. The default for other nodes is **Shutdown**.

5. If you have chosen a failure hierarchy that includes **Reset** or **FenceReset**, provide the following information.

- This node:
  - **Port Type:** select one of the following:
    - **L1** (Origin/Onyx 300/350, Origin/Onyx 3200C and Tezro)
    - **L2** (Altix 350, Altix 3300/3700, Prism, Origin 3400/3800, Onyx 3000, Origin 300 with NUMAlink module)
    - **MSC** (Origin 200, Onyx2 Deskside, SGI 2100/2200 deskside systems)
    - **MMSC** (Rackmount SGI 2400/2800, Onyx2).
  - **Reset Method**: The type of reset to be performed:
    - **Power Cycle** shuts off power to the node and then restarts it
    - **Reset** simulates the pressing of the reset button on the front of the machine
    - **NMI** (nonmaskable interrupt) performs a core-dump of the operating system kernel, which may be useful when debugging a faulty machine
  - **Port Password**: The password for the system controller port, **not** the node's `root` password or PROM password. On some nodes, the system administrator may not have set this password. If you wish to set or change the system controller port password, consult the hardware manual for your node.
  - **Temporarily Disable Port**: If you want to provide reset information now but do not want to allow the reset capability at this time, check this box. If this box is checked, CXFS cannot reset the node.
- Owner (node that sends the reset command):
  - **Logical Name**: Name of the node that sends the reset command. Serial cables must physically connect the node being defined and the owner node through the system controller port. At run time, the node must be defined in the CXFS pool.

    You can select a logical name from the pull-down list or enter the logical name of a node that is not yet defined. However, you must define the node in CXFS before you run the node connectivity diagnostics task.
  - **TTY Device**: Name of the terminal port (TTY) on the owner node to which the system controller is connected, such as `/dev/ttyd2`. The other end of

the cable connects to this node's system controller port, so the node can be controlled remotely by the other node.

6. Click **OK**.

You can use the IRIX `mkpart` command to determine the partition ID value:

- The `-n` option lists the partition ID (which is 0 if the system is not partitioned).
- The `-l` option lists the bricks in the various partitions (use *rack#.slot#* format in the GUI).

For example (output truncated here for readability):

```
# mkpart -n
Partition id = 1
# mkpart -l
partition: 3 = brick: 003c10 003c13 003c16 003c21 ...
partition: 1 = brick: 001c10 001c13 001c16 001c21 ...
```

You could enter one of the following for the **Partition ID** field:

```
1
001.10
```

On Linux, you can find the partition ID by reading the `proc` file. For example:

```
[root@linux64 root]# cat /proc/sgi_sn/partition_id
0
```

The `0` indicates that the system is not partitioned. If the system is partitioned, the number of partitions (such as `1`, `2`, etc.) is displayed.

Convert a FailSafe Node for use with CXFS with the GUI

This task appears on the CXFS GUI only if you also have FailSafe installed. It applies only to CXFS administration nodes.

You can convert an existing FailSafe node (of type `FailSafe`) to either of the following types:

- `CXFS and FailSafe`
- `CXFS`

Do the following:

1. Stop HA services on the node to be converted using the FailSafe GUI. See the *FailSafe Administrator's Guide for SGI InfiniteStorage*.
2. Add the second **Heartbeat and Control** NIC (for FailSafe use) to the node definition using the CXFS GUI. See "Modify a Node Definition with the GUI" on page 182.
3. Enter the following information:
   - **Logical Name**: Choose the logical name of the node from the pull-down list.
   - **Keep FailSafe Settings:**
     - To convert to type `CXFS and FailSafe`, click the checkbox
     - To convert to type `CXFS`, leave the checkbox blank
   - Click **OK**.

Note: If you want to rename a node, you must delete it and then define a new node.

To change other parameters, see "Modify a Node Definition with the GUI" on page 182. Ensure that modifications you make are appropriate for both FailSafe and CXFS.

To convert a CXFS node so that it applies to FailSafe, use the `cmgr` command or the FailSafe GUI. For information about the FailSafe GUI, see the *FailSafe Administrator's Guide for SGI InfiniteStorage*.

Delete a Node with the GUI

You must remove a node from a cluster before you can delete the node from the pool. For information, see "Modify a Cluster Definition with the GUI" on page 189.

To delete a node, do the following:

1. **Node to Delete**: Select the logical name of the node to be deleted from the pull-down list.
2. Click **OK**.

Test Node Connectivity with the GUI

The **Test Node Connectivity** screen requires `rsh` access between hosts. The `/.rhosts` file must contain the hosts and local host between which you want to test connectivity.

To test connectivity, do the following from the CXFS Manager:

1. Choose whether to test by network or serial connectivity by clicking the appropriate radio button.
2. Choose a node to be tested from the pull-down list and add it to the test list by clicking **Add**.

   To delete a node from the list of nodes to be tested, click the logical name to select it and then click **Delete**.
3. To start the tests, click **Start Tests**. To stop the tests, click **Stop Tests**.
4. To run another test, click **Clear Output** to clear the status screen and start over with step 3.
5. To exit from the window, click **Close**.

Display a Node with the GUI

After you define nodes, you can use the **View** selection in the view area to display the following:

- **Nodes and Cluster** shows the nodes that are defined as part of a cluster or as part of the pool (but not in the cluster)

Click any name or icon to view detailed status and configuration information.

Cluster Tasks with the GUI

The following tasks let you define, modify, delete, and display a cluster.

Note: The **Set Up a New Cluster** guided configuration task leads you through the tasks required to set up the cluster and nodes. See "Set Up a New Cluster with the GUI" on page 131.

Define a Cluster with the GUI

A *cluster* is a collection of nodes coupled to each other by a private network. A cluster is identified by a simple name. A given node may be a member of only one cluster.

To define a cluster, do the following:

1. Enter the following information:

   - **Cluster Name**: The logical name of the cluster. Logical names cannot begin with an underscore (_) or include any whitespace characters, and can be at most 255 characters. Clusters must have unique names.

   - **Cluster ID**: A unique number within your network in the range 1 through 128. The cluster ID is used by the operating system kernel to make sure that it does not accept cluster information from any other cluster that may be on the network. The kernel does not use the database for communication, so it requires the cluster ID in order to verify cluster communications. This information in the kernel cannot be changed after it has been initialized; therefore, you must not change a cluster ID after the cluster has been defined. Clusters must have unique IDs.

   - **Cluster Mode**: Usually, you should set the cluster to the default `Normal` mode.

     Setting the mode to `Experimental` turns off heartbeating in the CXFS kernel membership code so that you can debug the cluster without causing node failures. For example, this can be useful if you just want to disconnect the network for a short time (provided that there is no other cluster networking activity, which will also detect a failure even if there is no heartbeating) or if you want to enter the kernel debugger (which stops heartbeat) on a CXFS node. You should only use `Experimental` mode when debugging.

   - **Notify Administrator** (of cluster and node status changes):

     – **By e-mail**: This choice requires that you specify the e-mail program (`/usr/sbin/Mail` by default) and the e-mail addresses of those to be identified. To specify multiple addresses, separate them with commas. CXFS will send e-mail to the addresses whenever the status changes for a node or cluster. If you do not specify an address, notification will not be sent.

     – **By other command**: This choice requires that you specify the command to be run whenever the status changes for a node or cluster.

     – **Never**: This choice specifies that notification is not sent.

2. Click **OK**.

Modify a Cluster Definition with the GUI

To change how the cluster administrator is notified of changes in the cluster's state, do the following:

1. Enter the following information:
   - **Cluster Name**: Choose from the pull-down list.
   - **Cluster Mode**: Usually, you should set the cluster to the default `Normal` mode. See "Define a Cluster with the GUI" on page 188, for information about `Experimental` mode.
   - **Notify Administrator**: Select the desired notification. For more information, see "Define a Cluster with the GUI" on page 188.
2. Click **OK**.

To modify the nodes that make up a cluster, see "Add or Remove Nodes in the Cluster with the GUI" on page 181.

Note: If you want to rename a cluster, you must delete it and then define a new cluster. If you have started CXFS services on the node, you must either reboot it or reuse the cluster ID number when renaming the cluster.

However, be aware that if you already have CXFS filesystems defined and then rename the cluster, CXFS will not be able to mount the filesystems. For more information, see "Cannot Mount Filesystems" on page 408.

Convert a FailSafe Cluster for use with CXFS with the GUI

This task appears on the CXFS GUI only if you also have FailSafe installed.

To convert the information from an existing IRIS FailSafe cluster (that is, of type `FailSafe`) to create a cluster that applies to CXFS (that is, of type `CXFS and FailSafe` or of type `CXFS`), do the following:

1. Enter the following information:
   - **Cluster Name**: Choose from the pull-down list.

- **Cluster ID**: Enter a unique number within your network in the range 1 through 128. The cluster ID is used by the operating system kernel to make sure that it does not accept cluster information from any other cluster that may be on the network. The kernel does not use the database for communication, so it requires the cluster ID in order to verify cluster communications. This information in the kernel cannot be changed after it has been initialized; therefore, you must not change a cluster ID after the cluster has been defined.

2. Click **OK**.

The cluster will apply to both IRIS FailSafe and CXFS. To modify the nodes that make up a cluster, see "Add or Remove Nodes in the Cluster with the GUI" on page 181.

Note: If you want to rename a cluster, you must delete it and then define a new cluster.

Delete a Cluster with the GUI

You cannot delete a cluster that contains nodes; you must move those nodes out of the cluster first. For information, see "Add or Remove Nodes in the Cluster with the GUI" on page 181.

To delete a cluster, do the following:

1. **Cluster to Delete**: The name of the cluster is selected for you.
2. Click **OK**.

Display a Cluster with the GUI

From the **View** selection, you can choose elements to examine. To view details of the cluster, click the cluster name or icon; status and configuration information will appear in the details area on the right.

Cluster Services Tasks with the GUI

The following tasks let you start and stop CXFS services and set the log configuration.

Start CXFS Services with the GUI

To start CXFS services, and set the configuration to automatically restart CXFS services whenever the system is rebooted, do the following:

1. **Node(s) to Activate**: Select `All Nodes` or the individual node on which you want to start CXFS services.
2. Click **OK**.

Stop CXFS Services with the GUI

When CXFS services are stopped on a node, filesystems are automatically unmounted from that node.

To stop CXFS services temporarily (that is, allowing them to restart with a reboot if so configured), use the following command line in a shell window outside of the GUI:

- IRIX:

```
# /etc/init.d/cluster stop(on an admin node)
# /etc/init.d/cxfs stop (on an admin node)

# /etc/init.d/cxfs_client stop (on a client-only node)
```

- Linux:

```
# /etc/init.d/cxfs_cluster stop (on an admin node)
# /etc/init.d/cxfs stop (on an admin node)

# /etc/init.d/cxfs_client stop (on a client-only node)
```

You can stop CXFS on a specified node or cluster, and prevent CXFS services from being restarted by a reboot, by performing the following steps:

Note: If you stop CXFS services using this method, they will not restart when the node is rebooted.

1. Enter the following information:
   - **Force**: If you want to forcibly stop CXFS services even if there are errors (which would normally prevent the stop operation), click the **Force** checkbox.
   - **Node(s) to Deactivate**: Select `All Nodes` or the individual node on which you want to stop CXFS services.

     If you stop CXFS services on one node, that node will no longer have access to any filesystems. If that node was acting as the metadata server for a filesystem, another node in the list of potential metadata servers will be chosen. Clients of the filesystem will experience a delay during this process.
2. Click **OK**. It may take a few minutes to complete the process.

After you have stopped CXFS services on a node, the node is no longer an active member of the cluster. CXFS services will not be restarted when the system reboots.

Caution: You should stop CXFS services before using the `shutdown` or `reboot` commands. If you execute `shutdown` or `reboot` when CXFS services are active, the remaining nodes in the cluster will view it as a node failure and be forced to run recovery against that node.

Set Tiebreaker Node with the GUI

A *CXFS tiebreaker node* determines whether a CXFS kernel membership quorum is maintained when exactly half of the server-capable nodes are up and can communicate with each other. There is no default CXFS tiebreaker. For more information, see "CXFS Kernel Membership, Quorum, and Tiebreaker" on page 462.

Caution: If one of the server-capable nodes is the CXFS tiebreaker in a two server-capable cluster, failure of that node or stopping the CXFS services on that node will result in a cluster-wide forced shutdown. Therefore SGI recommends that you use client-only nodes as tiebreakers so that either server could fail but the cluster would remain operational via the other server.

To ensure data integrity, SGI recommends that you use system reset for all potential metadata servers and reset or or I/O fencing for all client-only nodes; reset is required for IRIS FailSafe.

The current CXFS tiebreaker node is shown in the detailed view of the cluster.

To set the CXFS tiebreaker node, do the following:

1. **Tie-Breaker Node**: Select the desired node from the list. If there currently is a CXFS tiebreaker, it is selected by default.

   To unset the CXFS tiebreaker node, select `None`.

2. Click **OK**.

Set Log Configuration with the GUI

CXFS maintains logs for each of the CXFS daemons. CXFS logs both normal operations and critical errors to individual log files for each log group and the system log file:

- IRIX: `/var/adm/SYSLOG`
- Linux: `/var/log/messages`

You can customize the logs according to the level of logging you wish to maintain.

Caution: Do not change the names of the log files. If you change the names, errors can occur.

When you define a log configuration, you specify the following information:

- **Log Group**: A set of one or more CXFS processes that use the same log configuration. A log group usually corresponds to one CXFS daemon, such as `crsd`.
- **Log Level**: A number controlling the amount of log messages that CXFS will write into an associated log group's log file.
- **Log File**: The file in which to log messages.

See also "Status in Log Files" on page 360.

Display Log Group Definitions with the GUI

To display log group definitions, do the following:

1. **Log Group**: Choose the log group to display from the menu.

   The current log level and log file for that log group will be displayed in the task window, where you can change those settings if you desire.

2. Click **OK**.

Configure Log Groups with the GUI

To configure a log group, do the following in the **Set Log Configuration** task:

1. Enter the appropriate information:

   - **Log Group**: Select the log group from the pull-down list. A *log group* is a set of processes that log to the same log file according to the same logging configuration. Each CXFS daemon creates a log group. Settings apply to all nodes in the pool for the `cli` and `crsd` log groups, and to all nodes in the cluster for the `clconfd` and `diags` log groups.
   - **Log Level**: Select the log level, which specifies the amount of logging.

   Caution: The **Default** log level is quite verbose; using it could cause space issues on your disk. You may wish to select a lower log level. Also see "Log File Management" on page 307, "`cad.options` on CXFS Administration Nodes" on page 96, and "`fs2d.options` on CXFS Administration Nodes" on page 98.

The values are as follows:

– **Off** gives no logging

– **Minimal** logs notifications of critical errors and normal operation (these messages are also logged to the IRIX `/var/adm/SYSLOG` and Linux `/var/log/messages` file)

– **Info** logs **Minimal** notifications plus warnings

– **Default** logs all **Info** messages plus additional notifications

– **Debug 0** through **Debug 9** log increasingly more debug information, including data structures

The `cmgr` command uses a set of numbers to indicate these log levels. See "Configure Log Groups with `cmgr`" on page 257.

2. **Log File**: Do not change this value.
3. Click **OK**.

Revoke Membership of the Local Node with the GUI

You should revoke CXFS kernel membership of the local node only in the case of error, such as when you need to perform a forced CXFS shutdown (see "Shutdown of the Database and CXFS" on page 300).

To revoke CXFS kernel membership for the local node, do the following:

1. **Local Node:** Verify the name of the local node, which will be displayed in the pop-up window.
2. Click **OK** to complete the task.

This result of this task will be considered as a node failure by the rest of the cluster. The rest of the cluster may then fail due to a loss of CXFS kernel membership quorum, or it may decide to reset the failed node. To avoid the reset, you can modify the node definition to disable the system controller status.

Allow Membership of the Local Node with the GUI

You must allow CXFS kernel membership for the local node (the node to which the GUI is connected) after fixing the problems that required a forced CXFS shutdown;

doing so allows the node to reapply for CXFS kernel membership in the cluster. A forced CXFS shutdown can be performed manually or can be triggered by the kernel. For more information, see "Shutdown of the Database and CXFS" on page 300.

You must actively allow CXFS kernel membership of the local node in the following situations:

- After a manual revocation as in "Revoke Membership of the Local Node with the GUI" on page 195.
- When instructed to by an error message on the console or in system log file:
  - IRIX: `/var/adm/SYSLOG`
  - Linux: `/var/log/messages`
- After a kernel-triggered revocation. This situation is indicated by the following message in system log file (IRIX `/var/adm/SYSLOG` or Linux `/var/log/messages`):

  ```
  Membership lost - withdrawing from cluster
  ```

To allow CXFS kernel membership for the local node, do the following:

1. **Local Node:** Verify the name of the local node, which will be displayed in the pop-up window.
2. Click **OK** to complete the task.

Switches and I/O Fencing Tasks with the GUI

The following tasks let you configure switches for the CXFS cluster and perform I/O fencing. See the release notes for supported switches.

Note: Nodes without system controllers require I/O fencing to protect data integrity.

Define a Switch with the GUI

This task lets you define a new Brocade switch to support I/O fencing in a cluster.

Note: To define a switch other than a Brocade switch, such as a QLogic switch, you must use the `hafence`(1M) command. (You cannot use the GUI to completely define switches other than Brocade.) See "Configuring Switches Other than Brocade" on page 288.

Do the following:

1. Enter the following information:

   - **Switch Name:** Enter the hostname of the switch; this is used to determine the IP address of the switch.
   - **Username:** Enter the user name to use when sending a `telnet` message to the switch. By default, this value is `admin`.
   - **Password:** Enter the password for the specified **Username** field.
   - **Mask:** Enter a hexadecimal string that represents the list of ports in the switch that will not be fenced. Ports are numbered from zero. If a given bit has a binary value of 0, the port that corresponds to that bit is eligible for fencing operations; if 1, then the port that corresponds to that bit will always be excluded from any fencing operations. For example, Figure 9-5 shows that a mask of `FF03` for a 16-port switch indicates that only ports 2–7 are eligible for fencing (because they have binary values of 0). Similarly, it shows that a mask of `A4` for an 8-port switch allows fencing only on ports 0, 1, 3, 4, and 6 (the port numbers corresponding to binary 0) — ports 2, 5, and 7 will never be fenced (the port numbers corresponding to the nonzero value).

16-port Switch (1= never fence, 0= may fence)

Port #	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0
Binary	1 1 1 1	1 1 1 1	0 0 0 0	0 0 1 1
Hexadecimal	F	F	0	3

8-port Switch

Port #	7 6 5 4	3 2 1 0
Binary	1 0 1 0	0 1 0 0
Hexadecimal	A	4

Figure 9-5 Bit Mask Representation for I/O Fencing

Note: You can only mask ports 0 through 63.

CXFS administration nodes automatically discover the available HBAs and, when fencing is triggered, will fence off all of the Fibre Channel HBAs when the **Fence** or **FenceReset** fail action is selected. However, masked HBAs will not be fenced. Masking allows you to prevent the fencing of devices that are attached to the SAN but are not shared with the cluster, to ensure that they remain available regardless of CXFS status. You would want to mask HBAs used for access to tape storage, or HBAs that are only ever used to access local (nonclustered) devices.

2. Click **OK** to complete the task.

Modify a Switch Definition with the GUI

This task lets you modify an existing Brocade switch definition.

Note: To modify the definition of another type of switch, such as QLogic, you must use the `hafence`(1M) command. See "Configuring Switches Other than Brocade" on page 288.

Do the following:

1. Enter the following information:
   - **Switch Name:** Select the hostname of the switch to be modified.
   - **Username:** Enter the user name to use when sending a `telnet` message to the switch. By default, this value is `admin`.
   - **Password:** Enter the password for the specified **Username** field.
   - **Mask:** Enter a hexadecimal string that represents the list of ports in the switch that will not be fenced. For more information, see "Define a Switch with the GUI" on page 196.
2. Click **OK** to complete the task.

Update Switch Port Information with the GUI

This task lets you update the mappings between the host bus adapters (HBAs) and switch ports. You should run this command if you reconfigure any switch or add ports. Click **OK** to complete the task.

Delete a Switch Definition with the GUI

This task lets you delete an existing switch definition. Do the following:

1. **Switch Name:** Select the hostname of the Fibre Channel switch to be deleted.
2. Click **OK** to complete the task.

Raise the I/O Fence for a Node with the GUI

This task lets you raise the I/O fence for a node. Raising an I/O fence isolates the node from the SAN; CXFS sends a messages via the `telnet` protocol to the switch and disables the port. After the node is isolated, it cannot corrupt data in the shared CXFS filesystem.

Do the following:

1. **Raise Fence for Node:** Select the name of the node you want to isolate. Only nodes that have been configured with a **Fence** or **FenceReset** fail action can be selected.
2. Click **OK** to complete the task.

Lower the I/O Fence for a Node with the GUI

This task lets you lower the I/O fence for a given node by reenabling the port. Lowering an I/O fence allows the node to reconnect to the SAN and access the shared CXFS filesystem.

Do the following:

1. **Lower Fence for Node:** Select the node you want to reconnect. Only nodes that have been configured with a **Fence** or **FenceReset** fail action can be selected.
2. Click **OK** to complete the task.

Filesystem Tasks with the GUI

The following tasks let you configure CXFS filesystems as shared XVM volumes. These shared volumes can be directly accessed by all nodes in a CXFS cluster. Each volume is identified by its device name. Each volume must have the same mount point on every node in the cluster.

Note: The **Set Up a New CXFS Filesystem** guided configuration task leads you through the steps required to set up a new CXFS filesystem. See "Set Up a New CXFS Filesystem with the GUI" on page 132.

Make Filesystems with the GUI

This task lets you create a filesystem on a volume that is online but not open. To create filesystems on multiple volume elements, use the **Browse** button.

Caution: Clicking OK will erase all data that exists on the target volume.

To make a filesystem, do the following:

1. Enter the following information:

   - **Domain**: Select the domain that will own the volume element to be created. Choose **Local** if the volume element or disk is defined for use only on the node to which the GUI is connected, or choose **Cluster** if it is defined for use on multiple nodes in the cluster.

   - **Volume Element**: Select the volumes on which to create the filesystem or select the volume elements whose parent volumes will be used for the filesystems. The menu lists only those volume elements that are available. (When volume elements other than volumes are created or detached, the system automatically creates a volume and a subvolume that are associated with the volume element. If you did not explicitly name an automatically generated volume, the GUI will display its children only.)

   - **Specify Sizes**: Check this box to modify the default options for the filesystem, including data region size, log size, and real-time section size.

     By default, the filesystem will be created with the data region size equal to the size of the data subvolume. If the volume contains a log subvolume, the log size will be set to the size of the log subvolume. If the volume contains a real-time subvolume, the real-time section size will be set to the size of the real-time subvolume.

2. If you checked the **Specify Sizes** box, click **Next** to move to page 2. On page 2, enter the following information. For more information about these fields, see the IRIX `mkfs_xfs` or Linux `mkfs.xfs` man page.

   - **Block Size**: Select the fundamental block size of the filesystem in bytes.

   - **Directory Block Size**: Select the size of the naming (directory) area of the filesystem in bytes.

   - **Inode Size**: Enter the number of blocks to be used for inode allocation, in bytes. The inode size cannot exceed one half of the **Block Size** value.

- **Maximum Inode Space**: Enter the maximum percentage of space in the filesystem that can be allocated to inodes. The default is 25%. (Setting the value to 0 means that the entire filesystem can become inode blocks.)
- **Flag Unwritten Extents**: Check this box to flag unwritten extents. If unwritten extents are flagged, filesystem write performance will be negatively affected for preallocated file extents because extra filesystem transactions are required to convert extent flags for the range of the file.

  You should disable this feature (by unchecking the box) if the filesystem must be used on operating system versions that do not support the flagging capability.
- **Data Region Size**: Enter the size of the data region of the filesystem as a number of 512-byte blocks. This number is usually equal to the size of the data subvolume. You should specify a size other than 0 only if the filesystem should occupy less space than the size of the data subvolume.
- **Use Log Subvolume for Log**: Check this box to specify that the log section of the filesystem should be written to the log subvolume of the XVM logical volume. If the volume does not contain a log subvolume, the log section will be a piece of the data section on the data subvolume.
- **Log Size**: Enter the size of the log section of the filesystem as a number of 512-byte blocks. You should specify a size other than 0 only if the log should occupy less space than the size of the log subvolume.
- **Real-Time Section Size**: Enter the size of the real-time section of the filesystem as a number of 512-byte blocks. This value is usually equal to the size of the real-time subvolume, if there is one. You should specify a size other than 0 only if the real-time section should occupy less space than the size of the real-time subvolume.

  Note: XVM on Linux does not support real-time subvolumes.

3. Click **OK**.

Grow a Filesystem with the GUI

This task lets you grow a mounted filesystem.

Note: In order to grow a filesystem, you must first increase the size of the logical volume on which the filesystem is mounted. For information on modifying XVM volumes, see the *XVM Volume Manager Administrator's Guide*.

To grow a filesystem, do the following:

1. Enter the following information:

   - **Filesystem**: Select the name of the filesystem you want to grow. The list of available filesystems is determined by looking for block devices containing XFS superblocks.

   - **Specify Sizes**: Check this option to modify the default options for the filesystem, including data region size and (if already present for the filesystem) log size and real-time section size.

     By default, the filesystem will be created with the data region size equal to the size of the data subvolume. If the volume contains a log subvolume, the log size will be set to the size of the log subvolume. If the volume contains a real-time subvolume, the real-time section size will be set to the size of the real-time subvolume.

2. If you checked the **Specify Sizes** box, click **Next** to move to page 2. For more information about these fields, see the IRIX `mkfs_xfs` or Linux `mkfs.xfs` man page.

   - **Data Region Size**: Enter the size of the data region of the filesystem as a number of 512-byte blocks. This number is usually equal to the size of the data subvolume. You should specify a size other than 0 only if the filesystem should occupy less space than the size of the data subvolume.

   - **Log Size**: Enter the size of the log section of the filesystem as a number of 512-byte blocks. You should specify a size other than 0 only if the log should occupy less space than the size of the log subvolume. This option only appears if the filesystem has a log subvolume.

   - **Real-Time Section Size**: Enter the size of the real-time section of the filesystem as a number of 512-byte blocks. This value is usually equal to the size of the real-time subvolume, if there is one. You should specify a size other than 0 only if the real-time section should occupy less space than the size of the real-time subvolume. This option only appears if the filesystem has a real-time subvolume.

Note: XVM on Linux does not support real-time subvolumes.

---

3. Click **OK**.

Define CXFS Filesystems with the GUI

This task lets you define one or more CXFS filesystems having the same ordered list of potential metadata servers and the same list of client nodes.

Note: If you select multiple device names, the path you enter for the mount point will be used as a prefix to construct the actual mount point for each filesystem.

This task assumes that you have created volume headers on your disk drives, created the XVM logical volumes, and made the filesystems. "Configuring with the GUI" on page 129.

To define filesystems, do the following:

1. Enter the following information:

   - **Device Name**: Select the device names of the XVM volumes on which the filesystems will reside.

   - **Mount Point**: The directory on which the specified filesystem will be mounted. This directory name must begin with a slash (`/`). The same mount point will be used on all the nodes in the cluster. For example, if you select the device name `/dev/cxvm/cxfs1` and want to mount it at `/mount/cxfs1`, you would enter `/mount/cxfs1` for the **Mount Point** value.

     If you selected multiple device names in order to define multiple CXFS filesystems, the mount point path will be constructed using the mount point you enter as a **prefix** and the name of each device name (not including the `/dev/cxvm` portion) as the suffix. For example, if you select two volume device names (`/dev/cxvm/cxfs1` and `/dev/cxvm/cxfs2`) and enter a mount point of `/mount/`, then the CXFS filesystems will be mounted as `/mount/cxfs1` and `/mount/cxfs2`, respectively. If instead you had entered `/mount` for the mount point, the filesystems would be mounted as `/mountcxfs1` and `/mountcxfs2`.

     For more information, see the `mount` man page.

- (*Optional*) **Mount Options**: These options are passed to the `mount` command and are used to control access to the specified XVM volume. Separate multiple options with a comma. For a list of the available options, see the `fstab` man page.
- **Force Unmount**: Select the default behavior for the filesystem. This option controls what action CXFS takes if there are processes that have open files or current directories in the filesystems that is to be unmounted. If you select **On**, the processes will be killed and the unmount will occur. If you select **Off**, the processes will not be killed and the filesystem will not be unmounted. SGI recommends that you set **Force Unmount** to **On** in order to improve the stability of the CXFS cluster. This value can be overridden when you perform a manual unmount; see "Unmount CXFS Filesystems with the GUI" on page 208.
- **Metadata Servers**: A list of administration nodes that are able to act as metadata servers. All potential metadata servers within a cluster must run the same type of operating system (that is, all IRIX or all Linux).

  To add a CXFS administration node to the list of servers, choose a name from the pull-down node list and click **Add**. To select all nodes listed, click **Add All**.

  Note: Relocation is disabled by default. Recovery and relocation are supported only when using standby nodes. Therefore, you should only define multiple metadata servers for a given filesystem if you are using the standby node model. See "Relocation" on page 20.

  To remove a node from the list of servers, click the name in the list to select it and then click **Remove**.

  Note: The order of servers is significant. The first node listed is the preferred metadata server. Click a logical name to select it and then click the arrow buttons to arrange the servers in the order that they should be used.

  However, it is impossible to predict which server will actually become the server during the boot-up cycle because of network latencies and other unpredictable delays. The first available node in the list will be used as the active metadata server.

- **Enable Mount on:** A choice of either all nodes in the cluster or a list of selected CXFS administration nodes that you specify on a second page. (The filesystem is always mounted on the current metadata server.)

- **If Nodes are Added to the Cluster Later:** This option permits the filesystem to be mounted on all nodes that might be added to the cluster at some later date. This option is selected by default.
- If you chose **Only Selected Nodes** above, click **Next** to move to the second page of the task.

  Selected Nodes: You can select the desired nodes from the **Node** list. You can also click **Add All** to select all nodes, which is the same as selecting **All Nodes Currently in Cluster**.

2. Click **OK**.

After defining the filesystems, you can mount them on the specified client nodes in the cluster by running the **Mount CXFS Filesystems** task.

Note: After a filesystem has been defined in CXFS, running `mkfs` on it (or using the "Make Filesystems with the GUI" on page 200 task) will cause errors to appear in the system log file. To avoid these errors, run `mkfs` before defining the filesystem in CXFS, or delete the CXFS filesystem before running `mkfs`. See "Delete a CXFS Filesystem with the GUI" on page 210.

Modify a CXFS Filesystem with the GUI

Note: You cannot modify a mounted filesystem.

To modify an existing filesystem, do the following:

1. Enter the following information:

   - **Filesystem to Modify**: Choose a filesystem from the pull-down menu. This displays information for that filesystem in the various fields.
   - **Mount Point** and **Mount Options**: Change the information displayed for the selected filesystem as needed. To erase text, backspace over the text or select the text and type over it.
   - (*Optional*) **Mount Options**: These options are passed to the `mount` command and are used to control access to the specified XVM volume. For a list of the available options, see the `fstab` man page.
   - **Metadata Servers**:
     - To delete a node from the list of servers, click its name and then click **Delete**.
     - To add a new CXFS administration node to the list of servers, select it from the pull-down list and click **Add**. To select all CXFS administration nodes, select **Add All**. The list for a given filesystem must consist of nodes running the same operating system.
     - To rearrange the priority of a server, select it by clicking its name and then click the arrow buttons as needed.
   - **Enable Mount on:** A choice of either all nodes in the cluster or a list of selected nodes that you specify on a second page. (The filesystem is always mounted on the current metadata server.)
   - **If Nodes are Added to the Cluster Later:**This option permits the filesystem to be mounted on all nodes that might be added to the cluster at some later date. This option is selected by default.
   - If you chose **Only Selected Nodes** above, click **Next** to move to the second page of the task.

Selected Nodes: You can select the desired nodes from the **Node** list. You can also click **Add All** to select all nodes, which is the same as selecting **All Nodes Currently in Cluster**.

2. Click **OK**.

Mount CXFS Filesystems with the GUI

To mount existing filesystems on all of their client nodes, do the following:

1. **Filesystem to Mount**: Choose the filesystem to be mounted.

2. Click **OK**.

If CXFS services are not active, mounting a filesystem will not completely succeed. The filesystem will be marked as ready to be mounted and a warning message will be displayed in the **Mount a Filesystem** task. The filesystem will not actually be mounted until you have started CXFS services. For information, see "Start CXFS Services with the GUI" on page 191.

Unmount CXFS Filesystems with the GUI

To unmount filesystems from all of their client nodes, do the following:

1. Enter the following information:

   - **Filesystem to Unmount**: Choose the filesystems to be unmounted.

   - **Force Unmount** : Click **On** to force an unmount for all selected filesystems (no matter how they have been defined) or **Default** to force an unmount for those filesystems that have the forced unmount option set in their definition.

     This option controls what action CXFS takes if there are processes that have open files or current directories in the filesystems that are to be unmounted. If forced is used (by selecting **On** or by selecting **Default** if force is the default behavior), the processes will be killed and the unmount will occur. If you select **Off**, the processes will not be killed and the filesystem will not be unmounted. The option is set to **Default** by default.

2. Click **OK**.

Mount a Filesystem Locally

This task lets you mount a filesystem only on the node to which the GUI is connected (the local node).

To mount a filesystem locally, do the following:

1. Enter the following information:
   - **Filesystem to Mount**: Select the filesystem you wish to mount. The list of available filesystems is determined by looking for block devices containing XFS superblocks.
   - **Mount Point**: Specify the directory on which the selected filesystem will be mounted.
   - (*Optional*) **Mount Options**: Specify the options that should be passed to the `mount` command. For more information about available options, see the `fstab` man page.
2. By default, the filesystem will remount every time the system starts. However, if you uncheck the box, the mount will take place only when you explicitly use this task.
3. Click **OK**.

For more information, see the `mount` man page.

Unmount a Local Filesystem

To unmount a filesystem from the local node, do the following:

1. Enter the following information:
   - **Filesystem to Unmount**: Choose the filesystem to be unmounted.
   - **Remove Mount Information**: Click the check box to remove the mount point from the `/etc/fstab` file, which will ensure that the filesystem will remain unmounted after the next reboot. This item is available only if the mount point is currently saved in `/etc/fstab`.
2. Click **OK**.

Delete a CXFS Filesystem with the GUI

You cannot delete a filesystem that is currently mounted. To unmount a filesystem, see "Unmount CXFS Filesystems with the GUI" on page 208.

To permanently delete an unmounted filesystem, do the following:

1. **Filesystem to Delete**: Choose the name of the filesystem from the pull-down list.
2. Click **OK**.

Remove Filesystem Mount Information

This task lets you delete a local filesystem's mount information in `/etc/fstab`.

Note: The filesystem will still be present on the volume.

Do the following:

1. **Filesystem Name**: Select the filesystem for which you want to remove mount information. The list of available filesystems is determined by looking for block devices containing XFS superblocks.
2. Click **OK**.

Relocate a Metadata Server for a CXFS Filesystem with the GUI

If relocation is explicitly enabled in the kernel with the `cxfs_relocation_ok` systune, you can relocate the metadata server for a filesystem to any other potential metadata server in the list (see "Relocation" on page 20). The filesystem must be mounted on the system to which the GUI is connected.

1. Enter the following information:
   - **Filesystem**: Select the desired filesystem from the list.
   - **Current Metadata Server:** The current metadata server will be displayed for you.
   - **New Metadata Server**: Select the desired node from the list.

The selected server will assume responsibility for moderating access to the selected filesystem **after** you run the **Start CXFS Services** task; see "Start CXFS Services with the GUI" on page 191.

2. Click **OK** to complete the task.

CXFS kernel membership is not affected by relocation. However, users may experience a degradation in filesystem performance while the metadata server is relocating.

Privileges Tasks with the GUI

The privileges tasks let you grant specific users the ability to perform specific tasks, and to revoke those privileges.

Note: You cannot grant or revoke tasks for users with a user ID of 0.

Grant Task Access to a User or Users

You can grant access to a specific task to one or more users at a time.

Note: Access to the task is only allowed on the node to which the GUI is connected; if you want to allow access on another node in the pool, you must connect the GUI to that node and grant access again.

Do the following:

1. Select the user or users for whom you want to grant access. You can use the following methods to select users:

   - Click to select one user at a time
   - `Shift`+click to select a block of users
   - `Ctrl`+click to toggle the selection of any one user, which allows you to select multiple users that are not contiguous
   - Click **Select All** to select all users

   Click **Next** to move to the next page.

2. Select the task or tasks to grant access to, using the above selection methods. Click **Next** to move to the next page.
3. Confirm your choices by clicking **OK**.

Note: If more tasks than you selected are shown, then the selected tasks run the same underlying privileged commands as other tasks, such that access to the tasks you specified cannot be granted without also granting access to these additional tasks.

To see which tasks a specific user can access, select **View: Users**. Select a specific user to see details about the tasks available to that user.

To see which users can access a specific task, select **View: Task Privileges**. Select a specific task to see details about the users who can access it and the privileged commands it requires.

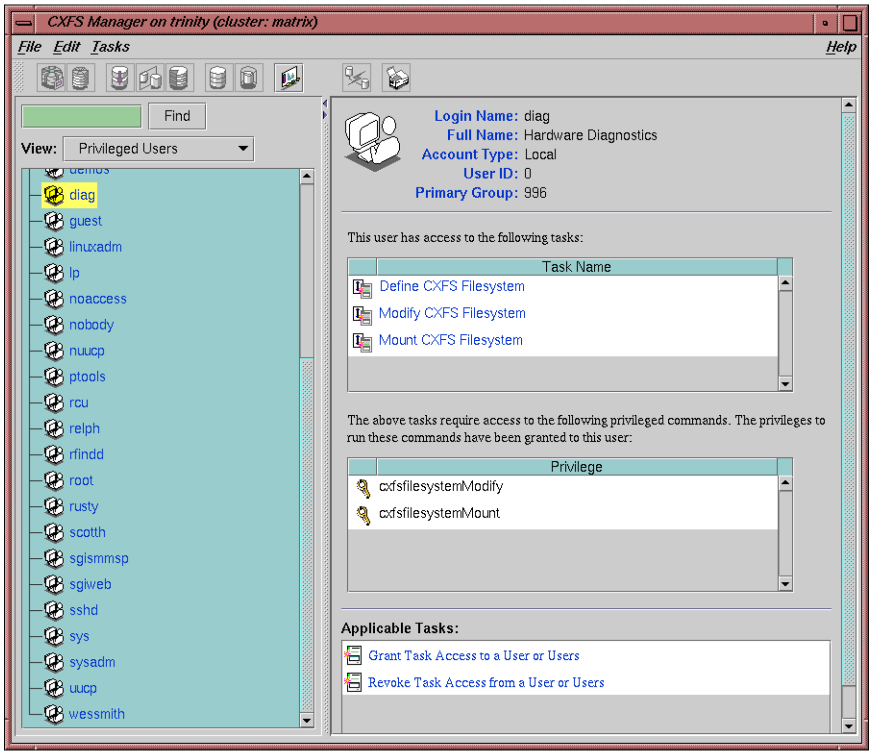
Granting Access to a Few Tasks

Suppose you wanted to grant user `diag` permission to define, modify, and mount CXFS filesystems. You would do the following:

1. Select `diag` and click **Next** to move to the next page.
2. Select the tasks you want `diag` to be able to execute:
   a. `Ctrl`+click **Define CXFS Filesystem**
   b. `Ctrl`+click **Modify CXFS Filesystem**
   c. `Ctrl`+click **Mount CXFS Filesystem**

   Click **Next** to move to the next page.
3. Confirm your choices by clicking **OK**.

Figure 9-6 shows the tasks that `diag` can now execute. This screen is displayed when you select **View: Users** and click `diag` to display information in the details area of the GUI window. The privileged commands listed are the underlying commands executed by the GUI tasks.



Login Name: diag
Full Name: Hardware Diagnostics
Account Type: Local
User ID: 0
Primary Group: 996

This user has access to the following tasks:

Task Name
Define CXFS Filesystem
Modify CXFS Filesystem
Mount CXFS Filesystem

The above tasks require access to the following privileged commands. The privileges to run these commands have been granted to this user:

Privilege
cxfsfilesystemModify
cxfsfilesystemMount

Applicable Tasks:

Grant Task Access to a User or Users

Revoke Task Access from a User or Users

Figure 9-6 Task Privileges for a Specific User

Granting Access to Most Tasks

Suppose you wanted to give user `sys` access to all tasks **except** changing the cluster contents (which also implies that `sys` cannot delete the nodes in the cluster, nor the cluster itself). The easiest way to do this is to select all of the tasks and then deselect the few you want to restrict. You would do the following:

1. Select `sys` and click **Next** to move to the next page.
2. Select the tasks you want `sys` to be able to execute:
   a. Click **Select All** to highlight all tasks.
   b. Deselect the task to which you want to restrict access. `Ctrl`+click **Add/Remove Nodes in Cluster**.

Click **Next** to move to the next page.

3. Confirm your choices by clicking **OK**.

Revoke Task Access from a User or Users

You can revoke task access from one or more users at a time.

Note: Access to the task is only revoked on the node to which the GUI is connected; if a user has access to the task on multiple nodes in the pool, you must connect the GUI to those other nodes and revoke access again.

Do the following:

1. Select the user or users from whom you want to revoke task access. You can use the following methods to select users:
   - Click to select one user at a time
   - `Shift`+click to select a block of users
   - `Ctrl`+click to toggle the selection of any one user, which allows you to select multiple users that are not contiguous
   - Click **Select All** to select all users

   Click **Next** to move to the next page.
2. Select the task or tasks to revoke access to, using the above selection methods. Click **Next** to move to the next page.
3. Confirm your choices by clicking **OK**.

   Note: If more tasks than you selected are shown, then the selected tasks run the same underlying privileged commands as other tasks, such that access to the tasks you specified cannot be revoked without also revoking access to these additional tasks.

To see which tasks a specific user can access, select **View: Users**. Select a specific user to see details about the tasks available to that user.

To see which users can access a specific task, select **View: Task Privileges**. Select a specific task to see details about the users who can access it.

Chapter 10

Reference to `cmgr` Tasks for CXFS

This chapter discusses the following:

For an overview of the tasks that must be performed to configure a cluster, see "Configuring with the `cmgr` Command" on page 133.

Tasks must be performed using a certain hierarchy. For example, to modify a partition ID, you must first identify the node name.

You can also use the `clconf_info` tool to view status. See Chapter 15, "Monitoring Status" on page 359.

Note: CXFS requires a license to be installed on each node. If you install the software without properly installing the license, you cannot use the `cmgr` command. For more information about licensing, see Chapter 2, "CXFS and XVM FLEXlm Licenses" on page 51, Chapter 4, "IRIX CXFS Installation" on page 65, and Chapter 5, "Linux CXFS Installation" on page 83. For information about licensing and nodes running an operating system other than IRIX or linux, see the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*

---

cmgr Overview

To use the `cmgr` command, you must be logged in as `root` on a CXFS administration node. Then enter either of the following:

```
# /usr/cluster/bin/cmgr
```

or

```
# /usr/cluster/bin/cluster_mgr
```

After you have entered this command, you will see the following message and the command prompt (`cmgr>`):

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

cmgr>
```

Making Changes Safely

Do not make configuration changes on two different administration nodes in the pool simultaneously, or use the CXFS GUI, `cmgr`, and `xvm` commands simultaneously to make changes. You should run one instance of the `cmgr` command or the CXFS GUI on a single administration node in the pool when making changes at any given time. However, you can use any node in the pool when requesting status or configuration information.

Getting Help

After the command prompt displays, you can enter subcommands. At any time, you can enter `?` or `help` to bring up the `cmgr` help display.

Using Prompt Mode

The `-p` option to `cmgr` displays prompts for the required inputs of administration commands that define and modify CXFS components. You can run in prompt mode in either of the following ways:

- Specify a `-p` option on the command line:

  ```
  # cmgr -p
  ```

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

  ```
  cmgr> set prompting on
  ```

  This method allows you to toggle in and out of prompt mode as you execute individual subcommands. To get out of prompt mode, enter the following:

  ```
  cmgr> set prompting off
  ```

The following shows an example of the questions that may be asked in prompting mode (the actual questions asked will vary depending upon your answers to previous questions):

```
cmgr> define node nodename
Enter commands, you may enter "done" or "cancel" at any time to exit

Hostname[optional] ?
Is this a FailSafe node <true|false> ?
Is this a CXFS node <true|false> ?
Operating System <IRIX|Linux32|Linux64|AIX|HPUX|Solaris|MacOSX|Windows> ?
Node Function <server_admin|client_admin|client_only> ?
Node ID ?[optional]
Partition ID ?[optional] (0)
Do you wish to define failure hierarchy[y/n]:
Reset type <powerCycle|reset|nmi> ? (powerCycle)
Do you wish to define system controller info[y/n]:
Sysctrl Type <msc|mmsc|l2|l1>? (msc)
Sysctrl Password[optional] ? ( )
Sysctrl Status <enabled|disabled> ?
Sysctrl Owner ?
Sysctrl Device ?
Sysctrl Owner Type <tty|network> ? (tty)
Number of Network Interfaces ? (1)
NIC 1 - IP Address ?
```

For details about this task, see "Define a Node with `cmgr`" on page 224.

Completing Actions and Cancelling

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

- `cancel`, which aborts the current mode and discards any changes you have made
- `done`, which executes the current definitions or modifications and returns to the `cmgr>` prompt

Using Script Files

You can execute a series of `cmgr` commands by using the `-f` option and specifying an input file:

```
cmgr -f input_file
```

Or, you could include the following as the first line of the file and then execute it as a script:

```
#!/usr/cluster/bin/cmgr -f
```

Each line of the file must be a valid `cmgr` command line, comment line (starting with #), or a blank line.

Note: You must include a `done` command line to finish a multilevel command and end the file with a `quit` command line.

If any line of the input file fails, `cmgr` will exit. You can choose to ignore the failure and continue the process by using the `-i` option with the `-f` option, as follows:

```
cmgr -if input_file
```

Or include it in the first line for a script:

```
#!/usr/cluster/bin/cmgr -if
```

Note: If you include `-i` when using a `cmgr` command line as the first line of the script, you must use this exact syntax (that is, `-if`).

For example, suppose the file `/tmp/showme` contains the following:

```
cxfs6# more /tmp/showme
show clusters
show nodes in cluster cxfs6-8
quit
```

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

```
cxfs6# /usr/cluster/bin/cmgr -if /tmp/showme

1 Cluster(s) defined
        cxfs6-8


Cluster cxfs6-8 has following 3 machine(s)
        cxfs6
        cxfs7
        cxfs8
```

Or you could include the `cmgr` command line as the first line of the script, give it execute permission, and execute `showme` itself:

```
cxfs6# more /tmp/showme
#!/usr/cluster/bin/cmgr -if
#
show clusters
show nodes in cluster cxfs6-8
quit

cxfs6# /tmp/showme

1 Cluster(s) defined
        cxfs6-8



Cluster cxfs6-8 has following 3 machine(s)
        cxfs6
        cxfs7
        cxfs8
```

For an example of defining a complete cluster, see "Script Example" on page 277.

Invoking a Shell from within cmgr

To invoke a shell from within `cmgr`, enter the following:

```
cmgr> sh
cxfs6#
```

To exit the shell and to return to the `cmgr>` prompt, enter the following:

```
cxfs6# exit
cmgr>
```

Entering Subcommands on the Command Line

You can enter some `cmgr` subcommands directly from the command line using the following format:

`cmgr -c "`*subcommand*`"`

where *subcommand* can be any of the following with the appropriate operands:

- `admin`, which allows you to perform certain actions such as resetting a node
- `delete`, which deletes a cluster or a node
- `help`, which displays help information
- `show`, which displays information about the cluster or nodes
- `start`, which starts CXFS services and sets the configuration so that CXFS services will be automatically restarted upon reboot
- `stop`, which stops CXFS services and sets the configuration so that CXFS services are not restarted upon reboot
- `test`, which tests connectivity

For example, to display information about the cluster, enter the following:

```
# cmgr -c "show clusters"
1 Cluster(s) defined
      eagan
```

See the `cmgr` man page for more information.

Template Scripts

The `/var/cluster/cmgr-templates` directory contains template `cmgr` scripts that you can modify to configure the different components of your system.

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

The `/var/cluster/cmgr-templates` directory contains the following templates to create a cluster and nodes:

- `cmgr-create-cluster`
- `cmgr-create-node`

To create a CXFS configuration, you can concatenate multiple templates into one file and execute the resulting script.

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

For example, for a three-node configuration, you would concatenate three copies of the `cmgr-create-node` file and one copy of the `cmgr-create-cluster` file.

Set Configuration Defaults with cmgr

You can set a default cluster and node to simplify the configuration process for the current session of `cmgr`. The default will then be used unless you explicitly specify a name. You can use the following commands to specify default values:

```
set cluster  clustername
set node hostname
```

clustername and *hostname* are logical names. Logical names cannot begin with an underscore (_) or include any whitespace characters, and can be at most 255 characters.

To view the current defaults, use the following:

```
show set defaults
```

For example:

```
cmgr> set cluster cxfs6-8
cmgr> set node cxfs6
cmgr> show set defaults
Default cluster set to: cxfs6-8

Default node set to: cxfs6
Default cdb set to: /var/cluster/cdb/cdb.db
Default resource_type is not set
Extra prompting is set off
```

Node Tasks with `cmgr`

This section tells you how to define, modify, delete, display, and reset a node using `cmgr`.

Note: The entire cluster status information is sent to each CXFS administration node each time a change is made to the cluster database; therefore, the more CXFS administration nodes in a configuration, the longer it will take.

Define a Node with `cmgr`

To define a node, use the following commands:

```
define node logical_hostname
    set hostname to hostname
    set nodeid to nodeID
    set node_function to server_admin|client_admin|client_only
    set partition_id to partitionID
    set reset_type to powerCycle|reset|nmi
    set sysctrl_type to msc|mmsc|l2|l1 (based on node hardware)
    set sysctrl_password to password
    set sysctrl_status to enabled|disabled
    set sysctrl_owner to node_sending_reset_command
    set sysctrl_device to port|IP_address_or_hostname_of_L2
    set sysctrl_owner_type to tty|network
    set is_failsafe to true|false
    set is_cxfs to true|false
    set operating_system to irix|linux32|linux64|aix|solaris|macosx|windows
    set weight to 0|1 (no longer needed)
    add nic IP_address_or_hostname (if DNS)
            set heartbeat to true|false
            set ctrl_msgs to true|false
            set priority to integer
    remove nic IP_address_or_hostname (if DNS)
    set hierarchy to [system][fence][reset][fencereset][shutdown]
```

Usage notes:

- *logical_hostname* is a simple hostname (such as `lilly`) or a fully qualified domain name (such as `lilly.mycompany.com`) or an entirely different name (such as `nodeA`). Logical names cannot begin with an underscore (_) or include any whitespace characters, and can be at most 255 characters.
- `hostname` is the fully qualified hostname unless the simple hostname is resolved on all nodes. Use the `ping` to display the fully qualified hostname. Do not enter an IP address. The default for *hostname* is the value for *logical_hostname*; therefore, you must supply a value for this command if you use a value other than the hostname or an abbreviation of it for *logical_hostname*.
- `nodeid` is an integer in the range 1 through 32767 that is unique among the nodes in the pool. You must not change the node ID number after the node has been defined.
  - For administration nodes, this value is optional. If you do not specify a number for an administration node, CXFS will calculate an ID for you. The default ID is a 5-digit number based on the machine's serial number and other machine-specific information; it is not sequential.
  - For client-only nodes, you must specify a unique value.
- `node_function` specifies the function of the node. Enter one of the following:
  - `server_admin` is an IRIX or linux node on which you will execute cluster administration commands and that you also want to be a CXFS metadata server. (You will use the **Define a CXFS Filesystem** task to define the specific filesystem for which this node can be a metadata servers.) Use this node function only if the node will be a metadata servers. You must install the `cluster_admin` product on this node.

- `client_admin` is an IRIX or linux node on which you will execute cluster administration commands but that you do not want to use as a CXFS metadata server. Use this node function only if the node will run FailSafe but you do not want it to be a metadata server. You must install the `cluster_admin` product on this node.
- `client_only`, is a node that shares CXFS filesystems but on which you will not execute cluster administration commands and that will not be a CXFS metadata server. Use this node function for all nodes other than those that will be metadata servers, or those that will run FailSafe without being a metadata server. You must install the `cxfs_client` product on this node. This node can run IRIX, SGI ProPack for Linux, Linux third-party, AIX, Solaris, Mac OS X, or Windows. (Nodes other than IRIX and SGI ProPack for Linux are **required** to be client-only nodes.)

AIX, Solaris, Mac OS X, and Windows nodes are automatically specified as `client-only`. You should specify `client-only` with `linux32`.

- `partition_id` uniquely defines a partition in a partitioned Origin 3000 or Altix 3000 system. The `set partition_id` command is optional; if you do not have a partitioned Origin 3000 system, you can skip this command or enter 0.

Note: In an Origin 3000 system, use the `mkpart` command to determine this value:

- The `-n` option lists the partition ID (which is 0 if the system is not partitioned).
- The `-l` option lists the bricks in the various partitions (use *rack#.slot#* format in `cmgr`)

  For example (output truncated here for readability):

```
# mkpart -n
Partition id = 1
# mkpart -l
partition: 3 = brick: 003c10 003c13 003c16 003c21 003c24 003c29 ...
partition: 1 = brick: 001c10 001c13 001c16 001c21 001c24 001c29 ...
```

  You could enter one of the following for the **Partition ID** field:

```
1
001.10
```

To unset the partition ID, use a value of `0` or `none`.

On an Altix 3000, you can find the partition ID by reading the `proc` file. For example:

```
[root@linux root]# cat /proc/sgi_sn/partition_id
0
```

The `0` indicates that the system is not partitioned. If the system is partitioned, the number of partitions (such as `1`, `2`, etc.) is displayed.

- `reset_type` can be one of the following:
  - `powerCycle` shuts off power to the node and then restarts it
  - `reset` simulates the pressing of the reset button on the front of the machine
  - `nmi` (nonmaskable interrupt) performs a core-dump of the operating system kernel, which may be useful when debugging a faulty machine
- `sysctrl_type` is the system controller type based on the node hardware, as show in Table 10-1.

Table 10-1 System Controller Types

`l1`	`l2`	`msc`	`mmsc`
Origin/Onyx 300/350	Altix 350	Origin 200	Rackmount SGI 2400/2800
Origin/Onyx 3200C	Altix 3300/3700	Onyx2 Deskside	Onyx2
Tezro	Prism	SGI 2100/2200 deskside systems	
	Origin 3400/3800		
	Onyx 3000		

l1	l2	msc	mmsc
	Origin 300 with NUMAlink module Altix systems with integrated L2s, such as a NUMAlink 4 R-brick, or Altix 3000 Bx2		

- `sysctrl_password` is the password for the system controller port, not the node's `root` password or PROM password. On some nodes, the system administrator may not have set this password. If you wish to set or change the system controller password, consult the hardware manual for your node.
- `sysctrl_status` allows you to provide information about the system controller but temporarily disable reset by setting this value to `disabled` (meaning that CXFS cannot reset the node). To allow CXFS to reset the node, enter `enabled`. For nodes without system controllers, set this to `disabled`; see "Requirements" on page 37.
- `sysctrl_device` is one of the following:
  - The port used for systems with serial ports. `/dev/ttyd2` is the most commonly used port, except on Origin 300 and Origin 350 systems, where `/dev/ttyd4` is commonly used.
  - The IP address or hostname of the L2 controller for Altix systems that use an integrated L2, such as a NUMAlink 4 R-brick, or SGI Altix 3000 Bx2 systems.
- `sysctrl_owner` is the name of the node that sends the reset command. Serial cables must physically connect the node being defined and the owner node through the system controller port. At run time, the node must be defined in the CXFS pool.
- `sysctrl_owner_type` is either `tty` for TTY devices or `network` for Altix systems without an integrated L2 (such as a NUMAlink 4 R-brick) or SGI Altix 3000 Bx2 systems. For example:
  - For a system with serial ports:

    ```
    Sysctrl Device? /dev/ttyd2
    Sysctrl Owner Type? <tty|network> tty
    ```

  – For an Altix system with an integrated L2, such as a NUMAlink 4 R-brick, or SGI Altix 3000 Bx2 systems:

```
Sysctrl Device? l2.company.com
Sysctrl Owner Type? <tty|network> network
```

- If you are running just CXFS on this node, set `is_cxfs` to `true` and `is_failsafe` to `false`. If you are running both CXFS and FailSafe on this node in a coexecution cluster, set both values to `true`.

- `operating_system` can be set to `irix`, `linux32`, `linux64`, `aix`, `solaris`, `macosx`, or `windows`. Choose `windows` for Windows 2000, Windows 2003, or Windows XP. Choose `linux32` when defining an AMD64/EM64T or Itanium 2 node.

  Note: For support details, see the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

  If you specify `aix`, `solaris`, `macosx` or `windows`, the `weight` is assumed to be 0. If you try to specify incompatible values for `operating_system` and `is_failsafe` or `weight`, the `define` command will fail.

- `weight`, which is automatically set internally to either 0 or 1 to specify how many votes a particular CXFS administration node has in CXFS kernel membership decisions. This information is now set by the `Node Function` field and this command is no longer needed.

  Note: Although it is possible to use the `set weight` command to set a weight other than 0 or 1, SGI recommends that you do not do so. There is no need for additional weight.

- `nic` is the IP address or hostname of the private network. (The hostname must be resolved in the `/etc/hosts` file.)

  There can be up to 8 network interfaces. The NIC number is not significant. Priority `1` is the highest priority. By default, only the priority `1` NICs will be used as the CXFS private network and they must be on the same subnet. However, you can use the `add net` command to configure the NICs of a given priority into a network; each network takes its priority from the set of NICs it contains. In this case, if the highest priority network fails, the second will be used, and so on; see "Define a Cluster with `cmgr`" on page 246.

Note: You cannot add a NIC or a network grouping while CXFS services are active (that is, when `start cx_services` has been executed); doing so can lead to cluster malfunction. If services have been started, they should be stopped with `stop cx_services`.

If you do not use the `add net` command to group the NICs into a set of networks, all NICs other than priority `1` are ignored.

SGI requires that this network be private; see "Private Network" on page 19.

For more information about using the hostname, see "Hostname Resolution and Network Configuration Rules" on page 57.

- `hierarchy` defines the fail action hierarchy, which determines what happens to a failed node. You can specify up to three options. The second option will be completed only if the first option fails; the third option will be completed only if both the first and second options fail. Options must be separated by commas and no whitespace.

  The option choices are as follows:

  – `system` deletes all hierarchy information about the node from the database, causing the system defaults to be used. The system defaults are the same as entering `reset,shutdown`. This means that a reset will be performed on a node with a system controller; if the reset fails or if the node does not have a system controller, CXFS services will be forcibly shut down. Therefore, you should choose a setting other than the default for nodes without system controllers; see "Requirements" on page 37. You cannot specify other hierarchy options if you specify the `system` option.

  – `fence` disables access to the storage area network (SAN) from the problem node. Fencing provides faster recovery of the CXFS kernel membership than reset. This action is available for all nodes.

    On nodes with system controllers, you would want to use I/O fencing for data integrity protection when CXFS is just a part of what the node is doing, and you prefer losing access to CXFS to having the system rebooted; for example, for a big compute server that is also a CXFS client. You would want to use reset for I/O protection on a node with a system controller when CXFS is a primary activity and you want to get it back online fast; for example, a CXFS file server.

On nodes without system controllers, your only choice for data integrity protection is I/O fencing.

Note: A switch is mandatory to support I/O fencing. See the release notes for supported switches.

– `fencereset` disables access to the SAN from the problem node and then, if the node is successfully fenced, **also** performs an asynchronous reset of the node; recovery begins without waiting for reset acknowledgement. This action is available only for nodes with system controllers; see "Requirements" on page 37.

Note: A potential metadata server must also include `reset` in its fail action hierarchy.

– `reset` performs a reset via a serial line connected to the system controller. This action is available only for nodes with system controllers. A potential metadata server must include `reset` in its fail action hierarchy.

– `shutdown` tells the other nodes in the cluster to wait before reforming the cluster. (Whether this action is set or not, the local node will automatically attempt to forcibly shut down CXFS services on the node in response to a loss of CXFS kernel membership quorum.) The surviving cluster delays the beginning of recovery to allow the node time to complete the shutdown. This action is available for all nodes.

⚠ **Caution:** If you have a cluster with **an even number of server-capable nodes** and **no tiebreaker**: to avoid a split-brain scenario, you should not use the `shutdown` setting for any server-capable node. For a more detailed explanation, see "Issues with the Shutdown Fail Action Setting" on page 29.

To perform a reset only if a fencing action fails, specify the following:

```
set hierarchy fence,reset
```

Note: If `shutdown` is not specified and the other actions fail, the node attempting to deliver the CXFS kernel membership will locally forcibly shut down CXFS services.

To perform a fence and an asynchronous reset, specify the following:

```
set hierarchy fencereset
```

To return to system defaults (`reset,shutdown`), specify the following:

```
set hierarchy system
```

For more information, see "CXFS Kernel Membership, Quorum, and Tiebreaker" on page 462, and "Define a Node with the GUI" on page 172.

In prompting mode, press the `Enter` key to use default information. (The `Enter` key is not shown in the examples.) For general information, see "Define a Node with the GUI" on page 172. Following is a summary of the prompts.

```
cmgr> define node logical_hostname
Enter commands, you may enter "done" or "cancel" at any time to exit

Hostname[optional] ? hostname
Is this a FailSafe node <true|false> ? true|false
Is this a CXFS node <true|false> ? truet
Operating System <IRIX|Linux32|Linux64|AIX|HPUX|Solaris|MacOSX|Windows> ?OS_type
Node Function <server_admin|client_admin|client_only> ? node_function
Node ID ?[optional] node_ID
Partition ID ?[optional] (0)partition_ID
Do you wish to define failure hierarchy[y/n]:y|n
Do you wish to define system controller info[y/n]:y|n
Reset type <powerCycle|reset|nmi> ? (powerCycle)
Do you wish to define system controller info[y/n]:y|n
Sysctrl Type <msc|mmsc|l2|l1>? (msc) model (based on node hardware)
Sysctrl Password[optional] ? ( )password
Sysctrl Status <enabled|disabled> ? enabled|disabled
Sysctrl Owner ? node_sending_reset_command
Sysctrl Device ? port|IP_address_or_hostname_of_L2
Sysctrl Owner Type <tty|network> ? (tty) tty|network
Number of Network Interfaces ? (1) number
NIC 1 - IP Address ? IP_address_or_hostname (if DNS)
```

For example, in normal mode:

```
# /usr/cluster/bin/cmgr
Welcome to SGI Cluster Manager Command-Line Interface
```

```
cmgr> define node foo
Enter commands, you may enter "done" or "cancel" at any time to exit

? set is_failsafe to false
? set is_cxfs to true
? set operating_system to irix
? set node_function to server_admin
? set hierarchy to fencereset,reset
? add nic 111.11.11.111
Enter network interface commands, when finished enter "done" or "cancel"

NIC 1 - set heartbeat to true
NIC 1 - set ctrl_msgs to true
NIC 1 - set priority to 1
NIC 1 - done
? done
```

For example, in prompting mode:

```
# /usr/cluster/bin/cmgr -p
Welcome to SGI Cluster Manager Command-Line Interface

cmgr> define node foo
Enter commands, you may enter "done" or "cancel" at any time to exit

Hostname[optional] ?
Is this a FailSafe node <true|false> ? false
Is this a CXFS node <true|false> ? true
Operating System <IRIX|Linux32|Linux64|AIX|HPUX|Solaris|MacOSX|Windows> ? irix
Node Function <server_admin|client_admin|client_only> server_admin
Node ID[optional]?
Partition ID ? [optional] (0)
Do you wish to define failure hierarchy[y|n]:y
Hierarchy option 0 <System|FenceReset|Fence|Reset|Shutdown>[optional] ? fencereset
Hierarchy option 1 <System|FenceReset|Fence|Reset|Shutdown>[optional] ? reset
Hierarchy option 2 <System|FenceReset|Fence|Reset|Shutdown>[optional] ?
Reset type <powerCycle|reset|nmi>  ? (powerCycle)
Do you wish to define system controller info[y/n]:n
Number of Network Interfaces ? (1)
NIC 1 - IP Address ? 111.11.11.111
NIC 1 - Heartbeat HB (use network for heartbeats) <true|false> ? true
NIC 1 - (use network for control messages) <true|false> ? true
NIC 1 - Priority <1,2,...> 1
```

Following is an example of defining a Solaris node in prompting mode (because it is a Solaris node, no default ID is provided, and you are not asked to specify the node function because it must be `client_only`).

```
cmgr> define node solaris1
Enter commands, you may enter "done" or "cancel" at any time to exit

Hostname[optional] ?
Is this a FailSafe node <true|false> ? false
Is this a CXFS node <true|false> ? true
Operating System <IRIX|Linux32|Linux64|AIX|HPUX|Solaris|MacOSX|Windows> ? solaris
Node ID ? 7
Do you wish to define failure hierarchy[y/n]:y
Hierarchy option 0 <System|FenceReset|Fence|Reset|Shutdown>[optional] ? fence
Hierarchy option 1 <System|FenceReset|Fence|Reset|Shutdown>[optional] ?
Number of Network Interfaces ? (1)
NIC 1 - IP Address ? 163.154.18.172
```

Modify a Node with cmgr

To modify an existing node, use the following commands:

```
modify node logical_hostname
    set hostname to hostname
    set partition_id to partitionID
    set reset_type to powerCycle|reset|nmi
    set sysctrl_type to msc|mmsc|l2|l1 (based on node hardware)
    set sysctrl_password to password
    set sysctrl_status to enabled|disabled
    set sysctrl_owner to node_sending_reset_command
    set sysctrl_device to port|IP_address_or_hostname_of_L2
    set sysctrl_owner_type to tty|network
    set is_failsafe to true|false
    set is_cxfs to true|false
    set weight to 0|1
    add nic IP_address_or_hostname (if DNS)
            set heartbeat to true|false
```

```
                set ctrl_msgs to true|false
                set priority to integer
        remove nic IP_address_or_hostname (if DNS)
        set hierarchy to [system] [fence][reset][fencereset][shutdown]
```

The commands are the same as those used to define a node. You can change any of the information you specified when defining a node except the node ID. For details about the commands, see "Define a Node with `cmgr`" on page 224.

Caution: Do not change the node ID number after the node has been defined.

You cannot add a NIC or a network grouping while CXFS services are active (that is, when `start cx_services` has been executed); doing so can lead to cluster malfunction. If services have been started, they should be stopped with `stop cx_services`.

You cannot modify the `operating_system` setting for a node; trying to do so will cause an error. If you have mistakenly specified the incorrect operating system, you must delete the node and define it again.

You cannot modify the node function. To change the node function, you must delete the node and redefine it (and reinstall software products, as needed); the node function for a Solaris or Windows node is always `client_only`.

Example of Partitioning

The following shows an example of partitioning an Origin 3000 system:

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

cmgr> modify node n_preston
Enter commands, when finished enter either "done" or "cancel"

n_preston ? set partition_id to 1
n_preston ? done

Successfully modified node n_preston
```

To perform this function with prompting, enter the following:

```
# cmgr -p
Welcome to SGI Cluster Manager Command-Line Interface

cmgr> modify node n_preston
Enter commands, you may enter "done" or "cancel" at any time to exit

Hostname[optional] ? (preston.engr.sgi.com)
Is this a FailSafe node <true|false> ? (true)
Is this a CXFS node <true|false> ? (true)
Node ID[optional] ? (606)
Partition ID[optional] ? (0) 1
Do you wish to modify failure hierarchy[y/n]:n
Reset type <powerCycle|reset|nmi> ? (powerCycle)
Do you wish to modify system controller info[y/n]:n
Number of Network Interfaces? (1)
NIC 1 - IP Address ? (preston)
NIC 1 - Heartbeat HB (use network for heartbeats)  ? (true)
NIC 1 - (use network for control messages)  ? (true)
NIC 1 - Priority <1,2,...> ? (1)

Successfully modified node n_preston

cmgr> show node n_preston
Logical Machine Name: n_preston
Hostname: preston.engr.sgi.com
Operating System: IRIX
Node Is FailSafe: true
Node Is CXFS: true
Node Function: client_admin
Nodeid: 606
Partition id: 1
Reset type: powerCycle
ControlNet Ipaddr: preston
ControlNet HB: true
ControlNet Control: true
ControlNet Priority: 1
```

To unset the partition ID, use a value of `0` or `none`.

Changing Failure Hierarchy

The following shows an example of changing the failure hierarchy for the node `perceval` from the system defaults to `fencereset,reset,shutdown` and back to the system defaults.

⚠ **Caution:** If you have a cluster with **an even number of server-capable nodes** and **no tiebreaker**: to avoid a split-brain scenario, you should not use the `shutdown` setting for any server-capable node. For a more detailed explanation, see "Issues with the Shutdown Fail Action Setting" on page 29.

```
cmgr> modify node perceval
Enter commands, you may enter "done" or "cancel" at any time to exit

Hostname[optional] ? (perceval.engr.sgi.com)
Is this a FailSafe node <true|false> ? (false)
Is this a CXFS node <true|false> ? (true)
Node ID[optional] ? (803)
Partition ID[optional] ? (0)
Do you wish to modify failure hierarchy[y/n]:y
Hierarchy option 0 <System|FenceReset|Fence|Reset|Shutdown>[optional] ?fencereset
Hierarchy option 1 <System|FenceReset|Fence|Reset|Shutdown>[optional] ?reset
Hierarchy option 2 <System|FenceReset|Fence|Reset|Shutdown>[optional] ?shutdown
Reset type <powerCycle|reset|nmi> ? (powerCycle)
Do you wish to modify system controller info[y/n]:n
Number of Network Interfaces ? (1)
NIC 1 - IP Address ? (163.154.18.173)

Successfully modified node perceval

cmgr> show node perceval
Logical Machine Name: perceval
Hostname: perceval.engr.sgi.com
Operating System: IRIX
Node Is FailSafe: false
Node Is CXFS: true
Node Function: client_admin
Nodeid: 803
Node Failure Hierarchy is: FenceReset Reset Shutdown
Reset type: powerCycle
```

```
ControlNet Ipaddr: 163.154.18.173
ControlNet HB: true
ControlNet Control: true
ControlNet Priority: 1
```

To return to system defaults:

```
cmgr> modify node perceval

Enter commands, you may enter "done" or "cancel" at any time to exit

Hostname[optional] ? (perceval.engr.sgi.com)
Is this a FailSafe node <true|false> ? (false)
Is this a CXFS node <true|false> ? (true)
Node ID[optional] ? (803)
Partition ID[optional] ? (0)
Do you wish to modify failure hierarchy[y/n]:y
Hierarchy option 0 <System|FenceReset|Fence|Reset|Shutdown>[optional] ?
(FenceReset) system
Reset type <powerCycle|reset|nmi> ? (powerCycle)
Do you wish to modify system controller info[y/n]:n
Number of Network Interfaces ? (1)
NIC 1 - IP Address ? (163.154.18.173)
NIC 1 - Heartbeat HB (use network for heartbeats) <true|false> ? (true)
NIC 1 - (use network for control messages) <true|false> ? (true)
NIC 1 - Priority <1,2,...> ? (1)

cmgr> show node perceval
Logical Machine Name: perceval
Hostname: perceval.engr.sgi.com
Operating System: IRIX
Node Is FailSafe: false
Node Is CXFS: true
Node Function: client_admin
Nodeid: 803
Reset type: powerCycle|reset|nmi
ControlNet Ipaddr: 163.154.18.173
ControlNet HB: true
ControlNet Control: true
ControlNet Priority: 1
```

Note: When the system defaults are in place for failure hierarchy, no status is displayed with the `show` command.

Reset a Node with `cmgr`

When CXFS is running, you can reset a node with a system controller by using the following command:

`admin reset node` *hostname*

This command uses the CXFS daemons to reset the specified node.

Even when the CXFS daemons are not running, you can reset a node with a system controller by using the `standalone` option of the `admin reset` command:

`admin reset standalone node` *hostname*

If you have defined the node but have not defined system controller information for it, you could use the following commands to connect to the system controller or reset the node:

`admin ping dev_name` *port*|*IP_address_or_hostname_of_L2* `of dev_type` *tty*|*network* `with sysctrl_type` *msc*|*mmsc*|*l2*|*l1*

`admin reset dev_name` *port*|*IP_address_or_hostname_of_L2* `of dev_type` *tty*|*network* `with sysctrl_type` *msc*|*mmsc*|*l2*|*l1*

For more information about the command elements, see "Define a Node with `cmgr`" on page 224.

The above command does not go through the `crsd` daemon.

Perform a Power Cycle on a Node with `cmgr`

When CXFS is running, you can perform a powercycle on a node with the following command:

`admin powerCycle node` *nodename*

This command uses the CXFS daemons to shut off power to the node and then restart it.

You can perform a powercycle on a node in a cluster even when the CXFS daemons are not running by using the `standalone` option:

`admin powerCycle standalone node` *nodename*

Th above command does not go through the `crsd` daemon.

If the node has not been defined in the cluster database, you can use the following command line:

`admin powerCycle dev_name` *port|IP_address_or_hostname_of_L2* `of dev_type` *tty|network* `with sysctrl_type` *msc|mmsc|l2|l1*

Perform an NMI on a Node with cmgr

When CXFS daemons are running, you can perform a nonmaskable interrupt (NMI) on a node with the following command:

`admin nmi node` *nodename*

This command uses the CXFS daemons to perform an NMI on the specified node.

You can perform an NMI on a node in a cluster even when the CXFS daemons are not running by using the `standalone` option:

`admin nmi standalone node` *nodename*

This command does not go through the CXFS daemons.

If the node has not been defined in the cluster database, you can use the following command line:

`admin nmi dev_name` *port|IP_address_or_hostname_of_L2* `of dev_type` *tty|network* `with sysctrl_type` *msc|mmsc|l2|l1*

Convert a Node to CXFS or FailSafe with cmgr

To convert an existing FailSafe node so that it also applies to CXFS, use the `modify` command to change the setting.

Note: You cannot turn off FailSafe or CXFS for a node if the respective high availability (HA) or CXFS services are active. You must first stop the services for the node.

For example, in normal mode:

```
cmgr> **modify node cxfs6**
Enter commands, when finished enter either "done" or "cancel"

cxfs6 ? **set is_FailSafe to true**
cxfs6 ? **done**

Successfully modified node cxfs6
```

For example, in prompting mode:

```
cmgr> **modify node cxfs6**
Enter commands, you may enter "done" or "cancel" at any time to exit

Hostname[optional] ? (cxfs6.americas.sgi.com)
Is this a FailSafe node <true|false> ? (false) **true**
Is this a CXFS node <true|false> ? (true)
Node ID[optional] ? (13203)
Partition ID[optional] ? (0)
Do you wish to modify failure hierarchy[y/n]:**n**
Reset type <powerCycle|reset|nmi> ? (powerCycle)
Do you wish to modify system controller info[y/n]:**n**
Number of Network Interfaces ? (1)
NIC 1 - IP Address ? (163.154.18.172)
NIC 1 - Heartbeat HB (use network for heartbeats) <true|false> ? (true)
NIC 1 - (use network for control messages) <true|false> ? (true)
NIC 1 - Priority <1,2,...> ? (1)

Successfully modified node cxfs6
```

Delete a Node with `cmgr`

To delete a node, use the following command:

`delete node` *hostname*

You can delete a node only if the node is not currently part of a cluster. If a cluster currently contains the node, you must first modify that cluster to remove the node from it.

For example, suppose you had a cluster named `cxfs6-8` with the following configuration:

```
cmgr> show cluster cxfs6-8
Cluster Name: cxfs6-8
Cluster Is FailSafe: true
Cluster Is CXFS: true
Cluster ID: 20
Cluster HA mode: normal
Cluster CX mode: normal


Cluster cxfs6-8 has following 3 machine(s)
        cxfs6
        cxfs7
        cxfs8
```

To delete node `cxfs8`, you would do the following in prompting mode (assuming that CXFS services have been stopped on the node):

```
cmgr> modify cluster cxfs6-8
Enter commands, when finished enter either "done" or "cancel"

Is this a FailSafe cluster <true|false> ? (false)
Is this a CXFS cluster <true|false> ? (true)
Cluster Notify Cmd [optional] ?
Cluster Notify Address [optional] ?
Cluster CXFS mode <normal|experimental>[optional] ? (normal)
Cluster ID ? (20)

Current nodes in cluster cxfs6-8:
Node - 1: cxfs6
Node - 2: cxfs7
Node - 3: cxfs8

Add nodes to or remove nodes/networks from cluster
Enter "done" when completed or "cancel" to abort


cxfs6-8 ? remove node cxfs8
cxfs6-8 ? done
Successfully modified cluster cxfs6-8
```

```
cmgr> show cluster cxfs6-8
Cluster Name: cxfs6-8
Cluster Is FailSafe: false
Cluster Is CXFS: true
Cluster ID: 20
Cluster CX mode: normal


Cluster cxfs6-8 has following 2 machine(s)
        cxfs6
        cxfs7
```

To delete `cxfs8` from the pool, enter the following:

```
cmgr> delete node cxfs8

Deleted machine (cxfs6).
```

Display a Node with `cmgr`

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

```
show node hostname
```

For example:

```
cmgr> show node cxfs6
Logical Machine Name: cxfs6
Hostname: cxfs6.americas.sgi.com
Operating System: IRIX
Node Is FailSafe: false
Node Is CXFS: true
Node Function: server_admin
Nodeid: 13203
Reset type: powerCycle
ControlNet Ipaddr: 163.154.18.172
ControlNet HB: true
ControlNet Control: true
ControlNet Priority: 1
```

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

```
show nodes in pool
```

For example:

```
cmgr> show nodes in pool

3 Machine(s) defined
        cxfs8
        cxfs6
        cxfs7
```

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

```
show nodes [in cluster clustername]
```

For example:

```
cmgr> show nodes in cluster cxfs6-8

Cluster cxfs6-8 has following 3 machine(s)
        cxfs6
        cxfs7
        cxfs8
```

If you have specified a default cluster, you do not need to specify a cluster when you use this command. For example:

```
cmgr> set cluster cxfs6-8
cmgr> show nodes

Cluster cxfs6-8 has following 3 machine(s)
        cxfs6
        cxfs7
        cxfs8
```

Test Node Connectivity with `cmgr`

You can use `cmgr` to test the network connectivity in a cluster. This test checks if the specified nodes can communicate with each other through each configured interface in the nodes. This test will not run if CXFS is running. This test requires that the `/etc/.rhosts` file be configured properly; see "IRIX Modifications for CXFS Connectivity Diagnostics" on page 80, "Linux Modifications for CXFS Connectivity Diagnostics" on page 93.

Use the following command to test the network connectivity for the nodes in a cluster:

```
test connectivity in cluster clustername [on node nodename1 node nodename2 ...]
```

For example:

```
cmgr> test connectivity in cluster cxfs6-8 on node cxfs7
Status: Testing connectivity...
Status: Checking that the control IP_addresses are on the same networks
Status: Pinging address cxfs7 interface ef0 from node cxfs7 [cxfs7]
Notice: overall exit status:success, tests failed:0, total tests executed:1
```

This test yields an error message when it encounters its first error, indicating the node that did not respond. If you receive an error message after executing this test, verify that the network interface has been configured up, using the `ifconfig` command. For example (line breaks added here for readability):

```
# /usr/etc/ifconfig ef0
ef0: flags=405c43 <UP,BROADCAST,RUNNING,FILTMULTI,MULTICAST,CKSUM,DRVRLOCK,IPALIAS>
inet 128.162.89.39 netmask 0xffff0000 broadcast 128.162.255.255
```

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

If the network interface is configured up, verify that the network cables are connected properly and run the test again.

Test the Serial Connections with `cmgr`

See "System Reset Connection for CXFS Administration Nodes" on page 102.

Cluster Tasks with `cmgr`

This section tells you how to define, modify, delete, and display a cluster using `cmgr`. It also tells you how to start and stop CXFS services.

Define a Cluster with `cmgr`

When you define a cluster with `cmgr`, you define a cluster and add nodes to the cluster with the same command. For general information, see "Define a Cluster with the GUI" on page 188.

Use the following commands to define a cluster:

```
define cluster clustername
    set is_failsafe to true|false
    set is_cxfs to true|false
    set clusterid to clusterID
    set notify_cmd to notify_command
    set notify_addr to email_address
    set ha_mode to normal|experimental
    set cx_mode to normal|experimental
```

```
add node node1name
add node node2name
add net network network_address mask netmask
```

Usage notes:

- `cluster` is the logical name of the cluster. Logical names cannot begin with an underscore (_) or include any whitespace characters, and can be at most 255 characters. Clusters must have unique names
- If you are running just CXFS, set `is_cxfs` to `true` and `is_failsafe` to `false`. If you are running a coexecution cluster, set both values to `true`.
- `clusterid` is a unique number within your network in the range 1 through 128. The cluster ID is used by the operating system kernel to make sure that it does not accept cluster information from any other cluster that may be on the network. The kernel does not use the database for communication, so it requires the cluster ID in order to verify cluster communications. This information in the kernel cannot be changed after it has been initialized; therefore, you must not change a cluster ID after the cluster has been defined. Clusters must have unique IDs.
- `notify_cmd` is the command to be run whenever the status changes for a node or cluster.
- `notify_addr` is the address to be notified of cluster and node status changes. To specify multiple addresses, separate them with commas. CXFS will send e-mail to the addresses whenever the status changes for a node or cluster. If you do not specify an address, notification will not be sent. If you use the `notify_addr` command, you must specify the e-mail program (such as `/usr/sbin/Mail`) as the *notify_command*.
- The `set ha_mode` and `set cx_mode` commands should usually be set to `normal`. The `set cx_mode` command applies only to CXFS, and the `set ha_mode` command applies only to IRIS FailSafe.
- `net` defines a set of NICs into a network. If the highest priority network (beginning with NIC priority `1`) fails, the next highest will be used. All NICs within one network must be at the same priority. NICs of a given priority (such as priority `2`) cannot be in two separate `net` networks. Although the primary network must be private, the backup network may be public.

  If you do not specify a `net` list, the set of priority `1` NICs are used by default as the CXFS heartbeat network and there will be no failover to any other set of NICs.

The `network` parameter specifies an IP network address (such as `1.2.3.0`) and the `mask` parameter specifies the subnet mask (such as `255.255.255.0`) in decimal notation. The order in which you specify `network` or `mask` is not important.

Note: You cannot add a NIC or a network grouping while CXFS services are active (that is, when `start cx_services` has been executed); doing so can lead to cluster malfunction. If services have been started, they should be stopped with `stop cx_services`.

The following shows the commands with prompting:

```
cmgr> define cluster clustername
Enter commands, you may enter "done" or "cancel" at any time to exit

Is this a FailSafe cluster <true|false> ? true|false
Is this a CXFS cluster  <true|false> ? true|false
Cluster Notify Cmd [optional] ?
Cluster Notify Address [optional] ?
Cluster CXFS mode <normal|experimental>[optional] use_default_of_normal
Cluster ID ? cluster_ID
No nodes in cluster clustername

No networks in cluster topiary

Add nodes to or remove nodes/networks from cluster clustername
Enter "done" when completed or "cancel" to abort

clustername ? add node node1name
clustername ? add node node2name
...
clustername ? done
Successfully defined cluster clustername

Added node <node1name> to cluster <clustername>
Added node <node2name> to cluster <clustername>

...
```

You should set the cluster to the default `normal` mode. Setting the mode to `experimental` turns off heartbeating in the CXFS kernel membership code so that you can debug the cluster without causing node failures. For example, this can be useful if you just want to disconnect the network for a short time (provided that there is no other cluster networking activity, which will also detect a failure even if there is no heartbeating). However, you should never use `experimental` mode on a production cluster and should only use it if directed to by SGI customer support. SGI does not support the use of `experimental` by customers.

For example:

```
cmgr> define cluster cxfs6-8
Enter commands, you may enter "done" or "cancel" at any time to exit

Is this a FailSafe cluster <true|false> ? false
Is this a CXFS cluster  <true|false> ? true
Cluster Notify Cmd [optional] ?
Cluster Notify Address [optional] ?
Cluster CXFS mode <normal|experimental>[optional]
Cluster ID ? 20

No nodes in cluster cxfs6-8

No networks in cluster topiary

Add nodes to or remove nodes/networks from cluster topiary
Enter "done" when completed or "cancel" to abort

cxfs6-8 ? add node cxfs6
cxfs6-8 ? add node cxfs7
cxfs6-8 ? add node cxfs8
cxfs6-8 ? done
Successfully defined cluster cxfs6-8

Added node <cxfs6> to cluster <cxfs6-8>
Added node <cxfs7> to cluster <cxfs6-8>
Added node <cxfs8> to cluster <cxfs6-8>
```

To do this without prompting, enter the following:

```
cmgr> define cluster cxfs6-8
Enter commands, you may enter "done" or "cancel" at any time to exit

cluster cxfs6-8? set is_cxfs to true
cluster cxfs6-8? set clusterid to 20
cluster cxfs6-8? add node cxfs6
cluster cxfs6-8? add node cxfs7
cluster cxfs6-8? add node cxfs8
cluster cxfs6-8? done
Successfully defined cluster cxfs6-8
```

Modify a Cluster with `cmgr`

The commands are as follows:

```
modify cluster clustername
    set is_failsafe to true
    set is_cxfs to true
    set clusterid to clusterID
    set notify_cmd to command
    set notify_addr to email_address
    set ha_mode to normal|experimental
    set cx_mode to normal|experimental
    add node node1name
    add node node2name
    remove node node1name
    remove node node2name
    add net network network_address mask netmask
    remove net network network_address
```

These commands are the same as the `define cluster` commands. For more information, see "Define a Cluster with `cmgr`" on page 246, and "Define a Cluster with the GUI" on page 188.

Note: If you want to rename a cluster, you must delete it and then define a new cluster. If you have started CXFS services on the node, you must either reboot it or reuse the cluster ID number when renaming the cluster.

However, be aware that if you already have CXFS filesystems defined and then rename the cluster, CXFS will not be able to mount the filesystems. For more information, see "Cannot Mount Filesystems" on page 408.

Convert a Cluster to CXFS or FailSafe with `cmgr`

To convert a cluster, use the following commands:

```
modify cluster clustername
  set is_failsafe to true|false
  set is_cxfs to true|false
  set clusterid to clusterID
```

- `cluster` is the logical name of the cluster. Logical names cannot begin with an underscore (_) or include any whitespace characters, and can be at most 255 characters.
- If you are running just CXFS, set `is_cxfs` to `true` and `is_failsafe` to `false`. If you are running a coexecution cluster, set both values to `true`.
- `clusterid` is a unique number within your network in the range 1 through 128. The cluster ID is used by the operating system kernel to make sure that it does not accept cluster information from any other cluster that may be on the network. The kernel does not use the database for communication, so it requires the cluster ID in order to verify cluster communications. This information in the kernel cannot be changed after it has been initialized; therefore, you must not change a cluster ID after the cluster has been defined.

For example, to convert CXFS cluster `cxfs6-8` so that it also applies to FailSafe, enter the following:

```
cmgr> modify cluster cxfs6-8
Enter commands, when finished enter either "done" or "cancel"

cxfs6-8 ? set is_failsafe to true
```

The cluster must support all of the functionalities (FailSafe and/or CXFS) that are turned on for its nodes; that is, if your cluster is of type `CXFS`, then you cannot modify a node that is part of the cluster so that it is of type `FailSafe` or of type `CXFS and FailSafe`. However, the nodes do not have to support all the functionalities of the cluster; that is, you can have a node of type `CXFS` in a cluster of type `CXFS and FailSafe`.

Delete a Cluster with cmgr

To delete a cluster, use the following command:

`delete cluster` *clustername*

However, you cannot delete a cluster that contains nodes; you must first stop CXFS services on the nodes and then redefine the cluster so that it no longer contains the nodes.

For example, in normal mode:

```
cmgr> modify cluster cxfs6-8
Enter commands, when finished enter either "done" or "cancel"

cxfs6-8 ? remove node cxfs6
cxfs6-8 ? remove node cxfs7
cxfs6-8 ? remove node cxfs8
cxfs6-8 ? done
Successfully modified cluster cxfs6-8

cmgr> delete cluster cxfs6-8

cmgr> show clusters

cmgr>
```

For example, in prompting mode:

```
cmgr> modify cluster cxfs6-8
Enter commands, you may enter "done" or "cancel" at any time to exit

Cluster Notify Cmd [optional] ?
Cluster Notify Address [optional] ?
Cluster mode <normal|experimental>[optional] ? (normal)
Cluster ID ? (55)

Current nodes in cluster cxfs6-8:
Node - 1: cxfs6
Node - 2: cxfs7
Node - 3: cxfs8

Add nodes to or remove nodes from cluster cxfs6-8
```

```
Enter "done" when completed or "cancel" to abort

cxfs6-8 ? remove node cxfs6
cxfs6-8 ? remove node cxfs7
cxfs6-8 ? remove node cxfs8
cxfs6-8 ? done
Successfully modified cluster cxfs6-8

cmgr> delete cluster cxfs6-8

cmgr> show clusters

cmgr>
```

Display a Cluster with `cmgr`

To display the clusters and their contents, use the following commands:

```
show clusters
show cluster clustername
```

For example, the following output shows that cluster `mycluster` has six nodes and two private networks, permitting network failover:

```
cmgr> show cluster topiary
Cluster Name: mycluster
Cluster Is FailSafe: false
Cluster Is CXFS: true
Cluster ID: 1
Cluster CX mode: normal


Cluster mycluster has following 6 machine(s)
        nodeA
 nodeB
 nodeC
 nodeD
 nodeE
 nodeF
```

```
CXFS Failover Networks:
    network 192.168.0.0, mask 255.255.255.0
    network 134.14.54.0, mask 255.255.255.0
```

The multiple networks listed indicates that if the higher priority network should fail, the next priority network will be used. However, the order in which the networks are listed in this output is not an indication of priority. To determine the priority of the networks, you must look at the NIC priorities in the node definition.

Cluster Services Tasks with `cmgr`

The following tasks tell you how to start and stop CXFS services and set log levels.

Start CXFS Services with `cmgr`

To start CXFS services, and set the configuration to automatically restart CXFS services whenever the system is rebooted, use one of the following commands:

```
start cx_services [on node hostname ] for cluster clustername
```

For example, to start CXFS services on all nodes in the cluster:

```
cmgr> start cx_services for cluster cxfs6-8
```

Stop CXFS Services with `cmgr`

When CXFS services are stopped on a node, filesystems are automatically unmounted from that node.

To stop CXFS services on a specified node or cluster, and prevent CXFS services from being restarted by a reboot, use the following command:

```
stop cx_services [on node hostname]for cluster clustername [force]
```

Note: This procedure is only recommended as needed for CXFS administration node because it updates the cluster database and is therefore intrusive to other nodes. When shutting down a CXFS client–only node, do not administratively stop the CXFS services on the node Rather, let the CXFS services stop by themselves when the client-only node is shut down.

For example:

```
cmgr> stop cx_services on node cxfs6 for cluster cxfs6-8

CXFS services have been deactivated on node cxfs6 (cluster cxfs6-8)

cmgr> stop cx_services for cluster cxfs6-8
```

After you have stopped CXFS services in a node, the node is no longer an active member of the cluster.

Caution: If you stop CXFS services, the node will be marked as `INACTIVE` and it will therefore not rejoin the cluster after a reboot. To allow a node to rejoin the cluster, you must restart CXFS services using `cmgr` or the GUI.

Set the Tiebreaker Node with `cmgr`

A *CXFS tiebreaker node* determines whether a CXFS kernel membership quorum is maintained when exactly half of the server-capable nodes can communicate with each other. There is no default CXFS tiebreaker.

Caution: If one of the server-capable nodes is the CXFS tiebreaker in a two server-capable cluster, failure of that node or stopping the CXFS services on that node will result in a cluster-wide forced shutdown. Therefore SGI recommends that you use client-only nodes as tiebreakers so that either server could fail but the cluster would remain operational via the other server.

The reset capability or I/O fencing with switches is **mandatory** to ensure data integrity for all nodes. Clusters should have an odd number of server-capable nodes. If you have an even number of server-capable administration nodes, define a CXFS tiebreaker node. SGI recommends making a client-only node the tiebreaker. (See "CXFS Recovery Issues in a Cluster with Only Two Server-Capable Nodes " on page 476.)

To set the CXFS tiebreaker node, use the `modify` command as follows:

```
modify cx_parameters
[on node nodename] in cluster clustername
set tie_breaker to hostname
```

To unset the CXFS tiebreaker node, use the following command:

```
set tie_breaker to none
```

For example, in normal mode:

```
cmgr> modify cx_parameters in cluster cxfs6-8
Enter commands, when finished enter either "done" or "cancel"

cxfs6-8 ? set tie_breaker to cxfs8
cxfs6-8 ? done
Successfully modified cx_parameters
```

For example, in prompting mode:

```
cmgr> modify cx_parameters in cluster cxfs6-8

(Enter "cancel" at any time to abort)

Tie Breaker Node ? (cxfs7) cxfs8
Successfully modified cx_parameters

cmgr> show cx_parameters in cluster cxfs6-8

_CX_TIE_BREAKER=cxfs8
```

Set Log Configuration with cmgr

For general information about CXFS logs, see "Set Log Configuration with the GUI" on page 193.

Display Log Group Definitions with cmgr

Use the following command to view the log group definitions:

```
show log_groups
```

This command shows all of the log groups currently defined, with the log group name, the logging levels, and the log files.

Use the following command to see messages logged by a specific daemon on a specific node:

```
show log_group LogGroupName [on node Nodename]
```

To exit from the message display, enter `Cntrl-C`.

Configure Log Groups with `cmgr`

You can configure a log group with the following command:

```
define log_group log_group on node adminhostname [in cluster clustername]
  set log_level to log_level
  add log_file log_file
  remove log_file log_file
```

Usage notes:

- `log_group` can be one of the following:

  ```
  clconfd
  cli
  crsd
  diags
  ```

- `log_level` can have one of the following values:
  - `0` gives no logging
  - `1` logs notifications of critical errors and normal operation (these messages are also logged to the `SYSLOG` file)
  - `2` logs `Minimal` notifications plus warnings
  - `5` through `7` log increasingly more detailed notifications
  - `10` through `19` log increasingly more debug information, including data structures
- *log_file*

Caution: Do not change the names of the log files. If you change the names, errors can occur.

For example, to define log group `cli` on node `cxfs6` with a log level of `5`:

```
cmgr> define log_group cli on node cxfs6 in cluster cxfs6-8

(Enter "cancel" at any time to abort)

Log Level ? (11) 5

CREATE LOG FILE OPTIONS

        1) Add Log File.
        2) Remove Log File.
        3) Show Current Log Files.
        4) Cancel. (Aborts command)
        5) Done. (Exits and runs command)

Enter option:5
Successfully defined log group cli
```

Modify Log Groups with cmgr

Use the following command to modify a log group:

`modify log_group` *log_group_name* `on node` *hostname* `[in cluster` *clustername*`]`

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

For example, to change the log level of `cli` to be `10`, enter the following:

```
cmgr> modify log_group cli on node cxfs6 in cluster cxfs6-8

(Enter "cancel" at any time to abort)

Log Level ? (2) 10

MODIFY LOG FILE OPTIONS

        1) Add Log File.
        2) Remove Log File.
        3) Show Current Log Files.
        4) Cancel. (Aborts command)
        5) Done. (Exits and runs command)
```

```
Enter option:**5**
Successfully modified log group cli
```

Revoke Membership of the Local Node with `cmgr`

To revoke CXFS kernel membership for the local node, such as before the forced CXFS shutdown, enter the following on the local node:

```
admin cxfs_stop
```

This command will be considered as a node failure by the rest of the cluster. The rest of the cluster may then fail due to a loss of CXFS kernel membership quorum, or it may decide to reset the failed node. To avoid the reset, you can modify the node definition to disable the system controller status.

Allow Membership of the Local Node with `cmgr`

Allowing CXFS kernel membership for the local node permits the node to reapply for CXFS kernel membership. You must actively allow CXFS kernel membership for the local node in the following situations:

- After a manual revocation as in "Revoke Membership of the Local Node with `cmgr`" on page 259.
- When instructed to by an error message on the console or in `/var/adm/SYSLOG`.
- After a kernel-triggered revocation. This situation is indicated by the following message in `/var/adm/SYSLOG`:

  ```
  Membership lost - withdrawing from cluster
  ```

To allow CXFS kernel membership for the local node, use the following command:

```
cmgr> admin cxfs_start
```

See also "Shutdown of the Database and CXFS" on page 300.

CXFS Filesystem Tasks with cmgr

This section tells you how to define a filesystem, specify the nodes on which it may or may not be mounted (the *enabled* or *disabled* nodes), and perform mounts and unmounts.

A given filesystem can be mounted on a given node when the following things are true:

- One of the following is true for the node:
  - The default local status is enabled and the node is not in the filesystem's list of explicitly disabled nodes
  - The default local status is disabled and the node is in the filesystem's list of explicitly enabled nodes

  See "Define a CXFS Filesystem with `cmgr`" on page 260.
- The global status of the filesystem is enabled. See "Mount a CXFS Filesystem with `cmgr`" on page 266.

Define a CXFS Filesystem with cmgr

Use the following commands to define a filesystem and the nodes on which it may be mounted:

```
define cxfs_filesystem logical_filesystem_name [in cluster clustername]
   set device_name to devicename
   set mount_point to mountpoint
   set mount_options to mount_options
   set force to true|false
   set dflt_local_status to enabled|disabled
   add cxfs_server admin_nodename
      set rank to 0|1|2|...
   add enabled_node nodename
   add disabled_node nodename
   remove cxfs_server admin_nodename
   remove enabled_node nodename
   remove disabled_node nodename
```

Usage notes:

- Relocation is disabled by default. Recovery and relocation are supported only when using standby nodes. Therefore, you should only define multiple metadata servers for a given filesystem if you are using the standby node model. See "Relocation" on page 20.
- The list of potential metadata servers for any given filesystem must all run the same operating system type.
- `cxfs_filesystem` can be any logical name. Logical names cannot begin with an underscore (_) or include any whitespace characters, and can be at most 255 characters.

  Note: Within the GUI, the default is to use the last portion of the device name; for example, for a device name of `/dev/cxvm/d76lun0s0`, the GUI will automatically supply a logical filesystem name of `d76lun0s0`. The GUI will accept other logical names defined with `cmgr` but the GUI will not allow you to modify a logical name; you must use `cmgr` to modify the logical name.

- `device_name` is the device name of an XVM volume that will be shared among all nodes in the CXFS cluster. The name must begin with `/dev/cxvm/`.
- `mount_point` is a directory to which the specified XVM volume will be attached. This directory name must begin with a slash (`/`). For more information, see the `mount` man page.
- `mount_options` are options that are passed to the `mount` command and are used to control access to the specified XVM volume. For a list of the available options, see the `fstab` man page.
- `force` controls what action CXFS takes if there are processes that have open files or current directories in the filesystem(s) that are to be unmounted. If set to `true`, then the processes will be killed and the unmount will occur. If set to `false`, the processes will not be killed and the filesystem will not be unmounted. The force option off (set to `true`) by default.
- `dflt_local_status` defines whether the filesystem can be mounted on all unspecified nodes or cannot be mounted on any unspecified nodes. You can then use the `add enabled_node` or `add disabled_node` commands as necessary to explicitly specify the nodes that differ from the default. There are multiple combinations that can have the same result.

For example, suppose you had a cluster with 10 nodes (node1 through node10). You could use the following methods:

– If you want the filesystem to be mounted on all nodes, and want it to be mounted on any nodes that are later added to the cluster, you would specify:

```
set dflt_local_status to enabled
```

– If you want the filesystem to be mounted on all nodes except node5, and want it to be mounted on any nodes that are later added to the cluster, you would specify:

```
set dflt_local_status to enabled
add disabled_node cxfs5
```

– If you want the filesystem to be mounted on all nodes except node5, and you also do **not** want it to be mounted on any nodes that are later added to the cluster, you would specify:

```
set dflt_local_status to disabled
add enabled_node cxfs1
add enabled_node cxfs2
add enabled_node cxfs3
add enabled_node cxfs4
add enabled_node cxfs6
add enabled_node cxfs7
add enabled_node cxfs8
add enabled_node cxfs9
add enabled_node cxfs10
```

– If you want the filesystem to be mounted on node5 through node10 and on any future nodes, you could specify:

```
set dflt_local_status to enabled
add disabled_node cxfs1
add disabled_node cxfs2
add disabled_node cxfs3
add disabled_node cxfs4
```

To actually mount the filesystem on the enabled nodes, see "Mount a CXFS Filesystem with cmgr" on page 266.

- cxfs_server adds or removes the specified CXFS administration node name to the list of potential metadata servers.

Note: After a filesystem has been defined in CXFS, running `mkfs` on it will cause errors to appear in the system log file. To avoid these errors, run `mkfs` before defining the filesystem in CXFS, or delete the CXFS filesystem before running `mkfs`. See "Delete a CXFS Filesystem with `cmgr`" on page 272.

The following examples shows two potential metadata servers for the `fs1` filesystem; if `cxfs6` (the preferred server, with rank 0) is not up when the cluster starts or later fails or is removed from the cluster, then `cxfs7` (rank 1) will be used. The filesystem is mounted on all nodes.

Note: Although the list of metadata servers for a given filesystem is ordered, it is impossible to predict which server will become the server during the boot-up cycle because of network latencies and other unpredictable delays.

For example, in normal mode:

```
cmgr> define cxfs_filesystem fs1 in cluster cxfs6-8

cxfs_filesystem fs1 ? set device_name to /dev/cxvm/d76lun0s0
cxfs_filesystem fs1 ? set mount_point to /mnts/fs1
cxfs_filesystem fs1 ? set force to false
cxfs_filesystem fs1 ? add cxfs_server cxfs6
Enter CXFS server parameters, when finished enter "done" or "cancel"

CXFS server - cxfs6 ? set rank to 0
CXFS server - cxfs6 ? done
cxfs_filesystem fs1 ? add cxfs_server cxfs7
Enter CXFS server parameters, when finished enter "done" or "cancel"

CXFS server - cxfs7 ? set rank to 1
CXFS server - cxfs7 ? done
cxfs_filesystem fs1 ? set dflt_local_status to enabled
cxfs_filesystem fs1 ? done
Successfully defined cxfs_filesystem fs1

cmgr> define cxfs_filesystem fs2 in cluster cxfs6-8

cxfs_filesystem fs2 ? set device_name to /dev/cxvm/d76lun0s1
cxfs_filesystem fs2 ? set mount_point to /mnts/fs2
cxfs_filesystem fs2 ? set force to false
cxfs_filesystem fs2 ? add cxfs_server cxfs8
Enter CXFS server parameters, when finished enter "done" or "cancel"

CXFS server - cxfs8 ? set rank to 0
CXFS server - cxfs8 ? done
cxfs_filesystem fs2 ? set dflt_local_status to enabled
cxfs_filesystem fs2 ? done
Successfully defined cxfs_filesystem fs2
```

For example, in prompting mode:

```
cmgr> define cxfs_filesystem fs1 in cluster cxfs6-8

(Enter "cancel" at any time to abort)

Device ? /dev/cxvm/d76lun0s0
Mount Point ? /mnts/fs1
Mount Options[optional] ?
Use Forced Unmount ? <true|false> ? false
Default Local Status <enabled|disabled> ? (enabled)

DEFINE CXFS FILESYSTEM OPTIONS

        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:1

No current servers

Server Node ? cxfs6
Server Rank ? 0

        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:1
Server Node ? cxfs7
Server Rank ? 1


        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:9
Successfully defined cxfs_filesystem fs1

cmgr> define cxfs_filesystem fs2 in cluster cxfs6-8

(Enter "cancel" at any time to abort)

Device ? /dev/cxvm/d77lun0s1
Mount Point ? /mnts/fs2
Mount Options[optional] ?
Use Forced Unmount ? <true|false> ? false
Default Local Status <enabled|disabled> ? (enabled)
```

```
DEFINE CXFS FILESYSTEM OPTIONS

        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:1

Server Node ? cxfs8
Server Rank ? 0

        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:9
Successfully defined cxfs_filesystem fs2
```

Mount a CXFS Filesystem with cmgr

To mount a filesystem on the enabled nodes, enter the following:

`admin cxfs_mount cxfs_filesystem` *logical_filesystem_name* `[on node` *nodename*`] [in cluster` *clustername*`]`

This command enables the *global status* for a filesystem; if you specify the *nodename*, it enables the *local status*. (The global status is only affected if a node name is not specified.) For a filesystem to mount on a given node, both global and local status must be enabled; see "CXFS Filesystem Tasks with `cmgr`" on page 260.

Nodes must first be enabled by using the `define cxfs_filesystem` and `modify cxfs_filesystem` commands; see "Define a CXFS Filesystem with `cmgr`" on page 260, and "Modify a CXFS Filesystem with `cmgr`" on page 268.

For example, to activate the `f1` filesystem by setting the global status to `enabled`, enter the following:

```
cmgr> admin cxfs_mount cxfs_filesystem fs1 in cluster cxfs6-8
```

The filesystem will then be mounted on all the nodes that have a local status of `enabled` for this filesystem.

To change the local status to `enabled`, enter the following:

```
cmgr> admin cxfs_mount cxfs_filesystem fs1 on node cxfs7 in cluster cxfs6-8
```

If the filesystem's global status is `disabled`, nothing changes. If the filesystem's global status is `enabled`, the node will mount the filesystem as the result of the change of its local status.

Note: If CXFS services are not active, mounting a filesystem will not completely succeed. The filesystem will be marked as ready to be mounted but the filesystem will not actually be mounted until you have started CXFS services. For more information, see "Start CXFS Services with `cmgr`" on page 254.

Unmount a CXFS Filesystem with `cmgr`

To unmount a filesystem, enter the following:

```
admin cxfs_unmount cxfs_filesystem filesystemname [on node nodename] [in cluster clustername]
```

Unlike the `modify cxfs_filesystem` command, this command can be run on an active filesystem.

For example, to deactivate the `f1` filesystem by setting the global status to `disabled`, enter the following:

```
cmgr> admin cxfs_unmount cxfs_filesystem fs1 in cluster cxfs6-8
```

The filesystem will then be unmounted on all the nodes that have a local status of `enabled` for this filesystem.

To change the local status to `disabled`, enter the following:

```
cmgr> admin cxfs_unmount cxfs_filesystem fs1 on node cxfs7 in cluster cxfs6-8
```

If the filesystem's global status is `disabled`, nothing changes. If the filesystem's global status is `enabled`, the node will unmount the filesystem as the result of the change of its local status.

Modify a CXFS Filesystem with cmgr

Note: You cannot modify a mounted filesystem.

Use the following commands to modify a filesystem:

```
modify cxfs_filesystem logical_filesystem_name [in cluster clustername]
   set device_name to devicename
   set mount_point to mountpoint
   set mount_options to options
   set force to true|false
   set dflt_local_status to enabled|disabled
   add cxfs_server servername
     set rank to 0|1|2|...
   modify cxfs_server servername
     set rank to 0|1|2|...
   add enabled_node nodename
   add disabled_node nodename
   remove cxfs_server nodename
   remove enabled_node nodename
   remove disabled_node nodename
```

These are the same commands used to define a filesystem; for more information, see "Define a CXFS Filesystem with `cmgr`" on page 260.

For example, in normal mode:

```
cmgr> show cxfs_filesystem fs1 in cluster cxfs6-8

Name: fs1
```

```
Device: /dev/cxvm/d76lun0s0
Mount Point: /mnts/fs1
Forced Unmount: false
Global Status: disabled
Default Local Status: enabled

Server Name: cxfs6
       Rank: 0
Server Name: cxfs7
       Rank: 1
Disabled Client: cxfs8

cmgr> modify cxfs_filesystem fs1 in cluster cxfs6-8
Enter commands, when finished enter either "done" or "cancel"

cxfs_filesystem fs3 ? modify cxfs_server cxfs6
Enter CXFS server parameters, when finished enter "done" or "cancel"

Current CXFS server cxfs6 parameters:
        rank : 0
CXFS server - cxfs6 ? set rank to 2
CXFS server - cxfs6 ? done
cxfs_filesystem fs1 ? done

Successfully modified cxfs_filesystem fs1
cmgr> show cxfs_filesystem fs1 in cluster cxfs6-8

Name: fs1
Device: /dev/cxvm/d76lun0s0
Mount Point: /mnts/fs1
Forced Unmount: false
Global Status: disabled
Default Local Status: enabled

Server Name: cxfs6
       Rank: 2
Server Name: cxfs7
       Rank: 1
Disabled Client: cxfs8
```

In prompting mode:

```
cmgr> show cxfs_filesystem fs1 in cluster cxfs6-8

Name: fs1
Device: /dev/cxvm/d76lun0s0
Mount Point: /mnts/fs1
Forced Unmount: false
Global Status: disabled
Default Local Status: enabled

Server Name: cxfs6
       Rank: 0
Server Name: cxfs7
       Rank: 1
Disabled Client: cxfs8

cmgr> modify cxfs_filesystem fs1 in cluster cxfs6-8

(Enter "cancel" at any time to abort)

Device ? (/dev/cxvm/d76lun0s0)
Mount Point ? (/mnts/fs1)
Mount Options[optional] ?
Use Forced Unmount ? <true|false>  ? (false)
Default Local Status <enabled|disabled> ? (enabled)

MODIFY CXFS FILESYSTEM OPTIONS

        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:0
```

```
Current servers:
CXFS Server 1 - Rank: 0         Node: cxfs6
CXFS Server 2 - Rank: 1         Node: cxfs7

Server Node ? cxfs6
Server Rank ? (0) 2

        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:7

Current settings for filesystem (fs1)

CXFS servers:
        Rank 2          Node cxfs6
        Rank 1          Node cxfs7

Default local status: enabled

No explicitly enabled clients

Explicitly disabled clients:
        Disabled Node: cxfs8

        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:9
Successfully modified cxfs_filesystem fs3
```

Relocate the Metadata Server for a Filesystem with `cmgr`

If relocation is explicitly enabled in the kernel with the `cxfs_relocation_ok` systune (see "Relocation" on page 20), you can relocate a metadata server to another node using the following command if the filesystem must be mounted on the system that is running `cmgr`:

```
admin cxfs_relocate cxfs_filesystem filesystem_name to node nodename [in cluster clustername]
```

Note: This function is only available on a live system.

To relocate the metadata server from `cxfs6` to `cxfs7` for `fs1` in cluster `cxfs6-8`, enter the following:

```
cmgr> admin cxfs_relocate cxfs_filesystem fs1 to node cxfs7 in cluster cxfs6-8
```

CXFS kernel membership is not affected by relocation. However, users may experience a degradation in filesystem performance while the metadata server is relocating.

For more details, see "Modify a CXFS Filesystem with `cmgr`" on page 268.

Delete a CXFS Filesystem with `cmgr`

Use the following command to delete a filesystem:

```
delete cxfs_filesystem filesystemname [in cluster clustername]
```

For example:

```
cmgr> delete cxfs_filesystem fs2 in cluster cxfs6-8
```

Switches and I/O Fencing Tasks with `cmgr`

The following tasks let you configure switches and I/O fencing. For general information, see "I/O Fencing" on page 30.

Note: Nodes without system controllers require I/O fencing to protect data integrity. A switch is mandatory to support I/O fencing; therefore, multiOS CXFS clusters require a switch. See the release notes for supported switches.

Define a Switch with `cmgr`

This section describes how to use the `cmgr` command to define a new Brocade switch to support I/O fencing in a cluster.

Note: To define a switch other than a Brocade switch, such as a QLogic switch, you must use the `hafence`(1M) command. (You cannot use the `cmgr` command to completely define a switch other than Brocade.) See "Configuring Switches Other than Brocade" on page 288.

To define a new Brocade switch, use the following command:

`define switch` *switch_hostname* `username` *username* `password` *password* `[mask` *mask*`]`

Usage notes:

- `switch` is the hostname of the Fibre Channel switch; this is used to determine the IP address of the switch.
- `username` is the user name to use when sending a `telnet` message to the switch.
- `password` is the password for the specified *username*.
- `mask` is a hexadecimal string that represents which ports in the switch are eligible for fencing. Ports are numbered from 0. If a given bit has a binary value of 0, the port that corresponds to that bit is eligible for fencing operations; if 1, then the port that corresponds to that bit will always be excluded from any fencing operations. For an example, see Figure 9-5 on page 198.

  Note: You can only mask ports 0 through 63.

CXFS administration nodes automatically discover the available HBAs and, when fencing is triggered, fence off all of the Fibre Channel HBAs when the `Fence` or `FenceReset` fail action is selected. However, masked HBAs will not be fenced. Masking allows you to prevent the fencing of devices that are attached to the SAN but are not shared with the cluster, to ensure that they remain available regardless of CXFS status. You would want to mask HBAs used for access to tape storage, or HBAs that are only ever used to access local (nonclustered) devices.

For example:

```
cmgr> define switch ptg-brocade username admin password password mask A4
```

Modify a Switch Definition with cmgr

To modify the user name, password, or mask for a Brocade switch, use the following command:

```
modify switch switch_hostname username username password password [mask mask]
```

The arguments are the same as for "Define a Switch with `cmgr`" on page 273.

Note: To modify the definition of another type of switch, such as QLogic, you must use the `hafence`(1M) command. See "Configuring Switches Other than Brocade" on page 288.

For example, to change the mask for switch `ptg-brocade` from `A4` to `0` (which means that all of the ports on the switch are eligible for fencing), enter the following:

```
cmgr> modify switch ptg-brocade username admin password password mask 0
```

Raise the I/O Fence for a Node with cmgr

Raising an I/O fence isolates the node from the SAN; CXFS sends a messages via the `telnet` protocol to the switch and disables the port. After the node is isolated, it cannot corrupt data in the shared CXFS filesystem. Use the following command:

```
admin fence raise [node nodename]
```

nodename is the name of the node to be isolated.

For example, to isolate the default node, enter the following:

```
cmgr> admin fence raise
```

To isolate node `Node3`, enter the following:

```
cmgr> admin fence raise node Node3
```

Lower the I/O Fence for a Node with `cmgr`

To lower the I/O fence for a given node in order to reenable the port, allowing the node to connect to the SAN and access the shared CXFS filesystem, use the following command:

```
admin fence lower [node nodename]
```

nodename is the name of the node to be reconnected.

For example, to provide access for the default node, enter the following:

```
cmgr> admin fence lower
```

To provide access for node `Node3`, enter the following:

```
cmgr> admin fence lower node Node3
```

Update Switch Port Information with `cmgr`

To update the mappings in the cluster database between the host bus adapters (HBAs) and switch ports, use the following command:

```
admin fence update
```

You should run this command if you reconfigure any switch or add ports.

Delete a Switch Definition with `cmgr`

To delete a switch, use the following command:

```
delete switch switch_hostname
```

switch_hostname is the hostname of the Fibre Channel switch; this is used to determine the IP address of the switch.

For example:

```
cmgr> delete switch ptg-brocade
Successfully updated switch config.
```

Show Switches with `cmgr`

To display the switches in the system, use the following command:

```
show switches
```

To show the switches for a given node, use the following command:

`show switch` *hostname*

For example:

```
cmgr> show switch ptg-brocade
  Switch[0]
      *Hostname ptg-brocade Username admin Password password Mask 0
      Vendor BROCADE Number of ports 8
            0 0000000000000000 Reset
            1 210000e08b0102c6 Reset
            2 210000e08b01fec5 Reset
            3 210000e08b019dc5 Reset
            4 210000e08b0113ce Reset
            5 210000e08b027795 Reset thump
            6 210000e08b019ef0 Reset
            7 210000e08b022242 Reset
```

Query Switch Status with `cmgr`

To query the status of each port on the switch, use the following command:

```
admin fence query
```

For example:

```
cmgr> admin fence query
  Switch[0] "brocade04" has 16 ports
    Port 4 type=FABRIC status=enabled  hba=210000e08b0042d8 on host o200c
    Port 5 type=FABRIC status=enabled  hba=210000e08b00908e on host cxfs30
    Port 9 type=FABRIC status=enabled  hba=2000000173002d3e on host cxfssun3
```

For more verbose display, (which shows all ports on the switch, rather than only those attached to nodes in the default cluster), use the following command:

```
admin fence query verbose
```

For example:

```
cmgr> admin fence query verbose
  Switch[0] "brocade04" has 16 ports
    Port 0 type=FABRIC status=enabled  hba=2000000173003b5f on host UNKNOWN
    Port 1 type=FABRIC status=enabled  hba=2000000173003adf on host UNKNOWN
    Port 2 type=FABRIC status=enabled  hba=210000e08b023649 on host UNKNOWN
    Port 3 type=FABRIC status=enabled  hba=210000e08b021249 on host UNKNOWN
    Port 4 type=FABRIC status=enabled  hba=210000e08b0042d8 on host o200c
    Port 5 type=FABRIC status=enabled  hba=210000e08b00908e on host cxfs30
    Port 6 type=FABRIC status=enabled  hba=2000000173002d2a on host UNKNOWN
    Port 7 type=FABRIC status=enabled  hba=2000000173003376 on host UNKNOWN
    Port 8 type=FABRIC status=enabled  hba=2000000173002c0b on host UNKNOWN
    Port 9 type=FABRIC status=enabled  hba=2000000173002d3e on host cxfssun3
    Port 10 type=FABRIC status=enabled  hba=2000000173003430 on host UNKNOWN
    Port 11 type=FABRIC status=enabled  hba=200900a0b80c13c9 on host UNKNOWN
    Port 12 type=FABRIC status=disabled hba=0000000000000000 on host UNKNOWN
    Port 13 type=FABRIC status=enabled  hba=200d00a0b80c2476 on host UNKNOWN
    Port 14 type=FABRIC status=enabled  hba=1000006069201e5b on host UNKNOWN
    Port 15 type=FABRIC status=enabled  hba=1000006069201e5b on host UNKNOWN
```

Script Example

The following script defines a three-node cluster of type CXFS. The nodes are of type CXFS.

Note: This example only defines one network interface. The hostname is used here for simplicity; however, you may wish to use the IP address instead to avoid confusion. This example does not address the system controller definitions.

```
#!/usr/cluster/bin/cmgr -if
#
#Script to define a three-node cluster


define node cxfs6
        set hostname to cxfs6
        set is_cxfs to true
        set operating_system to irix
        set node_function to server_admin
        add nic cxfs6
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 1
        done
done

define node cxfs7
        set hostname to cxfs7
        set is_cxfs to true
        set operating_system to irix
        set node_function to server_admin
        add nic cxfs7
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 1
        done
done

define node cxfs8
        set hostname to cxfs8
        set is_cxfs to true
        set operating_system to irix
        set node_function to server_admin
        add nic cxfs8
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 1
        done
done
```

```
define cluster cxfs6-8
        set is_cxfs to true
        set is_failsafe to true
        set clusterid to 20
        add node cxfs6
        add node cxfs7
        add node cxfs8
        done
quit
```

After running this script, you would see the following output:

```
Successfully defined node cxfs6

Successfully defined node cxfs7

Successfully defined node cxfs8

Successfully defined cluster cxfs6-8
```

The following script defines two filesystems; `fs1` is mounted on all but node `cxfs8`, and `fs2` is mounted on all nodes:

```
#!/usr/cluster/bin/cmgr -if
# Script to define two filesystems
# Define fs1, do not mount on cxfs8
define cxfs_filesystem fs1 in cluster cxfs6-8
set device_name to /dev/cxvm/d76lun0s0
set mount_point to /mnts/fs1
set force to false
add cxfs_server cxfs6
  set rank to 0
  done
add cxfs_server cxfs7
  set rank to 1
  done
set dflt_local_status to enabled
add disabled_node cxfs8
done
#
# Define fs2, mount everywhere
define cxfs_filesystem fs2 in cluster cxfs6-8
```

```
set device_name to /dev/cxvm/d76lun0s1
set mount_point to /mnts/fs2
set force to false
add cxfs_server cxfs8
set rank to 0
done
set dflt_local_status to enabled
done
```

Creating a `cmgr` Script Automatically

After you have configured the cluster database, you can use the `build_cmgr_script` command to automatically create a `cmgr` script based on the contents of the cluster database. The generated script will contain the following:

- Node definitions
- Cluster definition
- Switch definitions
- CXFS filesystem definitions
- Parameter settings
- Any changes made using either the `cmgr` command or the GUI
- FailSafe information (in a coexecution cluster only)

As needed, you can then use the generated script to recreate the cluster database after performing a `cdbreinit`.

Note: You must execute the generated script on the first node that is listed in the script. If you want to execute the generated script on a different node, you must modify the script so that the node is the first one listed.

By default, the generated script is named:

`/var/cluster/ha/tmp/cmgr_create_cluster_`*clustername_processID*

You can specify an alternative pathname by using the `-o` option:

`build_cmgr_script [-o` *script_pathname*`]`

For more details, see the `build_cmgr_script` man page.

For example:

```
# /var/cluster/cmgr-scripts/build_cmgr_script -o /tmp/newcdb
Building cmgr script for cluster clusterA ...
build_cmgr_script: Generated cmgr script is /tmp/newcdb
```

The example script file contents are as follows; note that because `nodeE` is the first node defined, you must execute the script on `nodeE`:

```
#!/usr/cluster/bin/cmgr -f

# Node nodeE definition
define node nodeE
        set hostname to nodeE.americas.sgi.com
        set operating_system to IRIX
        set is_failsafe to false
        set is_cxfs to true
        set node_function to server_admin
        set nodeid to 5208
        set reset_type to powerCycle
        add nic nodeE
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 1
        done
done

# Node nodeD definition
define node nodeD
        set hostname to nodeD.americas.sgi.com
        set operating_system to IRIX
        set is_failsafe to false
        set is_cxfs to true
        set node_function to server_admin
        set nodeid to 5181
        set reset_type to powerCycle
        add nic nodeD
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 1
        done
done

# Node nodeF definition
define node nodeF
        set hostname to nodeF.americas.sgi.com
        set operating_system to IRIX
        set is_failsafe to false
        set is_cxfs to true
        set node_function to server_admin
        set nodeid to 5401
        set reset_type to powerCycle
        add nic nodeF
                set heartbeat to true
                set ctrl_msgs to true
                set priority to 1
        done
done

# Define cluster and add nodes to the cluster
define cluster clusterA
        set is_failsafe to false
        set is_cxfs to true
        set cx_mode to normal
        set clusterid to 35
done

modify cluster clusterA
        add node nodeD
        add node nodeF
        add node nodeE
done

set cluster clusterA

define cxfs_filesystem fs1
        set device_name to /dev/cxvm/fs1
        set mount_point to /fs1
        set force to false
        set dflt_local_status to enabled
        add cxfs_server nodeE
```

```
                set rank to 1
        done
        add cxfs_server nodeD
                set rank to 2
        done
        add cxfs_server nodeF
                set rank to 0
        done
done

define cxfs_filesystem fs2
        set device_name to /dev/cxvm/fs2
        set mount_point to /fs2
        set force to false
        set dflt_local_status to enabled
        add cxfs_server nodeE
                set rank to 1
        done
        add cxfs_server nodeD
                set rank to 2
        done
        add cxfs_server nodeF
                set rank to 0
        done
done

define cxfs_filesystem fs2
        set device_name to /dev/cxvm/fs2
        set mount_point to /fs2
        set force to false
        set dflt_local_status to enabled
        add cxfs_server nodeE
                set rank to 1
        done
        add cxfs_server nodeD
                set rank to 2
        done
        add cxfs_server nodeF
                set rank to 0
        done
done
```

```
# Setting CXFS parameters
modify cx_parameters
        set tie_breaker to none
done

quit
```

Administration and Maintenance

You can perform offline administration tasks using the `cmgr` command when logged into any CXFS administration node (one that is installed with the `cxfs_cluster` product) in the pool, or when the GUI is connected to any CXFS administration node in the pool. However, when the filesystems are mounted, administration must be done from the metadata server. (You cannot use `cmgr` or connect the GUI to a client-only node.)

Note: You should perform reconfiguration and/or cluster manipulation (such as adding or deleting filesystems or nodes) on a scheduled cluster maintenance shift and not during production hours. You should stop CXFS services on an administration node before performing maintenance on a node.

The following are the same in CXFS and XFS:

- Disk concepts
- Filesystem concepts
- User interface
- Filesystem creation

For more information about these topics, see *IRIX Admin: Disks and Filesystems*.

The rest of this chapter discusses the following topics:

- "CXFS and Cluster Administration Initialization Commands" on page 287
- "Configuring Switches Other than Brocade" on page 288
- "CXFS `chkconfig` Arguments" on page 289
- "Granting Task Execution Privileges to Users" on page 291
- "Transforming an Existing Node into a Client-Only Node" on page 292
- "CXFS Mount Scripts" on page 293
- "Using `telnet` and I/O Fencing" on page 295
- "Using `fsr` and `xfs_fsr`" on page 296

- "Using `cron` in a CXFS Cluster" on page 296
- "Using Hierarchical Storage Management (HSM) Products" on page 296
- "Discovering the Active Metadata Server for a Filesystem" on page 297
- "Metadata Server Recovery" on page 299
- "Shutdown of the Database and CXFS" on page 300
- "Avoiding a CXFS Restart at Reboot" on page 306
- "Log File Management" on page 307
- "Volume Management" on page 308
- "Disk Management" on page 309
- "Filesystem Maintenance" on page 311
- "Dump and Restore" on page 313
- "System Tunable Parameters" on page 315
- "Hardware Changes and I/O Fencing" on page 321
- "Configuring Private Network Failover" on page 322
- "Removing and Restoring Cluster Members" on page 329
- "Discovering the WWNs" on page 335

See also Chapter 12, "Cluster Database Management" on page 337.

Note: If you have upgraded directly from IRIX 6.5.12f or earlier, you must manually convert you filesystem definitions to the new format. See "IRIX: Converting Filesystem Definitions for Upgrades" on page 109.

CXFS and Cluster Administration Initialization Commands

Table 11-1 summarizes the `/etc/init.d` initialization commands used for the CXFS control daemon and the cluster administration daemons. Paths may differ between IRIX and Linux systems.

Table 11-1 CXFS and Cluster Administration Initialization Commands wide

IRIX	Linux	Description
`/etc/init.d/cluster start`	`/etc/init.d/cxfs_cluster start`	Starts the `fs2d`, `cmond`, `cad`, and `crsd` (the cluster administration daemons) on the local node
`/etc/init.d/cxfs start`	`/etc/init.d/cxfs start`	Starts `clconfd` (the CXFS control daemon) on the local node
`/etc/init.d/cluster stop`	`/etc/init.d/cxfs_cluster stop`	Stops `fs2d`, `cmond`, `cad`, and `crsd` on the local node
`/etc/init.d/cxfs stop`	`/etc/init.d/cxfs stop`	Stops CXFS in the kernel (which withdraws membership) and `clconfd` on the local node
`/etc/init.d/cluster restart`	`/etc/init.d/cxfs_cluster restart`	Restarts the cluster administration daemons on the local node
`/etc/init.d/cxfs restart`	`/etc/init.d/cxfs start`	Restarts `clconfd` on the local node

IRIX	Linux	Description
`/etc/init.d/cluster status`	`/etc/init.d/cxfs_cluster status`	Gives status (`running` or `stopped`) of `fs2d`, `cmond`, `cad`, and `crsd` on the local node
`/etc/init.d/cxfs status`	`/etc/init.d/cxfs status`	Gives status (`running` or `stopped`) of `clconfd` on the local node

Configuring Switches Other than Brocade

To define or modify a switch other than a Brocade switch, such as a QLogic switch, you must use the `hafence` command:

```
/usr/cluster/bin/hafence -a -s switchhost -u username -p password -m mask -L vendorlibrary
```

For example, the following defines a QLogic switch named `myqlswitch` and uses no masking:

```
# /usr/cluster/bin/hafence -a -s myqlswitch -u admin -p ***  -m 0 -L qlogic
```

The above command line will attempt to load the `libcrf_vendor.so` library, which must be installed in a directory that is searched by `dlopen`(3), which is usually `/usr/lib` on Linux systems and `/usr/lib32` on IRIX systems. However, the shared library search path is platform dependent and site configurable; therefore, it may be somewhere else if the `LD_LIBRARY_PATH` environment variable has been set. See the `dlopen`(3) man page for details.

For more information, see the `hafence`(1M) man page. See the release notes for supported switches.

Note: You cannot use the GUI or the `cmgr` command to define or modify a switch other than a Brocade switch.

CXFS `chkconfig` Arguments

Table 11-2 summarizes the CXFS `chkconfig` arguments for IRIX and Linux nodes. These settings are not normally manipulated by the administrator; they are set or unset by the GUI or `cmgr`. These settings only control the processes, not the cluster. Stopping the processes that control the cluster will not stop the cluster (that is, will not drop the cluster membership or lose access to CXFS filesystems and cluster volumes), and starting the processes will start the cluster **only** if the CXFS services are marked as activated in the database.

Note: `cxfs_cluster` controls different daemons on IRIX than it does on Linux.

On Linux nodes, `chkconfig` settings are saved by updating various symbolic links in the `/etc/rc.`*n* directories.

The following shows the settings of the arguments on IRIX and Linux administration nodes:

- IRIX:

```
irix# chkconfig | grep cluster
        cluster             on
        cxfs_cluster        on
```

- Linux:

```
[root@linux root]# chkconfig --list | grep cxfs
cxfs_cluster    0:off   1:off   2:on    3:on    4:on    5:on    6:off
cxfs            0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

Table 11-2 CXFS `chkconfig` Arguments

IRIX Admin	Linux Admin	Client-Only (IRIX or Linux)	Description
`cluster`	`cxfs_cluster`	*N/A*	Controls the cluster administration daemons (`fs2d`, `crsd`, `cad`, and `cmond`). If this argument is turned `off`, the database daemons will not be started at the next reboot and the local copy of the database will not be updated if you make changes to the cluster configuration on the other nodes. This could cause problems later, especially if a majority of nodes are not running the database daemons. If the database daemons are not running, the cluster database will not be accessible locally and the node will not be configured to join the cluster.
`cxfs_cluster`	`cxfs`	*N/A*	Controls the `clconfd` daemon and whether or not the `cxfs_shutdown` command is used during a system shutdown. The `cxfs_shutdown` command attempts to withdraw from the cluster gracefully before rebooting. Otherwise, the reboot is seen as a failure and the other nodes have to recover from it. **Note:** `clconfd` cannot start unless `fs2d` is already running.
N/A	*N/A*	`cxfs_client`	Controls whether or not the `cxfs_client` daemon should be started

Configuring Real-Time Filesystems For IRIX Nodes

CXFS can write to real-time files in real-time volumes on IRIX nodes. For more details about real-time volumes, see the *XVM Volume Manager Administrator's Guide*.

When creating the CXFS filesystem, be aware of the following:

- To maintain appropriate performance of the real-time filesystem, do not flag unwritten extents. Use the following command:

  ```
  irix# mkfs_xfs -d unwritten=0
  ```

- Set the real-time extent size to a large value for maximum performance.This parameter should be a multiple of the basic filesystem block size, and can vary between 4 KB to 1 GB. SGI recommends 128 MB. You can set this value with the following command:

  ```
  irix# mkfs_xfs -r extsize=size_of_real-time_extent
  ```

- Use a large value for block size. Block size can vary between 512 bytes to 64 KB. SGI recommends 16 KB to allow all nodes other than Linux 32-bit to access the filesystem.

  Caution: Linux systems are not capable of accessing filesystems with block size larger than the system page size. (The default page sizes are as follows: 4 KB for Linux 32-bit, 16 KB for Linux.)

  Therefore, if the real-time filesystem is to be accessible by all possible nodes in the cluster, its block size would have to be the lowest possible common denominator (4 KB).

  You can set this value with the following command:

  ```
  irix# mkfs_xfs -b size=blocksize
  ```

Granting Task Execution Privileges to Users

The GUI lets you grant or revoke access to a specific GUI task for one or more specific users. By default, only `root` may execute tasks in the GUI. Access to the task is only allowed on the node to which the GUI is connected; if you want to allow access on another node in the pool, you must connect the GUI to that node and grant access again.

Note: You cannot grant or revoke tasks for users with a user ID of 0.

GUI tasks and the `cmgr` command operate by executing underlying privileged commands which are normally accessible only to `root`. When granting access to a task, you are in effect granting access to all of its required underlying commands, which results in also granting access to the other GUI tasks that use the same underlying commands.

For instructions about granting or revoking GUI privileges, see "Privileges Tasks with the GUI" on page 211.

To see which tasks a specific user can currently access, select **View: Users**. Select a specific user to see details about the tasks available to that user.

To see which users can currently access a specific task, select **View: Task Privileges**. Select a specific task to see details about the users who can access it and the privileged commands it requires.

Transforming an Existing Node into a Client-Only Node

If you are upgrading to 6.5.19f from 6.5.17f or earlier and you want to change an existing node with weight 1 (which as of 6.5.18f was defined as a *server-capable administration node*) to be a client-only node, you must do the following:

1. Ensure that the node is not listed as a potential metadata server for any filesystem. See "Modify a CXFS Filesystem with the GUI" on page 207, or "Modify a CXFS Filesystem with `cmgr`" on page 268.

2. Stop the CXFS services on the node. See "Stop CXFS Services with the GUI" on page 191 or "Stop CXFS Services with `cmgr`" on page 254.

3. Modify the cluster so that it no longer contains the node. See "Modify a Cluster Definition with the GUI" on page 189, or "Modify a Cluster with `cmgr`" on page 250.

4. Delete the node definition. See "Delete a Node with the GUI" on page 186, or "Delete a Node with `cmgr`" on page 241.

5. Install the node with the `cxfs_client` package and remove the `cluster_admin`, `cluster_control`, and `cluster_services` packages. See"IRIX Client-only Software Installation" on page 76.

6. Reboot the node to ensure that all previous node configuration information is removed.
7. Redefine the node and use a node function of client-only. See "Define a Node with the GUI" on page 172, or "Define a Node with `cmgr`" on page 224.
8. Modify the cluster so that it contains the node. See "Modify a Cluster Definition with the GUI" on page 189, or "Modify a Cluster with `cmgr`" on page 250.
9. Start the CXFS services on the node. See "Start CXFS Services with the GUI" on page 191, or "Start CXFS Services with `cmgr`" on page 254.

CXFS Mount Scripts

Scripts are provided for execution prior to and after a CXFS filesystem is mounted or unmounted on the following platforms:

- On server-capable nodes:

  ```
  /var/cluster/clconfd-scripts/cxfs-pre-mount
  /var/cluster/clconfd-scripts/cxfs-post-mount
  /var/cluster/clconfd-scripts/cxfs-pre-umount
  /var/cluster/clconfd-scripts/cxfs-post-umount
  ```

  The `clconfd` daemon executes the above scripts.

- On client-only nodes:

  ```
  /var/cluster/cxfs_client-scripts/cxfs-pre-mount
  /var/cluster/cxfs_client-scripts/cxfs-post-mount
  /var/cluster/cxfs_client-scripts/cxfs-pre-umount
  /var/cluster/cxfs_client-scripts/cxfs-post-umount
  ```

  The `cxfs_client` daemon executes the above scripts.

The scripts are used by CXFS to ensure that LUN path failover works properly after fencing by executing the following:

```
/etc/init.d/failover stop
/etc/init.d/failover start
```

These scripts can be customized to suit a particular environments. For example, an application could be started when a CXFS filesystem is mounted by extending the `cxfs-post-mount` script. The application could be terminated by changing the `cxfs-pre-umount` script.

On IRIX and SGI ProPack for Linux nodes, these scripts also allow you to use NFS to export the CXFS filesystems listed in `/etc/exports` if they are successfully mounted.

The appropriate daemon executes these scripts before and after mounting or unmounting CXFS filesystems specified in the `/etc/exports` file. The files must be named **exactly** as above and must have `root` execute permission.

Note: The `/etc/exports` file describes the filesystems that are being exported to NFS clients. If a CXFS mount point is included in the `exports` file, the empty mount point is exported unless the filesystem is re-exported after the CXFS mount using the `cxfs-post-mount` script.

The `/etc/exports` file cannot contain any filesystems managed by FailSafe.

The following arguments are passed to the files:

- `cxfs-pre-mount`: filesystem device name
- `cxfs-post-mount`: filesystem device name and exit code
- `cxfs-pre-umount`: filesystem device name
- `cxfs-post-umount`: filesystem device name and exit code

Because the filesystem name is passed to the scripts, you can write the scripts so that they take different actions for different filesystems; because the exit codes are passed to the `post` files, you can write the scripts to take different actions based on success or failure of the operation.

The `clconfd` or `cxfs_client` daemon checks the exit code for these scripts. In the case of failure (nonzero), the following occurs:

- For `cxfs-pre-mount` and `cxfs-pre-umount`, the corresponding mount or unmount is not performed.
- For `cxfs-post-mount` and `cxfs-post-umount`, `clconfd` will retry the entire operation (including the `-pre-` script) for that operation.

This implies that if you **do not** want a filesystem to be mounted on a host, the `cxfs-pre-mount` script should return a failure for that filesystem while the `cxfs-post-mount` script returns success.

The following script is run when needed to reprobe the Fibre Channel controllers:

- On server-capable nodes:

  ```
  /var/cluster/clconfd-scripts/cxfs-reprobe
  ```

- On client-only nodes:

  ```
  /var/cluster/cxfs_client-scripts/cxfs-reprobe
  ```

You may modify any of these scripts if needed.

Unmounting `lofs` File Systems

You must unmount `lofs` mounts of a CXFS filesystem before attempting to unmount the CXFS filesystem. You can use a script such as the following to unexport and locally unmount an `lofs` filesystem:

```
#!/bin/ksh
#/var/cluster/clconfd-scripts/cxfs-pre-umount
echo "$0: Preparing to unmount CXFS file system \"$1\""
MNTPNT=‘mount | grep "$1 " | cut -f 3 -d" "‘
print "MNTPNT $MNTPNT"
if [ -n "${MNTPNT}" ] ; then
    lofslist=‘mount | grep ’type lofs’ | grep "${MNTPNT}" | nawk ’{print $3}’‘
    set -e
    for lofs in ${lofslist}
    do
        echo "$0: unmounting $lofs"
        umount -k $lofs
    done
    if /usr/etc/exportfs | /sbin/grep -q "${MNTPNT}" ; then
        echo "$0: unexporting $MNTPNT"
        /usr/etc/exportfs -u ${MNTPNT}
    fi
fi
```

Using `telnet` and I/O Fencing

If there are problems with a node, the I/O fencing software sends a message via the `telnet` protocol to the appropriate Fibre Channel switch. The switch only allows one `telnet` session at a time; therefore, if you are using I/O fencing, you must keep the `telnet` port on the Fibre Channel switch free at all times. **Do not** perform a `telnet` to the switch and leave the session connected.

Using `fsr` and `xfs_fsr`

The IRIX `fsr` and the Linux `xfs_fsr` commands can **only** be used on the active metadata server for the filesystem; the `bulkstat` system call has been disabled for CXFS clients. You should use `fsr` or `xfs_fsr` manually, and only on the active metadata server for the filesystem.

Using `cron` in a CXFS Cluster

The `cron` daemon can cause severe stress on a CXFS filesystem if multiple nodes in a cluster start the same filesystem-intensive task simultaneously. An example of such a task is one that uses the `find` command to search files in a filesystem.

Any task initiated using `cron` on a CXFS filesystem should be launched from a single node in the cluster, preferably from the active metadata server.

Using Hierarchical Storage Management (HSM) Products

CXFS supports the use of hierarchical storage management (HSM) products through the data management application programming interface (DMAPI), also know as X/Open Data Storage Management Specification (XSDM). An example of an HSM product is the Data Migration Facility (DMF). DMF is the only HSM product currently supported with CXFS.

Note: CXFS does not support the relocation or recovery of DMAPI filesystems that are being served by Linux metadata servers.

The HSM application must make all of its DMAPI interface calls through the active metadata server. The CXFS client nodes do not provide a DMAPI interface to CXFS mounted filesystems. A CXFS client routes all of its communication to the HSM application through the metadata server. This generally requires that the HSM application run on the CXFS metadata server.

To use HSM with CXFS, do the following:

- Install `eoe.sw.dmi` on each CXFS administration node. For client-only nodes, no additional software is required.
- Use the `dmi` option when mounting a filesystem to be managed. For more information about this step, see "Define CXFS Filesystems with the GUI" on page 204, or "Modify a Cluster with `cmgr`" on page 250.
- Start the HSM application on the active metadata server for each filesystem to be managed.

Discovering the Active Metadata Server for a Filesystem

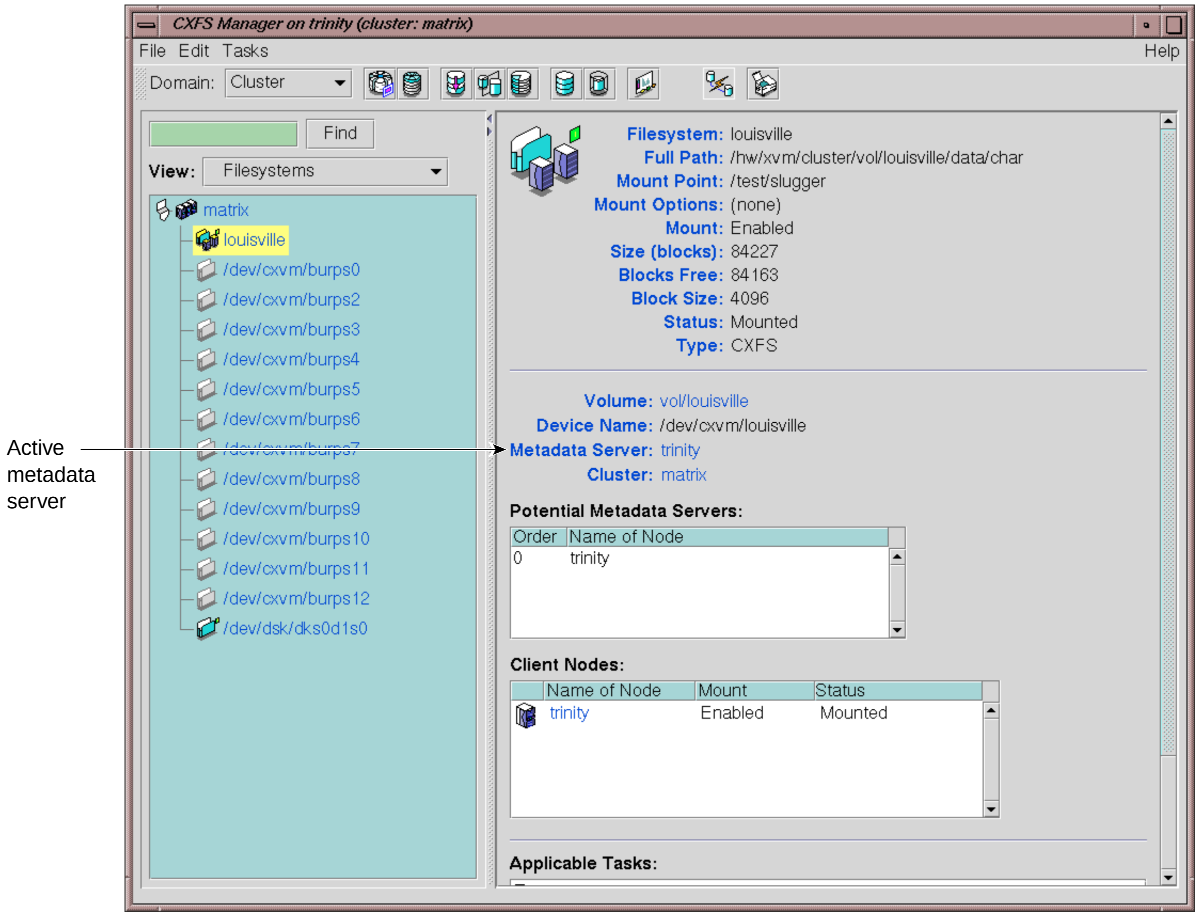
You can discover the active metadata server using the GUI or the `clconf_info` command.

Metadata Server Discovery with the GUI

Do the following:

1. Select **View: Filesystems**
2. In the view area, click the name of the filesystem you wish to view. The name of the active metadata server is displayed in the details area to the right.

Figure 11-1 shows an example.



matrix
- louisville
- /dev/cxvm/burps0
- /dev/cxvm/burps2
- /dev/cxvm/burps3
- /dev/cxvm/burps4
- /dev/cxvm/burps5
- /dev/cxvm/burps6
- /dev/cxvm/burps7
- /dev/cxvm/burps8
- /dev/cxvm/burps9
- /dev/cxvm/burps10
- /dev/cxvm/burps11
- /dev/cxvm/burps12
- /dev/dsk/dks0d1s0

Filesystem: louisville
Full Path: /hw/xvm/cluster/vol/louisville/data/char
Mount Point: /test/slugger
Mount Options: (none)
Mount: Enabled
Size (blocks): 84227
Blocks Free: 84163
Block Size: 4096
Status: Mounted
Type: CXFS

Volume: vol/louisville
Device Name: /dev/cxvm/louisville
Metadata Server: trinity
Cluster: matrix

Potential Metadata Servers:

Order	Name of Node
0	trinity

Client Nodes:

Name of Node	Mount	Status
trinity	Enabled	Mounted

Applicable Tasks:

Figure 11-1 Window Showing the Metadata Server

Metadata Server Discovery with `clconf_info`

You can use the `clconf_info` command to discover the active metadata server for a given filesystem. For example, the following shows that `cxfs7` is the metadata server:

```
cxfs6 # clconf_info

Event at [2004-04-16 09:20:59]
```

```
Membership since Fri Apr 16 09:20:56 2004

____________ ______ ________ ______ ______
Node         NodeID Status   Age    CellID
____________ ______ ________ ______ ______
cxfs6             6 up       0           2
cxfs7             7 up       0           1
cxfs8             8 up       0           0
____________ ______ ________ ______ ______
1 CXFS FileSystems
/dev/cxvm/concat0 on /concat0  enabled  server=(cxfs7)  2 client(s)=(cxfs8,cxfs6)
```

Metadata Server Recovery

Note: Recovery is supported only when using standby nodes.

If the node acting as the metadata server for a filesystem dies, another node in the list of potential metadata servers will be chosen as the new metadata server. This assumes that at least two potential metadata servers are listed when you define a filesystem. For more information, see "Define CXFS Filesystems with the GUI" on page 204, or "Modify a Cluster with `cmgr`" on page 250.

The metadata server that is chosen must be a filesystem client; other filesystem clients will experience a delay during the relocation process. Each filesystem will take time to recover, depending upon the number of active inodes; the total delay is the sum of time required to recover each filesystem. Depending on how active the filesystems are at the time of recovery, the total delay could take up to several minutes per filesystem.

If a CXFS client dies, the metadata server will clean up after the client. Other CXFS clients may experience a delay during this process. A delay depends on what tokens, if any, that the deceased client holds. If the client has no tokens, then there will be no delay; if the client is holding a token that must be revoked in order to allow another client to proceed, then the other client will be held up until recovery returns the failed nodes tokens (for example, in the case where the client has the write token and another client wants to read). The actual length of the delay depends upon the following:

- The total number of exported inodes on the metadata server

- CXFS kernel membership situation
- Whether any servers have died
- Where the servers are in the recovery order relative to recovering this filesystem

The deceased CXFS client is not allowed to rejoin the CXFS kernel membership until all metadata servers have finished cleaning up after the client.

Shutdown of the Database and CXFS

This section tells you how to perform the following:

- "Cluster Database Shutdown" on page 300
- "Normal CXFS Shutdown: Stop CXFS Services" on page 302
- "Forced CXFS Shutdown: Revoke Membership of Local Node" on page 304

If there are problems, see Chapter 17, "Troubleshooting" on page 381. For more information about states, Chapter 15, "Monitoring Status" on page 359.

Cluster Database Shutdown

A *cluster database shutdown* terminates the following user-space daemons that manage the cluster database:

- `cad`
- `clconfd`
- `cmond`
- `crsd`
- `fs2d`

After shutting down the database on a node, access to the shared filesystems remains available and the node is still a member of the cluster, but the node is not available for database updates. Rebooting of the node results in a restart of all services (restarting the daemons, joining cluster membership, enabling cluster volumes, and mounting CXFS filesystems_.

To perform a cluster database shutdown, enter the following:

- IRIX:

  ```
  irix# /etc/init.d/cluster stop
  ```

- Linux:

  ```
  [root@linux root]# /etc/init.d/cxfs_cluster stop
  ```

If you also want to disable the daemons from restarting at boot time, enter the following:

- IRIX:

  ```
  irix# chkconfig cluster off
  ```

- Linux:

  ```
  [root@linux root]# chkconfig cxfs_cluster off
  ```

For more information, see "CXFS `chkconfig` Arguments" on page 289.

Node Status and Cluster Database Shutdown

A cluster database shutdown is appropriate when you want to perform a maintenance operation on the node and then reboot it, returning it to `ACTIVE` status.

If you perform a cluster database shutdown, the node status will be `DOWN`, which has the following impacts:

- The `DOWN` node is still considered part of the cluster, but unavailable.
- The `DOWN` node does not get cluster database updates; however, it will be notified of all updates after it is rebooted.

  Missing cluster database updates can cause problems if the kernel portion of CXFS is active. That is, if the node continues to have access to CXFS, the node's kernel level will not see the updates and will not respond to attempts by the remaining nodes to propagate these updates at the kernel level. This in turn will prevent the cluster from acting upon the configuration updates.

Restart the Cluster Database

To restart the cluster database, enter the following:

- IRIX:

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

- Linux:

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

Normal CXFS Shutdown: Stop CXFS Services

You should perform a *normal CXFS shutdown* when you want to stop CXFS services on a node and remove it from the CXFS kernel membership quorum. A normal CXFS shutdown does the following:

- Unmounts all the filesystems except those for which it is the active metadata server; those filesystems for which the node is the active metadata server will become inaccessible from the node after it is shut down.
- Terminates the CXFS kernel membership of this node in the cluster.
- Marks the node as `INACTIVE`.

The effect of this is that cluster disks are unavailable and no cluster database updates will be propagated to this node. Rebooting the node leaves it in the shutdown state.

If the node on which you shut down CXFS services is an active metadata server for a filesystem, then that filesystem will be recovered by another node that is listed as one of its potential metadata servers. For more information, see "Define CXFS Filesystems with the GUI" on page 204, or "Modify a Cluster with `cmgr`" on page 250. The server that is chosen must be a filesystem client; other filesystem clients will experience a delay during the recovery process.

If the node on which the CXFS shutdown is performed is the sole potential metadata server (that is, there are no other nodes listed as potential metadata servers for the filesystem), then you should use the CXFS GUI or the `cmgr` command to unmount the filesystem from all nodes before performing the shutdown.

To perform a normal CXFS shutdown, enter the following `cmgr` command:

```
cmgr> stop cx_services on node nodename for cluster clustername
```

You could also use the GUI; see "Stop CXFS Services with the GUI" on page 191.

Note: This action deactivates CXFS services on **one** node, forming a new CXFS kernel membership after deactivating the node. If you want to stop CXFS services on multiple nodes, you must enter this command multiple times or perform the task using the GUI.

After you stop CXFS services on a node, the node is marked as inactive and is no longer used when calculating the CXFS kernel membership. See "Node Status" on page 365.

Node Status and Stopping CXFS Services

After performing stopping CXFS services on a node, its state will be `INACTIVE`; therefore, it will not impact CXFS kernel membership quorum calculation. See "Normal CXFS Shutdown: Stop CXFS Services" on page 302.

When You Should Not Perform Stop CXFS Services

You should not stop CXFS services under the following circumstances:

- On the *local node*, which is the CXFS administration node on which the cluster manager is running or the node to which the GUI is connected
- If stopping CXFS services on the node will result in loss of CXFS kernel membership quorum
- If the node is the only available metadata server for one or more active CXFS filesystems

If you want to perform a CXFS shutdown under these conditions, you must perform a forced CXFS shutdown. See "Forced CXFS Shutdown: Revoke Membership of Local Node" on page 304.

Rejoining the Cluster after a Stopping CXFS Services

The node will not rejoin the cluster after a reboot. The node will rejoin the cluster only when CXFS services are explicitly reactivated with the GUI (see "Start CXFS Services with the GUI" on page 191) or the following command:

```
cmgr> start cx_services on node nodename for cluster clustername
```

Forced CXFS Shutdown: Revoke Membership of Local Node

A *forced CXFS shutdown* (or *administrative CXFS stop*) is appropriate when you want to shutdown the local node even though it may drop the cluster below its CXFS kernel membership quorum requirement.

CXFS does the following:

- Shuts down all CXFS filesystems on the local node
- Attempts to access the CXFS filesystems result in I/O error (you may need to manually unmount the filesystems)
- Removes this node from the CXFS kernel membership
- Marks the node as `DOWN`
- Disables access from the local node to cluster-owned XVM volumes.
- Treats the stopped node as a failed node and executes the failure action defined for the node in the cluster database (`reset`, `fence`, `fencereset`, or `shutdown`).

Caution: A forced CXFS shutdown may cause the cluster to fail if the cluster drops below CXFS kernel membership quorum.

If you do a forced CXFS shutdown on an active metadata server, it loses membership immediately. At this point another potential metadata server must take over (and recover the filesystems) or quorum is lost and a forced shutdown follows on all nodes.

If you do a forced CXFS shutdown that forces a loss of quorum, the remaining part of the cluster (which now must also do an administrative stop) will **not** reset the departing node.

To perform an administrative stop, enter the following `cmgr` command to revoke the CXFS kernel membership of the local node:

```
cmgr> admin cxfs_stop
```

You can also perform this action with the GUI; see "Revoke Membership of the Local Node with the GUI" on page 195. This action can also be triggered automatically by the kernel after a loss of CXFS kernel membership quorum.

Node Status and Forced CXFS Shutdown

After a forced CXFS shutdown, the node is still considered part of the configured cluster and is taken into account when propagating the cluster database (these services are still running) and when computing the cluster database (`fs2d`) membership quorum (this could cause a loss of quorum for the rest of the cluster, causing the other nodes to do a forced CXFS shutdown). The state is `INACTIVE`.

It is important that this node stays accessible and keeps running the cluster infrastructure daemons to ensure database consistency. In particular, if more than half the nodes in the pool are down or not running the infrastructure daemons, cluster database updates will stop being propagated and will result in inconsistencies. To be safe, you should remove those nodes that will remain unavailable from the cluster and pool. See:

- "Add or Remove Nodes in the Cluster with the GUI" on page 181, or "Modify a Cluster with `cmgr`" on page 250
- "Delete a Node with the GUI" on page 186, or "Delete a Node with `cmgr`" on page 241

Rejoining the Cluster after a Forced CXFS Shutdown

After a forced CXFS shutdown, the local node will not resume CXFS kernel membership until the node is rebooted or until you explicitly allow CXFS kernel membership for the local node by entering the following `cmgr` command:

```
cmgr> admin cxfs_start
```

You can also perform this step with the GUI; see "Allow Membership of the Local Node with the GUI" on page 195.

If you perform a forced CXFS shutdown on a CXFS administration node, you must restart CXFS on that node before it can return to the cluster. If you do this while the cluster database still shows that the node is in a cluster and is activated, the node will restart the CXFS kernel membership daemon. Therefore, you may want to do this after resetting the database or after stopping CXFS services.

Reset Capability and a Forced CXFS Shutdown

Caution: If you perform an administrative CXFS stop on an administration node with system reset capability and the stop will not cause loss of cluster quorum, the node will be reset (rebooted) by the appropriate node.

For more information about resets, see "System Reset" on page 34.

Avoiding a CXFS Restart at Reboot

If the following `chkconfig` arguments are turned off, the `clconfd` and `cxfs_client` daemons on CXFS administration nodes and client-only nodes, respectively, will not be started at the next reboot and the kernel will not be configured to join the cluster:

- IRIX administration nodes: `cxfs_cluster`
- Linux administration nodes: `cxfs`
- Client-only nodes: `cxfs_client`

It is useful to turn these arguments off before rebooting if you want to temporarily remove the nodes from the cluster for system or hardware upgrades or for other maintenance work.

For example, do the following:

- IRIX administration node:

```
irix# /etc/chkconfig cxfs_cluster off
irix# /etc/chkconfig cluster off
irix# reboot
```

- Linux administration node:

```
[root@linux root]# /sbin/chkconfig cxfs off
[root@linux root]# /sbin/chkconfig cxfs_cluster off
[root@linux root]# reboot
```

For more information, see "CXFS `chkconfig` Arguments" on page 289.

Log File Management

You should rotate the log files at least weekly so that your disk will not become full. The following sections provide example scripts. For information about log levels, see "Configure Log Groups with the GUI" on page 194.

Rotating All Log Files

You can run the `/var/cluster/cmgr-scripts/rotatelogs` script to copy all files to a new location. This script saves log files with the day and the month name as a suffix. If you run the script twice in one day, it will append the current log file to the previous saved copy. The `root crontab` file has an entry to run this script weekly.

The script syntax is as follows:

```
/var/cluster/cmgr-scripts/rotatelogs [-h] [-d|-u]
```

If no option is specified, the log files will be rotated. Options are as follows:

`-h`	Prints the help message. The log files are not rotated and other options are ignored.
`-d`	Deletes saved log files that are older than one week before rotating the current log files. You cannot specify this option and `-u`.
`-u`	Unconditionally deletes all saved log files before rotating the current log files. You cannot specify this option and `-d`.

By default, the `rotatelogs` script will be run by `crontab` once a week, which is sufficient if you use the default log levels. If you plant to run with a high debug level for several weeks, you should reset the `crontab` entry so that the `rotatelogs` script is run more often.

On heavily loaded machines, or for very large log files, you may want to move resource groups and stop CXFS services before running `rotatelogs`.

Rotating Large Log Files

You can use a script such as the following to copy large files to a new location. The files in the new location will be overwritten each time this script is run.

```
#!/bin/sh
# Argument is maximum size of a log file (in characters) - default: 500000

size=${1:-500000}
find /var/cluster/ha/log -type f ! -name ’*.OLD’ -size +${size}c -print | while read log_file; do
        cp ${log_file} ${log_file}.OLD
        echo ’*** LOG FILE ROTATION ’ ‘date‘ ’***’ > ${log_file}
done
```

Also see "`cad.options` on CXFS Administration Nodes" on page 96, and "`fs2d.options` on CXFS Administration Nodes" on page 98

Volume Management

CXFS uses the XVM volume manager. XVM can combine many disks into high transaction rate, high bandwidth, and highly reliable filesystems. CXFS uses XVM to provide the following:

- Disk striping
- Mirroring
- Concatenation
- Advanced recovery features

Note: The `xvm` command must be run on a CXFS administration node. If you try to run an XVM command before starting the CXFS daemons, you will get a warning message and be put into XVM’s *local domain*.

When you are in XVM’s local domain, you could define your filesystems, but then when you later start up CXFS you will not see the filesystems. When you start up CXFS, XVM will switch to *cluster domain* and the filesystems will not be recognized because you defined them in local domain; to use them in the cluster domain, you would have to use the `give` command. Therefore, it is better to define the volumes directly in the cluster domain.

For more information, see the *XVM Volume Manager Administrator's Guide*.

Disk Management

This section describes the CXFS differences for backups, NFS, Quotas, and Samba.

Disk Backups

CXFS enables the use of commercial backup packages such as VERITAS NetBackup and Legato NetWorker for backups that are free from the local area network (LAN), which allows the backup server to consolidate the backup work onto a backup server while the data passes through a storage area network (SAN), rather than through a lower-speed LAN.

For example, a backup package can run on a host on the SAN designated as a backup server. This server can use attached tape drives and channel connections to the SAN disks. It runs the backup application, which views the filesystems through CXFS and transfers the data directly from the disks, through the backup server, to the tape drives.

This allows the backup bandwidth to scale to match the storage size, even for very large filesystems. You can increase the number of disk channels, the size of the backup server, and the number of tape channels to meet the backup-bandwidth requirements.

Note: Do not run backups on a client node because it causes heavy use of non-swappable kernel memory on the metadata server. During a backup, every inode on the filesystem is visited, and if done from a client, it imposes a huge load on the metadata server. The metadata server may experience typical out-of-memory symptoms, and in the worst case can even become unresponsive or crash.

NFS

You can put an NFS server on top of CXFS so that computer systems that are not part of the cluster can share the filesystems. You should run the NFS server on the CXFS active metadata server for optimal performance.

Quotas

XFS quotas are supported. However, the quota mount options must be the same on all mounts of the filesystem. You can administer quotas from any IRIX or Linux node in the cluster that has the quota administration software installed. You must install the quota administration software on the potential server administration nodes in the cluster.

Samba

You can run Samba on top of CXFS, allowing Windows machines to support CXFS and have access to the filesystem. Samba should run on the active metadata server for optimal performance. You should not serve the same CXFS filesystem from multiple nodes in a cluster.

The architecture of Samba assumes that each share is exported by a single server. Because all Samba client accesses to files and directories in that share are directed through a single Samba server, the Samba server is able to maintain private metadata state to implement the required concurrent access controls (in particular, share modes, write caching and oplock states). This metadata is not necessarily promulgated to the filesystem and there is no protocol for multiple Samba servers exporting the same share to communicate this information between them.

Running multiple Samba servers on one or more CXFS (or NFS) clients exporting a single share that maps to a common underlying filesystem has the following risks:

- File data corruption from writer-writer concurrency
- Application failure due to inconsistent file data from writer-reader concurrency

These problems do not occur when a single Samba server is deployed, because that server maintains a consistent view of the metadata used to control concurrent access across all Samba clients.

It may be possible to deploy multiple Samba servers under one of the following circumstances:

- There are no writers, so a read-only share is exported
- Application-level protocols and/or work-flow guarantee that only one application is ever writing a file, and concurrent file writing and reading does not take place

Caution: The onus is on the customer to ensure these conditions are met, as there is nothing in the Samba architecture to verify it. Therefore, SGI recommends that you do not use multiple Samba servers.

Filesystem Maintenance

Although filesystem information is traditionally stored in `/etc/fstab`, the CXFS filesystems information is relevant to the entire cluster and is therefore stored in the replicated cluster database instead.

As the administrator, you will supply the CXFS filesystem configuration by using the CXFS Cluster Manager tools. For information about the GUI, see "Filesystem Tasks with the GUI" on page 200; for information about `cmgr`, see "Cluster Tasks with `cmgr`" on page 246.

The information is then automatically propagated consistently throughout the entire cluster. The cluster configuration daemon mounts the filesystems on each node according to this information, as soon as it becomes available.

A CXFS filesystem will be automatically mounted on all the nodes in the cluster. You can add a new CXFS filesystem to the configuration when the cluster is active.

Whenever the cluster configuration daemon detects a change in the cluster configuration, it does the equivalent of a `mount -a` command on all the filesystems that are configured.

Caution: You must not modify or remove a CXFS filesystem definition while the filesystem is mounted. You must unmount it first and then mount it again after the modifications.

Mounting Filesystems

You supply mounting information with the GUI **Mount a Filesystem** task (which is part of the **Set Up a New Filesystem** guided configuration task) or with the `modify` subcommand to `cmgr`(1M). See the following:

- For information about mounting using the GUI, see "Set Up a New CXFS Filesystem with the GUI" on page 132, and "Define CXFS Filesystems with the GUI" on page 204.
- For information about defining and mounting a new filesystem with `cmgr`, see "Modify a Cluster with `cmgr`" on page 250.
- For information about mounting a filesystem that has already been defined but is currently unmounted, see "Define a CXFS Filesystem with `cmgr`" on page 260.

When properly defined and mounted, the CXFS filesystems are automatically mounted on each node by the local cluster configuration daemon, `clconfd`, according to the information collected in the replicated database. After the filesystems configuration has been entered in the database, no user intervention is necessary.

Caution: Do not attempt to use the `mount` command to mount a CXFS filesystem. Doing so can result in data loss and/or corruption due to inconsistent use of the filesystem from different nodes.

Mount points cannot be nested when using CXFS. That is, you cannot have a filesystem within a filesystem, such as `/usr` and `/usr/home`.

Unmounting Filesystems

To unmount CXFS filesystems, use the GUI **Unmount a Filesystem** task or the `admin` subcommand to `cmgr`. For information, see "Unmount CXFS Filesystems with the GUI" on page 208, or "Unmount a CXFS Filesystem with `cmgr`" on page 267.

These tasks unmount a filesystem from all nodes in the cluster. Although this action triggers an unmount on all the nodes, some might fail if the filesystem is busy. On active metadata servers, the unmount cannot succeed before all of the CXFS clients have successfully unmounted the filesystem. All nodes will retry the unmount until it succeeds, but there is no centralized report that the filesystem has been unmounted on all nodes.

To verify that the filesystem has been unmounted from all nodes, do one of the following:

- Check the `SYSLOG` files on the metadata servers for a message indicating that the filesystem has been unmounted.
- Run the GUI or `cmgr` on the metadata server, disable the filesystem from the server, and wait until the GUI shows that the filesystem has been fully disabled. (It will be an error if it is still mounted on some CXFS clients and the GUI will show which clients are left.)

Growing Filesystems

To grow a CXFS filesystem, do the following:

1. Unmount the CXFS filesystem. For information, see "Unmount CXFS Filesystems with the GUI" on page 208, or "Unmount a CXFS Filesystem with `cmgr`" on page 267.
2. Change the domain of the XVM volume from a cluster volume to a local volume using the XVM `give` command. See the *XVM Volume Manager Administrator's Guide*.
3. Mount the filesystem as an XFS filesystem. See *IRIX Admin: Disks and Filesystems*.
4. Use the `xfs_growfs` command or the GUI task; see "Grow a Filesystem with the GUI" on page 202.
5. Unmount the XFS filesystem. See *IRIX Admin: Disks and Filesystems*.
6. Change the domain of the XVM volume back to a cluster volume using the `give` command. See the *XVM Volume Manager Administrator's Guide*.
7. Mount the filesystem as a CXFS filesystem. See "Mount CXFS Filesystems with the GUI" on page 208, or "Mount a CXFS Filesystem with `cmgr`" on page 266.

Dump and Restore

You must perform the backup of a CXFS filesystem from the metadata server of that filesystem. The `xfsdump` and `xfsrestore` commands make use of special system calls that will only function on the metadata server.

If there are multiple potential metadata servers for a filesystem and the primary server goes down because of this problem, the backup metadata server will gather information on all open files in the cluster. Unless the backup server has much larger memory than the primary server, the result is that it too will go down with exactly the same symptoms that caused the primary server to crash.

You must perform dump and restore procedures from the active metadata server.

The filesystem can have active clients during a dump process.

In a clustered environment, a CXFS filesystem may be directly accessed simultaneously by many CXFS clients and the active metadata server. With failover or simply metadata server reassignment, a filesystem may, over time, have a number of metadata servers. Therefore, in order for `xfsdump` to maintain a consistent inventory, it must access the inventory for past dumps, even if this information is located on another node.

SGI recommends that the inventory be made accessible by potential metadata server nodes in the cluster using one of the following methods:

- Relocate the inventory to a shared filesystem.

  For example, where *shared_filesystem* is replaced with the actual name of the filesystem to be shared:

  – On the node currently containing the inventory, enter the following:

  ```
  # cd /var
  # cp -r xfsdump /shared_filesystem
  # mv xfsdump xfsdump.bak
  # ln -s /shared_filesystem/xfsdump xfsdump
  ```

  – On all other administration nodes in the cluster, enter the following:

  ```
  # cd /var
  # mv xfsdump xfsdump.bak
  # ln -s /shared_filesystem/xfsdump xfsdump
  ```

- Export the directory using an NFS shared filesystem.

  For example:

  – On the IRIX node currently containing the inventory, add `/var/xfsdump` to `/etc/exports` and then enter the following:

  ```
  irix# exportfs -a
  ```

(On a Linux, the path is `/var/lib/xfsdump`.)

– On all other IRIX administration nodes in the cluster, enter the following:

```
# cd /var
# mv  xfsdump  xfsdump.bak
# ln -s /hosts/hostname/var/xfsdump  xfsdump
```

Note: It is the IRIX `/var/xfsdump` directory (Linux `/var/lib/xfsdump`) that should be shared, rather than the IRIX `/var/xfsdump/inventory` directory (Linux `/var/lib/xfsdump/inventory`). If there are inventories stored on various nodes, you can use `xfsinvutil` to merge them into a single common inventory, prior to sharing the inventory among the cluster.

System Tunable Parameters

Table 11-3 shows the system tunable parameters available with CXFS. You can use the `sysctl` command to manipulate these parameters. On IRIX you can also use the `systune` command. On Linux, you can also specify them in the `/etc/modules.conf` file using the following format:

`options` *modulename parameter1=value1,parameter2=value2*

For more information, see the `sysctl`(1M), `systune`(1M), and `modules.conf`(5) man page.

Table 11-3 System Tunable Parameters

Parameter	Description	Location
`cms_fence_timeout`	Specifies the number of seconds to wait for `clconfd` to acknowledge a fence request. 0 is an infinite wait and is the default. If a non-zero value is set and the time-out expires, CXFS takes the action specified by the `cms_fence_timeout_action` parameter. This parameter may be changed at run time. Before setting the time-out, you should understand the ramifications of doing so on your system. Modification of this parameter is not generally recommended.	IRIX: `/var/sysgen/mtune/cell` Linux: `kernel.cell` (`sgi-cell` module)
`cms_fence_timeout_action`	Specifies the action to be taken when `clconfd` does not acknowledge a reset request (determined by `cms_fence_timeout`). `cms_fence_timeout_action` may be changed at run time, and may be set to one of the following. Before setting the time-out, you should understand the ramifications of doing so on your system. Modification of this parameter is not generally recommended.	IRIX: `/var/sysgen/mtune/cell` Linux: `kernel.cell` (`sgi-cell` module)

Parameter	Description	Location
	• 0 - Causes the node waiting for the fence acknowledgement to forcibly withdraw from the cluster, equivalent to a forced CXFS shutdown that occurs when a node loses quorum (default). If `clconfd` is still present and functioning properly, it will then restart the kernel `cms` daemon and the node will attempt to rejoin the cluster. • 1 - Clears all pending fence requests and continues (that is, fakes acknowledgment). **CAUTION:** Setting this value is potentially dangerous. • 2 - Panics the local node	
`cms_reset_timeout`	Specifies the number of seconds to wait for `clconfd` to acknowledge a reset request. 0 is an infinite wait and is the default. If a non-zero value is set and the time-out expires, CXFS takes the action specified by the `cms_reset_timeout_action` parameter. This parameter may be changed at run time.	IRIX: `/var/sysgen/mtune/cell` Linux: `kernel.cell` (`sgi-cell` module)
`cms_reset_timeout_action`	Specifies the action to be taken when `clconfd` does not acknowledge a reset request (determined by `cms_reset_timeout`). `cms_reset_timeout_action` may be changed at run time, and may be set to one of the following:	IRIX: `/var/sysgen/mtune/cell` Linux: `kernel.cell` (`sgi-cell` module)

Parameter	Description	Location
	• 0 - Causes the node waiting for the reset acknowledgement to forcibly withdraw from the cluster, equivalent to a forced CXFS shutdown that occurs when a node loses quorum (default). If `clconfd` is still present and functioning properly, it will then restart the kernel `cms` daemon and the node will attempt to rejoin the cluster. • 1 - Clears all pending resets and continues (that is, fakes acknowledgment). **CAUTION:** Setting this value is potentially dangerous. • 2 - Panics the local node	
`cxfsd_min`	Specifies the minimum number of `cxfsd` threads to run per CXFS filesystem.	IRIX: `/var/sysgen/mtune/cxfs` Linux: `fs.cxfs` (`sgi-cxfs` module)

Parameter	Description	Location
	The `cxfsd` threads do the disk block allocation for delayed allocation buffers in CXFS and the flushing of buffered data for files that are being removed from the local cache by the metadata server. The threads are allocated at filesystem mount time. The value of the `cxfsd_min` parameter at mount time remains in effect for a filesystem until it is unmounted. The legal value for `cxfsd_min` is an integer in the range 1 through 256. The default is 2 times the number of CPUS, and the number of actual running `cxfsd` threads is dynamic. Customers should not need to modify the default settings.	
`cxfsd_max`	Specifies the maximum number of `cxfsd` threads to run per CXFS filesystem. The value of the `cxfsd_max` parameter at mount time remains in effect for a filesystem until it is unmounted. The legal value for `cxfsd_max` is an integer in the range 8 through 4096. The default is 16. The value for `cxfsd_max` cannot be less than the value specified for `cxfsd_min`.	IRIX: `/var/sysgen/mtune/cxfs` Linux: `fs.cxfs` (`sgi-cxfs` module)
`cxfs_prefetch`	Enables (`1`) or disables (`0`) token obtain optimization. Enabled by default.	IRIX: `/var/sysgen/mtune/cxfs` Linux: `fs.cxfs` (`sgi-cxfs` module)

Parameter	Description	Location
`cxfs_relocation_ok`	Specifies whether relocation is disabled or enabled (must be specified on the active metadata server): • 0 - Disables relocation • 1 - Enables relocation **Note:** Relocation is disabled by default and is only supported on standby nodes.	IRIX: `/var/sysgen/mtune/cxfs` Linux: `fs.cxfs` (`sgi-cxfs` module)
`cxfs_shutdown_time`	Specifies the time other nodes will wait for the node to take media offline after they have recognized that it has lost quorum, if the node has neither fencing nor reset configured. SGI recommends a value of 50 (0.5 seconds).	IRIX: `/var/sysgen/mtune/cell` Linux: `kernel.cell` (`sgi-cell` module)
`mtcp_nodelay`	Enables `TCP_NODELAY` on CXFS message channels. SGI recommends that you do not change this value.	IRIX: `/var/sysgen/mtune/cell` Linux: `kernel.cell` (`sgi-cell` module)
`mtcp_hb_period`	Specifies the length of time, in Hz, that CXFS waits for heartbeat from other nodes before declaring node failure. SGI recommends a value of 500 (5 seconds). You should only change this value at the recommendation of SGI support. The same value must be used on all nodes in the cluster; if you change this value, you must create new kernels and reboot them on each node.	IRIX: `/var/sysgen/mtune/cell` Linux: `kernel.cell` (`sgi-cell` module)

Parameter	Description	Location
`mtcp_reserve_size`	Sets the size of the TCP window. SGI recommends that you do not change this value.	IRIX: `/var/sysgen/mtune/cell` Linux: `kernel.cell` (`sgi-cell` module)
`mtcp_mesg_validate`	Enables checksumming on top of what TCP is already doing. Normally, this is not needed and is only used if TCP data corruption is suspected. The legal values are as follows: • 0 - Performs no validation • 1 - Generates checksums, but does not perform validation • 2 - Generates and validates checksums, warns (via a `SYSLOG` message) on validation failure • 3 - Generates and validates checksums, warns and returns an error message on validation failure • 4 - Generates and validates checksums, warns and panics on validation error	IRIX: `/var/sysgen/mtune/cell` Linux: `kernel.cell` (`sgi-cell` module)

Hardware Changes and I/O Fencing

If you use I/O fencing and then make changes to your hardware configuration, you must verify that switch ports are properly enabled so that they can discover the WWPN of the HBA for I/O fencing purposes.

You must check the status of the switch ports involved whenever any of the following occur:

- An HBA is replaced on a node
- A new node is plugged into the switch for the first time

- A Fibre Channel cable rearrangement occurs

Note: The affected nodes should be shutdown before rearranging cables.

To check the status, use the following command on a CXFS administration node:

```
hafence -v
```

If any of the affected ports are found to be disabled, you must manually enable them before starting CXFS on the affected nodes:

1. Connect to the switch using `telnet`.
2. Use the `portenable` command to enable the port.
3. Close the `telnet` session.

After the port is enabled, the metadata server will be able to discover the new (or changed) WWPN of the HBA connected to that port and thus correctly update the switch configuration entries in the cluster database.

Configuring Private Network Failover

This section provides an example of modifying a cluster to provide private network failover. For more information, see "Define a Cluster with `cmgr`" on page 246 and "Modify a Cluster with `cmgr`" on page 250.

Suppose your cluster has the following configuration:

```
irix# cxfs-config
Global:
    cluster: mycluster (id 1)
    cluster state: enabled
    tiebreaker: yellow

Networks:

Machines:
...
    node red: node 55    cell 4  enabled  IRIX    server_admin
        hostname: red.mycompany.com
        fail policy: Fence, Shutdown
        nic 0: address: 192.168.0.1 priority: 1

    node yellow: node 2     cell 3  enabled  IRIX    server_admin
        hostname: yellow.mycompany.com
        fail policy: Fence, Shutdown
        nic 0: address: 192.168.0.2 priority: 1
```

To change the configuration to support private network failover, you would do the following:

1. Ensure that CXFS services are not active.

   Note: You cannot add a NIC or a network grouping while CXFS services are active (that is, when `start cx_services` has been executed); doing so can lead to cluster malfunction.

   If services have been started, stopped them as follows:

```
[root@linux root]# cmgr -p
Welcome to SGI Cluster Manager Command-Line Interface

cmgr> stop cx_services for cluster mycluster

CXFS services have been deactivated in cluster mycluster
```

2. Add another set of NICs to support a second CXFS network. (The second network will be used as the failover network and can be the public network and thus does not have to be a second CXFS private network.) For example:

```
cmgr> modify node red
Enter commands, you may enter "done" or "cancel" at any time to exit

Hostname[optional] ? (red.mycompany.com)
Is this a FailSafe node <true|false> ? (false)
Is this a CXFS node <true|false> ? (true)
Partition ID[optional] ? (0)
Do you wish to modify failure hierarchy[y/n]:n
Reset type <powerCycle|reset|nmi> ? (powerCycle)
Do you wish to modify system controller info[y/n]:n
Number of Network Interfaces ? (1) 2
NIC 1 - IP Address ? (192.168.0.1)
NIC 1 - Heartbeat HB (use network for heartbeats) <true|false> ? (true)
NIC 1 - (use network for control messages) <true|false> ? (true)
NIC 1 - Priority <1,2,...> ? (1)
NIC 2 - IP Address ? 192.168.1.1
NIC 2 - Heartbeat HB (use network for heartbeats) <true|false> ? true
NIC 2 - (use network for control messages) <true|false> ? true
NIC 2 - Priority <1,2,...> ? 2

Successfully modified node red

cmgr> modify node yellow
Enter commands, you may enter "done" or "cancel" at any time to exit

Hostname[optional] ? (yellow.mycompany.com)
Is this a FailSafe node <true|false> ? (false)
Is this a CXFS node <true|false> ? (true)
Partition ID[optional] ? (0)
Do you wish to modify failure hierarchy[y/n]:n
Reset type <powerCycle|reset|nmi> ? (powerCycle)
Do you wish to modify system controller info[y/n]:n
Number of Network Interfaces ? (1) 2
NIC 1 - IP Address ? (192.168.0.2)
NIC 1 - Heartbeat HB (use network for heartbeats) <true|false> ? (true)
NIC 1 - (use network for control messages) <true|false> ? (true)
NIC 1 - Priority <1,2,...> ? (1)
NIC 2 - IP Address ? 192.168.1.2
NIC 2 - Heartbeat HB (use network for heartbeats) <true|false> ? true
NIC 2 - (use network for control messages) <true|false> ? true
NIC 2 - Priority <1,2,...> ? 2

Successfully modified node yellow
```

Repeat this process for each node. You can use the `cxfs-config` command to display the defined NICs. For example:

```
irix# cxfs-config
...
        node red: node 55    cell 4  enabled  IRIX    server_admin
        hostname: red.mycompany.com
        fail policy: Fence, Shutdown
        nic 0: address: 192.168.0.1 priority: 1
        nic 1: address: 192.168.1.1 priority: 2
```

```
    node yellow: node 2     cell 3  enabled  IRIX    server_admin
        hostname: yellow.mycompany.com
        fail policy: Fence, Shutdown
        nic 0: address: 192.168.0.2 priority: 1
        nic 1: address: 192.168.1.2 priority: 2
```

3. Configure the NICs into networks. (CXFS will ignore NICs other than priority 1 unless you configure the NICs into networks.)

   a. Configure the primary network:

```
cmgr> modify cluster mycluster
Enter commands, you may enter "done" or "cancel" at any time to exit

Is this a FailSafe cluster <true|false> ? (false)
Is this a CXFS cluster <true|false> ? (true)
Cluster Notify Cmd [optional] ?
Cluster Notify Address [optional] ?
Cluster CXFS mode <normal|experimental>[optional] ? (normal)
Cluster ID ? (1)

Current nodes in cluster mycluster:
Node - 1: green
Node - 2: orange
Node - 3: red
Node - 4: purple
Node - 5: yellow
Node - 6: blue


No networks in cluster mycluster

Add nodes to or remove nodes/networks from cluster mycluster
Enter "done" when completed or "cancel" to abort

mycluster ? add net network 192.168.0.0 mask 255.255.255.0
mycluster ? done
Successfully modified cluster mycluster
```

At this point, `cxfs-config` will show the primary network (network `0`):

```
irix# cxfs-config
...
Networks:
    net 0: type tcpip  192.168.0.0      255.255.255.0

Machines:
...
    node red: node 55    cell 4  enabled  IRIX    server_admin
        hostname: red.mycompany.com
        fail policy: Fence, Shutdown
        nic 0: address: 192.168.0.1 priority: 1 network: 0
        nic 1: address: 192.168.1.1 priority: 2 network: none

    node yellow: node 2     cell 3  enabled  IRIX    server_admin
        hostname: yellow.mycompany.com
        fail policy: Fence, Shutdown
        nic 0: address: 192.168.0.2 priority: 1 network: 0
        nic 1: address: 192.168.1.2 priority: 2 network: none
...
```

b. Configure the secondary network:

```
cmgr> modify cluster mycluster
Enter commands, you may enter "done" or "cancel" at any time to exit

Is this a FailSafe cluster <true|false> ? (false)
Is this a CXFS cluster <true|false> ? (true)
Cluster Notify Cmd [optional] ?
Cluster Notify Address [optional] ?
Cluster CXFS mode <normal|experimental>[optional] ? (normal)
Cluster ID ? (1)

Current nodes in cluster mycluster:
Node - 1: green
Node - 2: orange
Node - 3: red
Node - 4: purple
Node - 5: yellow
Node - 6: blue
```

```
No networks in cluster mycluster

Add nodes to or remove nodes/networks from cluster mycluster
Enter "done" when completed or "cancel" to abort

mycluster ? **add net network 192.168.1.0 mask 255.255.255.0**
mycluster ? **done**
Successfully modified cluster mycluster
```

The `cxfs-config` command will now display the secondary network (network 1):

```
irix# cxfs-config
...
Networks:
    net 0: type tcpip  192.168.0.0      255.255.255.0
    net 1: type tcpip  192.168.1.0      255.255.255.0

Machines:
...
    node red: node 55    cell 4  enabled  IRIX    server_admin
        hostname: red.mycompany.com
        fail policy: Fence, Shutdown
        nic 0: address: 192.168.0.1 priority: 1 network: 0
        nic 1: address: 192.168.1.1 priority: 2 network: 1

    node yellow: node 2     cell 3  enabled  IRIX    server_admin
        hostname: yellow.mycompany.com
        fail policy: Fence, Shutdown
        nic 0: address: 192.168.0.2 priority: 1 network: 0
        nic 1: address: 192.168.1.2 priority: 2 network: 1
```

During this process, the console will display the membership transitions as the nodes upload the changed network configuration to their kernels. When the second network is added, the following messages will be displayed on the console for `yellow`:

```
NOTICE: Starting tcp server for interface 192.168.1.2 channel 0
NOTICE: Starting tcp server for interface 192.168.1.2 channel 1
NOTICE: Membership delivered. Membership contains 3(5) 4(3)  cells
```

```
NOTICE: Discovered cell 4 (red) [tcp priority 2 at 192.168.1.1 via 192.168.1.2]
```

To delete a network, do the following:

```
cmgr> modify cluster mycluster
Enter commands, you may enter "done" or "cancel" at any time to exit

Is this a FailSafe cluster <true|false> ? (false)
Is this a CXFS cluster <true|false> ? (true)
Cluster Notify Cmd [optional] ?
Cluster Notify Address [optional] ?
Cluster CXFS mode <normal|experimental>[optional] ? (normal)
Cluster ID ? (1)

Current nodes in cluster mycluster:
Node - 1: green
Node - 2: orange
Node - 3: red
Node - 4: purple
Node - 5: yellow
Node - 6: blue


Current networks in cluster mycluster:
Network 0 - network 192.168.0.0, mask 255.255.255.0
Network 1 - network 192.168.1.0, mask 255.255.255.0

cmgr> modify cluster mycluster
Enter commands, when finished enter either "done" or "cancel"

mycluster ? remove net network 192.168.1.0
mycluster ? done
Successfully modified cluster mycluster
```

While there are networks defined, the cluster will try to use the highest priority network and failover as needed to the lower priority networks as possible. Deleting all networks will return the cluster to the default mode, in which a network consisting only of the priority 1 NICs is used.

Removing and Restoring Cluster Members

This section discusses removing and restoring cluster members for maintenance:

- "Removing a Metadata Server from the Cluster" on page 329
- "Restoring a Metadata Server to the Cluster" on page 330
- "Removing a Single Client-Only Node from the Cluster" on page 331
- "Restoring a Single Client-Only Node to the Cluster" on page 332
- "Stopping CXFS for the Entire Cluster" on page 333
- "Restarting the Entire Cluster" on page 334

These procedures are the absolute safest way to perform these tasks but in some cases are not the most efficient. They should be followed if you have having problems using standard operating procedures (performing a stop/start of CXFS services or a simple host shutdown or reboot).

Removing a Metadata Server from the Cluster

If you have a cluster with multiple active metadata servers and you must perform maintenance on one of them, you must stop CXFS services on it.

Do the following:

1. Stop the CXFS services for the `exMDS` node using either the GUI or the `cmgr` command running on another metadata server. For example:

```
anotherAdmin# stop cx_services on node exMDS for cluster clustername force
```

2. Reboot or power-down the `exMDS` node.

If you do not want the cluster administration daemons and the CXFS control daemon to run during maintenance, execute the following commands:

- IRIX:

```
irix-exMDS# chkconfig cxfs_cluster off
irix-exMDS# chkconfig cluster off
```

- Linux:

```
[root@linux-exMDS root]# chkconfig cxfs off
[root@linux-exMDS root]# chkconfig cxfs_cluster off
```

If you do an upgrade of the cluster software, these arguments will be automatically reset to `on` and the cluster administration daemons and the CXFS control daemon will be started.

Note: In a system reset configuration, `exMDS` will be reset shortly after losing its membership. The machine will also be configured to reboot automatically instead of stopping in the PROM. This means that you must watch the console and intervene manually to prevent a full reboot.

In a fencing configuration, `exMDS` will lose access to the SAN when it is removed from the cluster membership

---

For more information, see "CXFS `chkconfig` Arguments" on page 289.

Restoring a Metadata Server to the Cluster

To restore a metadata server to the cluster, do the following:

1. Allow the cluster administration daemons and CXFS control daemon to be started upon reboot:

   - IRIX:

     ```
     irix-exMDS# chkconfig cxfs_cluster on
     irix-exMDS# chkconfig cluster on
     ```

   - Linux:

     ```
     [root@linux-exMDS root]# chkconfig cxfs on
     [root@linux-exMDS root]# chkconfig cxfs_cluster on
     ```

2. Immediately start cluster administration daemons on the node:

   - IRIX:

     ```
     exMDS# /etc/init.d/cluster start
     ```

- Linux:

  ```
  exMDS# /etc/init.d/cxfs_cluster start
  ```

3. Immediately start the CXFS control daemon on the node:

   ```
   exMDS# /etc/init.d/cxfs start
   ```

4. Start CXFS services on this node from another CXFS administration node:

```
anotherAdmin# start cx_services on node exMDS for cluster clustername force
```

Removing a Single Client-Only Node from the Cluster

To remove a single client-only node from the cluster, do the following:

1. Verify that the configuration is consistent by running the following on each active metadata server and comparing the output:

   ```
   MDS# /usr/cluster/bin/clconf_info
   ```

   If the client is not consistent with the metadata servers, or if the metadata servers are not consistent, then you should abort this procedure and address the health of the cluster. If a client is removed while the cluster is unstable, attempts to get the client to rejoin the cluster are likely to fail.

2. Flush the system buffers on the client you want to remove in order in order to minimize the amount of buffered information that may be lost:

   ```
   client# sync
   ```

3. Stop CXFS services on the client:

   ```
   client# /etc/init.d/cxfs_client stop
   client# chkconfig cxfs_client off
   ```

   Note: The path to `cxfs_client` varies across the operating systems supported by CXFS. For more information, see *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

4. Verify that CXFS services have stopped:

- Verify that the CXFS client daemon is not running on the client (success means no output):

  ```
  client# ps -ef | grep cxfs_client
  client#
  ```

- Monitor the `cxfs_client` log on the client you wish to remove and look for filesystems that are unmounting successfully. For example:

```
Apr 18 13:00:06 cxfs_client: cis_setup_fses Unmounted green0: green0 from /cxfs/green0
```

- Monitor the `SYSLOG` on the active metadata server and look for membership delivery messages that do not contain the removed client. For example, the following message indicates that cell 2, the node being shut down, is not included in the membership:

```
Apr 18 13:01:03 5A:o200a unix: NOTICE: Membership delivered. Membership contains 0(7) 1(5)  cells
```

- Use the following command to show that filesystems are not mounted:

  ```
  client# df -hl
  ```

5. Verify that the configuration is consistent and does not contain the removed client by running the following on each active metadata server and comparing the output:

   ```
   MDS# /usr/cluster/bin/clconf_info
   ```

Restoring a Single Client-Only Node to the Cluster

To restore a single client-only node to the cluster, do the following:

1. Verify that the configuration is consistent by running the following on each active metadata server and comparing the output:

   ```
   MDS# /usr/cluster/bin/clconf_info
   ```

2. Start CXFS on the client-only node:

   ```
   client# /etc/init.d/cxfs_client on
   client# chkconfig cxfs_client start
   ```

   Note: The path to `cxfs_client` varies across the operating systems supported by CXFS. For more information, see *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

3. Verify that CXFS has started:

   - Verify that the CXFS client daemon is running on the client-only node:

```
client# ps -ef | grep cxfs_client
                  root        716          1  0 12:59:14 ?       0:05 /usr/cluster/bin/cxfs_client
```

   - Monitor the `SYSLOG` on the active metadata server and look for a cell discovery message for the client and a membership delivered message containing the client cell. For example (line breaks added for readability):

```
Apr 18 13:07:21 4A:o200a unix: WARNING: Discovered cell 2 (woody) [priority 1 at 128.162.240.41
     via 128.162.240.34]
Apr 18 13:07:31 5A:o200a unix: NOTICE: Membership delivered. Membership contains 0(9) 1(7) 2(1)  cells
```

   - Monitor the `cxfs_client` log on the client you and look for filesystem mounts that are processing successfully. For example:

```
Apr 18 13:06:56 cxfs_client: cis_setup_fses Mounted green0: green0 on /cxfs/green0
```

   - Use the following command to show that filesystems are mounted:

     ```
     client# df -hl
     ```

4. Verify that the configuration is consistent and contains the client by running the following on each active metadata server and comparing the output:

   ```
   MDS# /usr/cluster/bin/clconf_info
   ```

Stopping CXFS for the Entire Cluster

To stop CXFS for the entire cluster, do the following:

1. Stop CXFS services on a client-only node:

   ```
   client# /etc/init.d/cxfs_client stop
   ```

   Repeat this step on each client-only node.

2. Stop CXFS services on a metadata server:

   ```
   MDS# /etc/init.d/cxfs stop
   ```

   Repeat this step on each potential metadata server.

3. Stop the cluster daemons on a metadata server:

   - IRIX:

     ```
     irixMDS# /etc/init.d/cluster stop
     ```

   - Linux:

     ```
     [root@linuxMDS root]# /etc/init.d/cxfs_cluster stop
     ```

   Repeat this step on each potential metadata server.

Restarting the Entire Cluster

To restart the entire cluster, do the following:

1. Start the cluster daemons on a potential metadata server:

   - IRIX:

     ```
     /etc/init.d/cluster start
     ```

   - Linux:

     ```
     /etc/init.d/cxfs_cluster start
     ```

   Repeat this step on each potential metadata server.

2. Start CXFS services on a metadata server:

   ```
   MDS# /etc/init.d/cxfs start
   ```

   Repeat this step on each potential metadata server.

3. Start CXFS services on a client-only node:

   ```
   client# /etc/init.d/cxfs_client start
   ```

   Repeat this step on each client-only node.

Discovering the WWNs

The `cxfs-reprobe` script enumerates the worldwide names (WWNs) on the host that are known to CXFS. For example, from an IRIX metadata server with two single-port HBAs:

```
irix# /var/cluster/clconfd-scripts/cxfs-enumerate-wwns
# cxfs-enumerate-wwns
# scsi @ /hw/scsi_ctlr/0/bus
# scsi @ /hw/scsi_ctlr/1/bus
# scsi @ /hw/scsi_ctlr/2/bus
# scsi @ /hw/scsi_ctlr/3/bus
210000e08b12ba14
# scsi @ /hw/scsi_ctlr/4/bus
210100e08b32ba14
```

Mapping XVM Volumes to Storage Targets

This section discusses mapping XVM volumes to storage targets on the IRIX and SGI ProPack for Linux platforms.

Mapping XVM Volumes to Storage Targets on IRIX

Do the following:

1. Get visible controller port WWNs.
2. Display the desired fields:

```
ls -d /dev/dsk/* | egrep -v "dks|root|swap" | cut -f4 -d"/" | sort -u
```

Mapping XVM Volumes to Storage Targets on ProPack3

Do the following for ProPack 3:

1. Get visible controller port WWNs.
2. Display the desired fields:

```
ls -d /dev/xscsi/pci*/node* | cut -f5 -d"/" | sort -u | cut -f2 -d"e"
```

Cluster Database Management

This chapter contains the following:

- "Performing Cluster Database Backup and Restoration"
- "Checking the Cluster Configuration with `cxfs-config`" on page 342

Performing Cluster Database Backup and Restoration

You should perform a database backup whenever you want to save the database and be able to restore it to the current state at a later point.

You can use the following methods to restore the database:

- If the database is accidentally deleted from a node, use the `fs2d` daemon to replicate the database from another node in the pool.
- If you want to be able to recreate the current configuration, use the `build_cmgr_script` script. You can then recreate this configuration by running the script generated.
- If you want to retain a copy of the database and all node-specific information such as local logging, use the `cdbBackup` and `cdbRestore` commands. You should periodically backup the cluster database on all administration nodes using the `cdbBackup` command either manually or by adding an entry to the `root crontab` file.

Restoring the Database from Another Node

If the database has been accidentally deleted from an individual administration node, you can replace it with a copy from another administration node. Do not use this method if the cluster database has been corrupted.

Do the following:

1. Stop the CXFS daemons by running the following command on each administration node:

   - IRIX:

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

   - Linux:

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

2. Run `cdbreinit` on administration nodes that are missing the cluster database.

3. Restart the daemons by running the following commands on each administration node:

   - IRIX:

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

   - Linux:

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

   The `fs2d` daemon will then replicate the cluster database to those nodes from which it is missing

Using `build_cmgr_script` for the Cluster Database

You can use the `build_cmgr_script` command from one node in the cluster to create a `cmgr` script that will recreate the node, cluster, switch, and filesystem definitions for all nodes in the cluster database. You can then later run the resulting script to recreate a database with the same contents; this method can be used for missing or corrupted cluster databases.

Note: The `build_cmgr_script` script does not contain local logging information, so it cannot be used as a complete backup/restore tool.

To perform a database backup, use the `build_cmgr_script` script from one node in the cluster, as described in "Creating a `cmgr` Script Automatically" on page 280.

Caution: Do not make configuration changes while you are using the `build_cmgr_script` command.

By default, this creates a `cmgr` script in the following location:

`/var/cluster/ha/tmp/cmgr_create_cluster`*_clustername_processID*

You can specify another filename by using the `-o` option.

To perform a restore on all nodes in the pool, do the following:

1. Stop CXFS services on all nodes in the cluster.
2. Stop the cluster database daemons on each node.
3. Remove all copies of the old database by using the `cdbreinit` command on each node.
4. Execute the `cmgr` script (which was generated by the `build_cmgr_script` script) on the node that is defined first in the script. This will recreate the backed-up database on each node.

   Note: If you want to run the generated script on a different node, you must modify the generated script so that the node is the first one listed in the script.

5. Restart cluster database daemons on each node.

For example, to backup the current database, clear the database, and restore the database to all administration nodes, do the following on administration nodes as directed:

On one node:

```
# /var/cluster/cmgr-scripts/build_cmgr_script -o /tmp/newcdb
Building cmgr script for cluster clusterA ...
build_cmgr_script: Generated cmgr script is /tmp/newcdb
```

On one node:

```
# stop cx_services for cluster clusterA
```

On each node:

IRIX:

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

Linux:

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

On each:

```
# /usr/cluster/bin/cdbreinit
```

On each:

IRIX:

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

Linux:

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

*On the *first* node listed in the /tmp/newcdb script:*

```
# /tmp/newcdb
```

Using `cdbBackup` and `cdbRestore` for the Cluster Database and Logging Information

The `cdbBackup` and `cdbRestore` commands backup and restore the cluster database and node-specific information, such as local logging information. You must run these commands individually for each node.

To perform a backup of the cluster, use the `cdbBackup` command on each node.

Caution: Do not make configuration changes while you are using the `cdbBackup` command.

To perform a restore, run the `cdbRestore` command on each node. You can use this method for either a missing or a corrupted cluster database. Do the following:

1. Stop CXFS services on all nodes in the cluster.
2. Stop the cluster administration daemons on each administration node.
3. Remove the old database by using the `cdbreinit` command on each node.

4. Stop the cluster administration daemons again (these were restarted automatically by `cdbreinit` in the previous step) on each node.
5. Use the `cdbRestore` command on each node.
6. Start the cluster administration daemons on each node.

For example, to backup the current database, clear the database, and then restore the database to all administration nodes, do the following as directed on administration nodes in the cluster:

```
On each node:
# /usr/cluster/bin/cdbBackup

On one node:
# stop cx_services for cluster clusterA

On each node:
     IRIX:
     # /etc/init.d/cluster stop

     Linux:
     # /etc/init.d/cxfs_cluster stop

On each:
# /usr/cluster/bin/cdbreinit

On each node (again):
     IRIX:
     # /etc/init.d/cluster stop

     Linux:
     # /etc/init.d/cxfs_cluster stop

On each node:
# /usr/cluster/bin/cdbRestore

On each node:
     IRIX:
     # /etc/init.d/cluster start

     Linux:
```

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

For more information, see the `cdbBackup` and `cdbRestore` man page.

Checking the Cluster Configuration with `cxfs-config`

The `cxfs-config` command displays and checks configuration information in the cluster database. You can run it on any administration node in the cluster.

By default, `cxfs-config` displays the following:

- Cluster name and cluster ID
- Tiebreaker node
- Networks for CXFS failover networks
- Nodes in the pool:
  - Node ID
  - Cell ID (as assigned by the kernel when added to the cluster and stored in the cluster database)
  - Status of CXFS services (configured to be enabled or disabled)
  - Operating system
  - Node function
- CXFS filesystems:
  - Name, mount point (`enabled` means that the filesystem is configured to be mounted; if it is not mounted, there is an error)
  - Device name
  - Mount options
  - Potential metadata servers
  - Nodes that should have the filesystem mounted (if there are no errors)
  - Switches:

  - Switch name, user name to use when sending a `telnet` message, mask (a hexadecimal string representing a 64-bit port bitmap that indicates the list of ports in the switch that will not be fenced)
  - Ports on the switch that have a client configured for fencing at the other end
- Warnings or errors

For example:

```
thump# /usr/cluster/bin/cxfs-config
Global:
    cluster: topiary (id 1)
    tiebreaker: <none>

Networks:
    net 0: type tcpip  192.168.0.0      255.255.255.0
    net 1: type tcpip  134.14.54.0      255.255.255.0

Machines:
    node leesa: node 6     cell 2  enabled  Linux32 client_only
        fail policy: Fence
        nic 0: address: 192.168.0.164 priority: 1 network: 0
        nic 1: address: 134.14.54.164 priority: 2 network: 1

    node thud: node 8     cell 1  enabled  IRIX    client_admin
        fail policy: Fence
        nic 0: address: 192.168.0.204 priority: 1 network: 0
        nic 1: address: 134.14.54.204 priority: 2 network: 1

    node thump: node 1     cell 0  enabled  IRIX    server_admin
        fail policy: Fence
        nic 0: address: 192.168.0.186 priority: 1 network: 0
        nic 1: address: 134.14.54.186 priority: 2 network: 1

Filesystems:
    fs dxm: /mnt/dxm             enabled
        device = /dev/cxvm/tp9500a4s0
        options = []
        servers = thump (1)
        clients = leesa, thud, thump
```

```
Switches:
    switch 0: admin@asg-fcsw1      mask 0000000000000000
        port 8: 210000e08b0ead8c thump
        port 12: 210000e08b081f23 thud

    switch 1: admin@asg-fcsw0      mask 0000000000000000

Warnings/errors:
    enabled machine leesa has fencing enabled but is not present in switch database
```

The following options are of particular interest:

- `-all` lists all available information
- `-ping` contacts each NIC in the machine list and displays if the packets is transmitted and received. For example:

```
node leesa: node 6     cell 2  enabled  Linux32 client_only
   fail policy: Fence
   nic 0: address: 192.168.0.164 priority: 1
       ping: 5 packets transmitted, 5 packets received, 0.0% packet loss
       ping: round-trip min/avg/max = 0.477/0.666/1.375 ms
   nic 1: address: 134.14.54.164 priority: 2
       ping: 5 packets transmitted, 5 packets received, 0.0% packet loss
       ping: round-trip min/avg/max = 0.469/0.645/1.313 ms
```

- `-xfs` lists XFS information for each CXFS filesystem, such as size. For example:

```
Filesystems:
    fs dxm: /mnt/dxm             enabled
        device = /dev/cxvm/tp9500a4s0
        options = []
        servers = thump (1)
        clients = leesa, thud, thump
        xfs:
            magic: 0x58465342
            blocksize: 4096
            uuid: 3459ee2e-76c9-1027-8068-0800690dac3c
            data size 17.00 Gb
```

- `-xvm` lists XVM information for each CXFS filesystem, such as volume size and topology. For example:

```
Filesystems:
    fs dxm: /mnt/dxm             enabled
        device = /dev/cxvm/tp9500a4s0
        options = []
        servers = thump (1)
        clients = leesa, thud, thump
        xvm:
            vol/tp9500a4s0                    0 online,open
                subvol/tp9500a4s0/data     35650048 online,open
                    slice/tp9500a4s0           35650048 online,open

            data size: 17.00 Gb
```

- `-check` performs extra verification, such as XFS filesystem size with XVM volume size for each CXFS filesystem. This option may take a few moments to execute.

The following example shows errors reported by `cxfs-config`:

```
aiden # /usr/cluster/bin/cxfs-config -check -all
Global:
    cluster: BP (id 555)
    cluster state: enabled
    tiebreaker:
Networks:
    net 0: type tcpip  10.11.0.0        255.255.255.0
    net 1: type tcpip  128.162.242.0    255.255.255.0

Machines:
    node aiden: node 27560 cell 0  enabled  IRIX    server_admin
        hostname: aiden.americas.sgi.com
        fail policy: Fence, Shutdown
        nic 0: address: 10.11.0.241 priority: 1 network: 0
            ping: 5 packets transmitted, 5 packets received, 0.0% packet loss
            ping: round-trip min/avg/max = 0.136/0.171/0.299 ms
        nic 1: address: 128.162.242.12 priority: 2 network: 1
            ping: 5 packets transmitted, 5 packets received, 0.0% packet loss
            ping: round-trip min/avg/max = 0.130/0.171/0.303 ms

    node brigid: node 31867 cell 2  enabled  IRIX    server_admin
```

```
        hostname: brigid.americas.sgi.com
        fail policy: Fence, Shutdown
        nic 0: address: 10.11.0.240 priority: 1 network: 0
            ping: 5 packets transmitted, 5 packets received, 0.0% packet loss
            ping: round-trip min/avg/max = 0.303/0.339/0.446 ms
        nic 1: address: 128.162.242.11 priority: 2 network: 1
            ping: 5 packets transmitted, 5 packets received, 0.0% packet loss
            ping: round-trip min/avg/max = 0.336/0.430/0.799 ms

    node flynn: node 1     cell 1  enabled  linux64 client_only
        hostname: flynn.americas.sgi.com
        fail policy: Fence, Shutdown
        nic 0: address: 10.11.0.234 priority: 1 network: 0
            ping: 5 packets transmitted, 5 packets received, 0.0% packet loss
            ping: round-trip min/avg/max = 0.323/0.370/0.539 ms
        nic 1: address: 128.162.242.189 priority: 2 network: 1
            ping: 5 packets transmitted, 5 packets received, 0.0% packet loss
            ping: round-trip min/avg/max = 0.283/0.312/0.424 ms

Filesystems:
    fs concatfs: /concatfs            enabled
        device = /dev/cxvm/concatfs
        force = true
        options = [rw,quota]
        servers = aiden (1), brigid (2)
        clients = aiden, brigid, flynn
        xvm:
            vol/concatfs                      0 online,open
                subvol/concatfs/data     2836134016 online,open
                subvol/concatfs/data     2836134016 online,open
                    concat/concat0           2836134016 online,tempname,open
                        slice/lun2s0             1418067008 online,open
                        slice/lun3s0             1418067008 online,open

            data size: 1.32 TB
        xfs:
            magic: 0x58465342
            blocksize: 4096
            uuid: 9616ae39-3a50-1029-8896-080069056bf5
            data size 1.32 TB
    fs stripefs: /stripefs            enabled
        device = /dev/cxvm/stripefs
        force = true
        options = [rw,quota]
        servers = aiden (1), brigid (2)
        clients = aiden, brigid, flynn
        xvm:
            vol/stripefs                      0 online,open
                subvol/stripefs/data     2836133888 online,open
                    stripe/stripe0           2836133888 online,tempname,open
                        slice/lun0s0             1418067008 online,open
                        slice/lun1s0             1418067008 online,open

            data size: 1.32 TB
        xfs:
            magic: 0x58465342
            blocksize: 4096
            uuid: 9616ae38-3a50-1029-8896-080069056bf5
            data size 1.32 TB

Switches:
    switch 0: 32 port brocade admin@fcswitch12         port 12: 210000e08b00e6eb brigid
        port 28: 210000e08b041a3a aiden

    switch 1: 32 port brocade admin@fcswitch13         port 7: 210000e08b08793f flynn
        port 12: 210100e08b28793f flynn

cxfs-config warnings/errors:
    server aiden fail policy must not contain "Shutdown" for cluster with
even number of enabled servers and no tiebreaker
    server brigid fail policy must not contain "Shutdown" for cluster with
even number of enabled servers and no tiebreaker
```

For a complete list of options, see the `cxfs-config` man page.

Chapter 13

Coexecution with FailSafe

This chapter discusses the following:

Also see "Communication Paths in a Coexecution Cluster" on page 453.

Why Run CXFS and FailSafe Together?

CXFS allows groups of computers to coherently share large amounts of data while maintaining high performance.

The SGI FailSafe product provides a general facility for providing highly available (HA) services. If one of the administration nodes in the cluster or one of the node's components fails, a different administration node in the cluster restarts the HA services of the failed node. To CXFS clients, the services on the replacement node are indistinguishable from the original services before failure occurred. It appears as if the original node has crashed and rebooted quickly. The CXFS clients notice only a brief interruption in the HA service.

FailSafe assumes that CXFS filesystems are highly available and will recover from CXFS failures (including loss of CXFS membership). FailSafe will wait for CXFS to recover CXFS filesystems before the resource group containing the CXFS resource is started on another FailSafe node in the cluster.

You can therefore use FailSafe in a CXFS cluster (known as *coexecution*) to provide HA services (such as NFS or web) running on a CXFS filesystem. This combination provides high-performance shared data access for highly available applications in a clustered system.

Coexecution Release Levels

CXFS 6.5.10 or later and IRIS FailSafe 2.1 or later (plus relevant patches) may be installed and run on the same system.

Size of the Coexecution Cluster

A subset of administration nodes in a coexecution cluster can be configured to be used as FailSafe nodes; a coexecution cluster can have up to eight nodes that run FailSafe.

All nodes in a CXFS cluster will run CXFS, and up to eight of those administration nodes can also run FailSafe. All administration nodes must run IRIX (FailSafe is not supported on Linux). Even when you are running CXFS and FailSafe, there is still only one pool, one cluster, and one cluster configuration.

It is recommended that a production cluster be configured with an odd number of server-capable nodes. (A cluster with reset cables and only two server-capable nodes is supported, but there are inherent issues with this configuration; see "CXFS Recovery Issues in a Cluster with Only Two Server-Capable Nodes " on page 476.)

Cluster Type

The cluster can be one of three types:

- `FailSafe`. In this case, all nodes will also be of type `FailSafe`. The nodes must all be administration nodes.
- `CXFS`. In this case, all nodes will be of type `CXFS`. The nodes can be either administration nodes or client-only nodes.

- `CXFS and FailSafe` (coexecution). In this case, all nodes will be a mix of type `CXFS` (any nodes running other operating systems) and type `CXFS and FailSafe` (administration nodes), using FailSafe for application-level high availability and CXFS.

Note: Although it is possible to configure a coexecution cluster with type `FailSafe` only nodes, SGI does not support this configuration.

Figure 13-1 describes some of the various legal and illegal combinations.

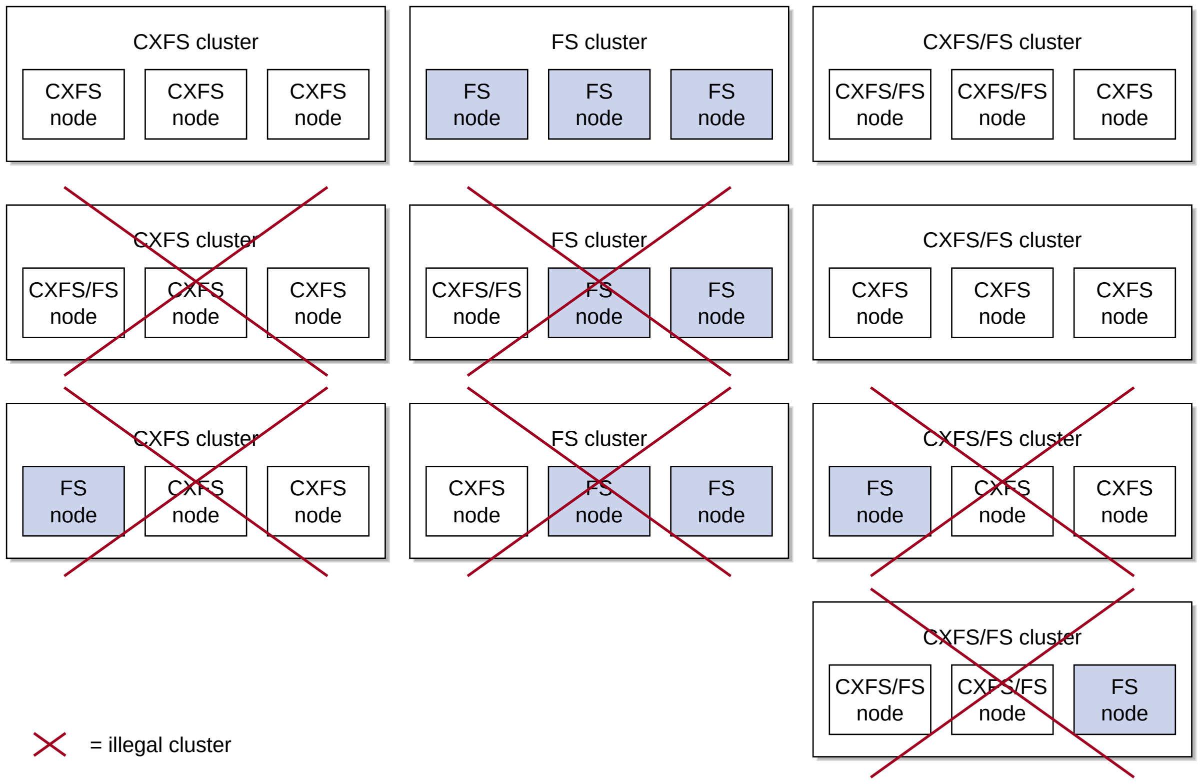


Figure 13-1 Cluster and Node Type Combinations

Metadata Server Node Types

All potential metadata server nodes must be of one of the following types:

- `CXFS`
- `CXFS and FailSafe`

Separate GUIs

There is one `cmgr` (`cluster_mgr`) command but separate graphical user interfaces (GUIs) for CXFS and for FailSafe. You must manage CXFS configuration with the CXFS GUI and FailSafe configuration with the FailSafe GUI; you can manage both with `cmgr`.

Conversion

Using the CXFS GUI or `cmgr`, you can convert an existing `FailSafe` cluster and nodes to type `CXFS` or to type `CXFS and FailSafe`. You can perform a parallel action using the FailSafe GUI. A converted node can be used by FailSafe to provide application-level high-availability and by CXFS to provide clustered filesystems. See "Set Up an Existing FailSafe Cluster for CXFS with the GUI" on page 170.

However:

- You cannot change the type of a node if the respective HA or CXFS services are active. You must first stop the services for the node.
- The cluster must support all of the functionalities (FailSafe and/or CXFS) that are turned on for its nodes; that is, if your cluster is of type `CXFS`, then you **cannot** modify a node that is already part of the cluster so that it is of type `FailSafe`. However, the nodes do not have to support all the functionalities of the cluster; that is, you can have a `CXFS` node in a `CXFS and FailSafe` cluster.

See "Convert a Node to CXFS or FailSafe with `cmgr`" on page 240, and "Convert a Cluster to CXFS or FailSafe with `cmgr`" on page 251.

Network Interfaces

For FailSafe, you must have at least two network interfaces. However, CXFS uses only one interface for **both** heartbeat and control messages. (The CXFS GUI appears to let you select only heartbeat or only control for a network, but you must not choose these selections.)

Metadata Servers and Failover Domain

The metadata server list must exactly match the failover domain list (the names and the order of names).

CXFS Resource Type for FailSafe

FailSafe provides a `CXFS` resource type that can be used to fail over applications that use CXFS filesystems. CXFS resources must be added to the resource group that contain the resources that depend on a CXFS filesystem. The `CXFS` resource type name is the CXFS filesystem mount point.

The `CXFS` resource type has the following characteristics:

- It does not start all resources that depend on CXFS filesystem until the CXFS filesystem is mounted on the local node.
- The `start` and `stop` action scripts for the `CXFS` resource type do not mount and unmount CXFS filesystems, respectively. (The `start` script waits for the CXFS filesystem to become available; the `stop` script does nothing but its existence is required by FailSafe.) Users should use the CXFS GUI or `cmgr` command to mount and unmount CXFS filesystems.
- It monitors CXFS filesystem for failures.
- Optionally, for applications that must run on a CXFS metadata server, the `CXFS` resource type relocates the CXFS metadata server when there is an application failover. In this case, the application failover domain (AFD) for the resource group should consists of the CXFS metadata server and the meta-data server backup nodes.

The CXFS filesystems that an NFS server exports should be mounted on all nodes in the failover domain using the CXFS GUI or the `cmgr` command.

For example, following are the commands used to create resources `NFS`, `CXFS`, and `statd_unlimited` based on a CXFS filesystem mounted on `/FC/lun0_s6`. (This example assumes that you have defined a cluster named `test-cluster` and have already created a failover policy named `cxfs-fp` and a resource group named `cxfs-group` based on this policy. Line breaks added for readability.)

```
cmgr> define resource /FC/lun0_s6 of resource_type CXFS in cluster test-cluster
Enter commands, when finished enter either "done" or "cancel"

Type specific attributes to create with set command:

Type Specific Attributes - 1: relocate-mds


No resource type dependencies to add

resource /FC/lun0_s6 ? set relocate-mds to false
resource /FC/lun0_s6 ? done


============================================

cmgr> define resource /FC/lun0_s6 of resource_type NFS in cluster test-cluster
Enter commands, when finished enter either "done" or "cancel"

Type specific attributes to create with set command:

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


No resource type dependencies to add

resource /FC/lun0_s6 ? set export-info to rw
resource /FC/lun0_s6 ? set filesystem to /FC/lun0_s6
resource /FC/lun0_s6 ? done


============================================
cmgr> define resource /FC/lun0_s6/statmon of resource_type statd_unlimited in cluster
test-cluster
```

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

Type specific attributes to create with set command:

Type Specific Attributes - 1: ExportPoint


Resource type dependencies to add:

Resource Dependency Type - 1: NFS


resource /FC/lun0_s6/statmon ? set ExportPoint to /FC/lun0_s6
resource /FC/lun0_s6/statmon ? add dependency /FC/lun0_s6 of type NFS
resource /FC/lun0_s6/statmon ? done

==============================================
cmgr> define resource_group cxfs-group in cluster test-cluster
Enter commands, when finished enter either "done" or "cancel"

resource_group cxfs-group ? set failover_policy to cxfs-fp
resource_group cxfs-group ? add resource /FC/lun0_s6 of resource_type NFS
resource_group cxfs-group ? add resource /FC/lun0_s6 of resource_type CXFS
resource_group cxfs-group ? add resource /FC/lun0_s6/statmon of resource_type statd_unlimited
resource_group cxfs-group ? done
```

For more information about resource groups and failover domains, see the *FailSafe Administrator's Guide for SGI InfiniteStorage*.

Chapter 14

Trusted IRIX and CXFS

CXFS has been qualified in an SGI Trusted IRIX cluster with the Data Migration Facility (DMF) and Tape Management Facility (TMF).

If you want to run CXFS and Trusted IRIX, all server-capable nodes in the cluster must run Trusted IRIX. The client-only nodes can run IRIX. Other platforms are not supported in a cluster with Trusted IRIX.

Installation Tips for CXFS and Trusted IRIX

SGI recommends that you install all of the software products you intend to run (Trusted IRIX, CXFS, DMF, TMF, and so on) at the same time.

After installing these products, you must do the following:

1. From the system console, go to the system maintenance menu. For example:

   ```
   # init 0
   ```

   (If your system is set to automatically reboot to multiuser mode, you will need to press `Esc` to reach the menu.)

2. Choose `5` from the menu in order to enter the command monitor:

   ```
   System Maintenance Menu

   1) Start System
   2) Install System Software
   3) Run Diagnostics
   4) Recover System
   5) Enter Command Monitor

   Option? 5
   ```

3. Enter single user mode by using the `single` command:

   ```
   >> single
   ```

4. Enter the root password when prompted.

5. Ensure that you are in the root directory:

   ```
   # cd /
   ```

6. Set the following attributes for Trusted IRIX and CXFS:

   ```
   # suattr -C all+eip
   ```

7. Execute the Trusted IRIX configuration command, which sets the appropriate extended attributes on files:

   ```
   # /etc/trix.config
   ```

For more information, see:

- *Trusted IRIX Read Me First Notice*
- *Trusted IRIX/CMW Security Features User's Guide*

Mandatory Access Controls

In a mixed Trusted IRIX and IRIX cluster, an IRIX CXFS client will require but not have a mandatory access control (MAC) label associated with its credentials when it attempts to access a Trusted IRIX server. In order to address this, a MAC label is provided in one of the following ways:

- The filesystem can be mounted with the `eag:mac-ip=label` option to specify the label used for IRIX CXFS clients.
- If the mount option is not used, the default label in the `rhost` database entry for the IRIX original node is used.
- If the `rhost` database entry is unavailable or invalid, the following label is used: `msenlow, minthigh`.

Chapter 15

Monitoring Status

You can view the system status in the following ways:

Note: Administrative tasks must be performed using the GUI when it is connected to a CXFS administration node (a node that has the `cluster_admin` software package installed) or using the `cmgr` command when logged in to a CXFS administration node. Administration commands must be run on a CXFS administration node; the `cxfs_info` status command is run on a client-only node.

- Monitor log files. See "Status in Log Files" on page 360
- Use the GUI or the `tail` command to view the end of the system log file:
  - IRIX: `/var/adm/SYSLOG`
  - Linux: `/var/log/messages`
- Keep continuous watch on the state of a cluster using the GUI view area or the `clconf_info -e` command.
- Query the status of an individual node or cluster using either the GUI or the `cmgr` command.
- Manually test the filesystems with the `ls` command.
- Monitor the system with Performance Co-Pilot. You can use Performance Co-Pilot to monitor the read/write throughput and I/O load distribution across all disks and for all nodes in the cluster. The activity can be visualized, used to generate alarms, or archived for later analysis. You can also monitor XVM statistics. See the *Performance Co-Pilot for IA-64 Linux User's and Administrator's Guide*, *Performance Co-Pilot for IRIX Advanced User's and Administrator's Guide*, the *Performance Co-Pilot Programmer's Guide*, and the `dkvis`, `pmie`, `pmieconf`, and `pmlogger` man pages.

  Note: You must manually install the XVM statistics for the Performance Co-Pilot package; it is not installed by default. See Chapter 4, "IRIX CXFS Installation" on page 65.

The following sections describe the procedures for performing some of these tasks:

- "Status in Log Files"
- "Cluster Status" on page 362
- "Node Status" on page 365
- "XVM Statistics" on page 367
- "I/O Fencing Status" on page 369
- "Heartbeat Timeout Status" on page 370

Status in Log Files

You should monitor the following log files listed for problems:

- Administration node logs:
  - System log:
    - IRIX: `/var/adm/SYSLOG`
    - Linux: `/var/log/messages`

    Look for a `Membership delivered` message to indicate that a cluster was formed.
  - Events from the GUI and `clconfd`: `/var/cluster/ha/log/cad_log`
  - Kernel status: `/var/cluster/ha/log/clconfd_`*hostname*
  - Command line interface log:`/var/cluster/ha/log/cli_`*hostname*
  - Monitoring of other daemons:`/var/cluster/ha/log/cmond_log`
  - Reset daemon log: `/var/cluster/ha/log/crsd_`*hostname*
  - Output of the diagnostic tools such as the serial and network connectivity tests: `/var/cluster/ha/log/diags_`*hostname*
  - Cluster database membership status: `/var/cluster/ha/log/fs2d_log`

  - System administration log, which contains a list of the commands run by the GUI:
    - IRIX: `/var/sysadm/salog`
    - Linux: `/var/lib/sysadm/salog`
- Client-only node log files:
  - `cxfs_client` log file:
    - IRIX: `/var/adm/cxfs_client`
    - Linux: `/var/log/cxfs_client`
  - System log:
    - IRIX: `/var/adm/SYSLOG`
    - Linux: `/var/log/messages`

    Look for a `Membership delivered` message to indicate that a cluster was formed.
  - Output of the diagnostic tools such as the serial and network connectivity tests: `/var/cluster/ha/log/diags_`*hostname*
- The Linux platform uses the `logrotate` system utility to rotate the `cxfs_client` logs:
  - The `/etc/logrotate.conf` file specifies how often system logs are rotated
  - The `/etc/logrotate.d/cxfs_client` file specifies the manner in which `cxfs_client` logs are rotated

For information about client-only nodes running other operating systems, see *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

If the disk is filling with log messages, see "Log File Management" on page 307.

Caution: Do not change the names of the log files. If you change the names, errors can occur.

Cluster Status

You can monitor system status with the following tools:

- The CXFS GUI connected to a CXFS administration node
- The `clconf_info` or `cmgr` command on a CXFS administration node
- The `cxfs_info` command on a client-only node

Also see "Key to Icons and States" on page 166

Check Cluster Status with the GUI

The easiest way to keep a continuous watch on the state of a cluster is to use the view area and choose the following:

Edit
  > Expand All

The cluster status can be one of the following:

- **ACTIVE**, which means the cluster is up and running.
- **INACTIVE**, which means the start CXFS services task has not been run.
- **ERROR**, which means that some nodes are in a **DOWN** state; that is, the cluster **should** be running, but it is not.
- **UNKNOWN**, which means that the state cannot be determined because CXFS services are not running on the node performing the query. For more information, see in "Node Status" on page 365.

Check Cluster Status with `clconf_info`

If the cluster is up, you can see detailed information by using `/usr/cluster/bin/clconf_info` on a CXFS administration node.

The `clconf_info` command has the following options:

`-e`	Waits for events from `clconfd` and displays the new information
`-n` *nodename*	Displays information for the specified logical node name

`-p`	Persists until the membership is formed
`-q`	(Quiet mode) Decreases verbosity of output. You can repeat this option to increase the level of quiet; that is, `-qq` specifies more quiet (less output) than `-q`).
`-s`	Sorts the output alphabetically by name for nodes and by device for filesystems. By default, the output is not sorted.
`-v`	(Verbose mode) Specifies the verbosity of output (`-vv` specifies more verbosity than `-v`). Deferred implementation.

For example:

```
# /usr/cluster/bin/clconf_info

Event at [2004-04-16 09:20:59]

Membership since Fri Apr 16 09:20:56 2004
____________ ______ ________ ______ ______
Node         NodeID Status   Age    CellID
____________ ______ ________ ______ ______
leesa             0 inactive      -      0
whack             2 up           16      3
lustre            8 up            5      5
thud             88 up           16      1
cxfs2           102 DOWN          -      2
____________ ______ ________ ______ ______
2 CXFS FileSystems
/dev/cxvm/tp9500_0 on /mnt/cxfs0  enabled  server=(whack)  2 client(s)=(thud,lustre)  status=UP
/dev/cxvm/tp9500a4s0 on /mnt/tp9500a4s0  disabled  server=()  0 client(s)=()  status=DOWN
```

This command displays the following fields:

- Node name
- Node ID
- Status (`up`, `DOWN`, or `inactive`)
- Age, which indicates how many membership transitions in which the node has participated. The age is 1 the first time a node joins the membership and will

increment for each time the membership changes. This number is dynamically allocated by the CXFS software (the user does not define the age).

- CellID, which is allocated when a node is added into the cluster with the **Modify a Cluster** task (in the GUI or `cmgr`). It persists until the node is removed from the cluster with the **Modify a Cluster** task. The kernel also reports the cell ID in console messages.

Check Cluster Status with `cmgr`

To query node and cluster status, use the following `cmgr` command on a CXFS administration node:

```
cmgr> show status of cluster cluster_name
```

Check Cluster/Node/Filesystem Status with `cxfs_info`

The `cxfs_info` command provides information about the cluster status, node status, and filesystem status. `cxfs_info` is run from a client-only node:

```
/usr/cluster/bin/cxfs_info
```

You can use the `-e` option to display information continuously, updating the screen when new information is available; use the `-c` option to clear the screen between updates. For less verbose output, use the `-q` (quiet) option.

For example, on a Solaris node named `cxfssun4`:

```
cxfssun4 # /usr/cxfs_cluster/bin/cxfs_info
cxfs_client status [timestamp Sep 03 12:16:06 / generation 18879]

Cluster:
    sun4 (4) - enabled
Local:
    cxfssun4 (2) - enabled, state: stable, cms: up, xvm: up, fs: up
Nodes:
    cxfs27     enabled  up    1
    cxfs28     enabled  up    0
    cxfsnt4    enabled  up    3
    cxfssun4   enabled  up    2
    mesabi     enabled  DOWN  4
```

```
Filesystems:
    lun1s0     enabled  mounted          lun1s0              /lun1s0
    mirror0    disabled unmounted        mirror0             /mirror0
```

Node Status

To query the status of a node, you provide the logical name of the node. The node status can be one of the following:

- **UP**, which means that CXFS services are started and the node is part of the CXFS kernel membership. For more information, see "CXFS Kernel Membership, Quorum, and Tiebreaker" on page 462.
- **DOWN**, which means that although CXFS services are started and the node is defined as part of the cluster, the node is not in the current CXFS kernel membership.
- **INACTIVE**, which means that the start CXFS services task has not been run.
- **UNKNOWN**, which means that the state cannot be determined because CXFS services are not running on the node performing the query.

State information is exchanged by daemons that run only when CXFS services are started. A given CXFS administration node must be running CXFS services in order to report status on other nodes.

For example, CXFS services must be started on `node1` in order for it to show the status of `node2`. If CXFS services are started on `node1`, then it will accurately report the state of all other nodes in the cluster. However, if `node1`'s CXFS services are not started, it will report the following states:

- **INACTIVE** for its own state, because it can determine that the start CXFS services task has not been run
- **UNKNOWN** as the state of all other nodes, because the daemons required to exchange information with other nodes are not running, and therefore state cannot be determined

The following sections provide different methods to monitor node status. Also see "Check Cluster/Node/Filesystem Status with `cxfs_info`" on page 364.

Monitoring Node Status with the GUI

You can use the view area to monitor the status of the nodes. Select **View: Nodes and Cluster**.

To determine whether a node applies to CXFS, to FailSafe, or both, double-click the node name in the display.

Querying Node Status with `cmgr`

To query node status, use the following `cmgr` command:

```
cmgr> show status of node node_name
```

Monitoring Node Status with `clconf_info`

You can use the `clconf_info` command to monitor the status of the nodes in the cluster.

For example:

```
# /usr/cluster/bin/clconf_info

Event at [2004-04-16 09:20:59]

Membership since Fri Apr 16 09:20:56 2004
____________ ______ ________ ______ ______
Node         NodeID Status   Age    CellID
____________ ______ ________ ______ ______
leesa             0 inactive      -      0
whack             2 up           16      3
lustre            8 up            5      5
thud             88 up           16      1
cxfs2           102 DOWN          -      2
____________ ______ ________ ______ ______
2 CXFS FileSystems
/dev/cxvm/tp9500_0 on /mnt/cxfs0  enabled  server=(whack)  2 client(s)=(thud,lustre)  status=UP
/dev/cxvm/tp9500a4s0 on /mnt/tp9500a4s0  disabled  server=()  0 client(s)=()  status=DOWN
```

Pinging the System Controller with `cmgr`

When CXFS is running, you can determine whether the system controller on a node is responding by using the following `cmgr` command:

```
cmgr> admin ping node node_name
```

This command uses the CXFS daemons to test whether the system controller is responding.

You can verify reset connectivity on a node in a cluster even when the CXFS daemons are not running by using the `standalone` option of the `admin ping` command:

```
cmgr> admin ping standalone node node_name
```

This command calls the `ping` command directly to test whether the system controller on the indicated node is responding.

Monitoring Reset Lines with `cmgr`

You can use the `cmgr` command to ping the system controller at a node as follows (line break for readability):

```
cmgr> admin ping dev_name device_name of dev_type device_type
with sysctrl_type system_controller_type
```

XVM Statistics

Note: This feature assumes that you have installed the `pcp_eoe` and `pcp_eoe.sw.xvm` packages; see Chapter 4, "IRIX CXFS Installation" on page 65.

You can use Performance Co-Pilot to monitor XVM statistics. To do this, you must enable the collection of statistics:

- To enable the collection of statistics for the local host, enter the following:

  ```
  $ pmstore xvm.control.stats_on 1
  ```

- To disable the collection of statistics for the local host, enter the following:

  ```
  $ pmstore xvm.control.stats_on 0
  ```

You can gather XVM statistics in the following ways:

- By using the `pmval` command from the IRIX `pcp_eoe.sw.monitor` package and the Linux bit `pcp` RPM. It can be used to produce an ASCII report of selected metrics from the `xvm` group in the Performance Co-Pilot namespace of available metrics.
- By using the optional `pmgxvm` command provided with the Performance Co-Pilot `pcp.sw.monitor` package (an optional product available for purchase).

  If you have the `pcp.sw.monitor` package, you can also use the `pmchart` command to view time-series data in the form of a moving graph. Figure 15-1 shows an example.

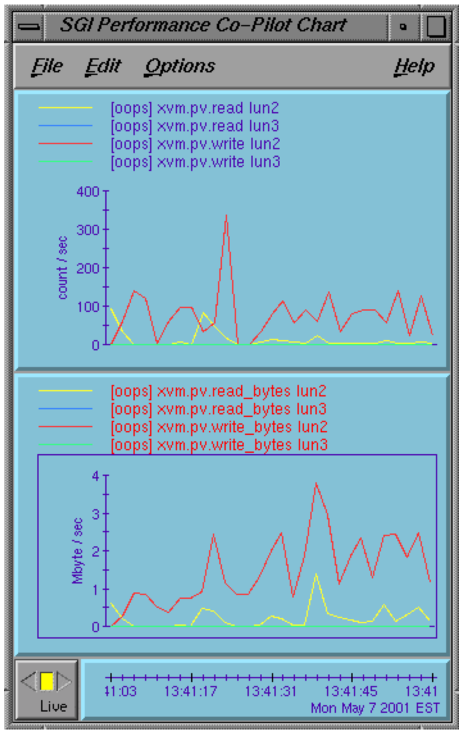


Figure 15-1 pmgxvm chart

I/O Fencing Status

To check the current fencing status, select **View: Switches** in the GUI view area, or use the `admin fence query` command in `cmgr`, or use the `hafence` command as follows:

```
/usr/cluster/bin/hafence -q
```

For example, the following output shows that all nodes are enabled.

```
# /usr/cluster/bin/hafence -q
  Switch[0] "ptg-brocade" has 8 ports
    Port 1 type=FABRIC status=enabled  hba=210000e08b0102c6 on host thunderbox
    Port 2 type=FABRIC status=enabled  hba=210000e08b01fec5 on host whack
    Port 5 type=FABRIC status=enabled  hba=210000e08b027795 on host thump
    Port 6 type=FABRIC status=enabled  hba=210000e08b019ef0 on host thud
```

A fenced port shows `status=disabled`. For example:

```
# /usr/cluster/bin/hafence -q
  Switch[0] "brocade04" has 16 ports
    Port 4 type=FABRIC status=enabled  hba=210000e08b0042d8 on host o200c
    Port 5 type=FABRIC status=enabled  hba=210000e08b00908e on host cxfs30
    Port 9 type=FABRIC status=enabled  hba=2000000173002d3e on host cxfssun3
```

Verbose (`-v`) output would be as follows:

```
# /usr/cluster/bin/hafence -v
  Switch[0] "brocade04" has 16 ports
    Port 0 type=FABRIC status=enabled  hba=2000000173003b5f on host UNKNOWN
    Port 1 type=FABRIC status=enabled  hba=2000000173003adf on host UNKNOWN
    Port 2 type=FABRIC status=enabled  hba=210000e08b023649 on host UNKNOWN
    Port 3 type=FABRIC status=enabled  hba=210000e08b021249 on host UNKNOWN
    Port 4 type=FABRIC status=enabled  hba=210000e08b0042d8 on host o200c
    Port 5 type=FABRIC status=enabled  hba=210000e08b00908e on host cxfs30
    Port 6 type=FABRIC status=enabled  hba=2000000173002d2a on host UNKNOWN
    Port 7 type=FABRIC status=enabled  hba=2000000173003376 on host UNKNOWN
    Port 8 type=FABRIC status=enabled  hba=2000000173002c0b on host UNKNOWN
    Port 9 type=FABRIC status=enabled  hba=2000000173002d3e on host cxfssun3
    Port 10 type=FABRIC status=enabled  hba=2000000173003430 on host UNKNOWN
    Port 11 type=FABRIC status=enabled  hba=200900a0b80c13c9 on host UNKNOWN
    Port 12 type=FABRIC status=disabled hba=0000000000000000 on host UNKNOWN
    Port 13 type=FABRIC status=enabled  hba=200d00a0b80c2476 on host UNKNOWN
    Port 14 type=FABRIC status=enabled  hba=1000006069201e5b on host UNKNOWN
    Port 15 type=FABRIC status=enabled  hba=1000006069201e5b on host UNKNOWN
```

A status of `enabled` for an `UNKNOWN` host indicates that the port is connected to a system that is not a node in the cluster. A status of `disabled` for an `UNKNOWN` host indicates that the node has been fenced (disabled), and the port may or may not be connected to a node in the cluster. A status of `enabled` with a specific name host indicates that the port is not fenced and is connected to the specified node in the cluster.

To check current failure action settings, use the `show node` *nodename* command in `cmgr` or use the `cms_failconf` command as follows:

```
/usr/cluster/bin/cms_failconf -q
```

For example, the following output shows that all nodes except `thud` have the system default failure action configuration. The node `thud` has been configured for fencing and resetting.

```
# cms_failconf -q
CMS failure configuration:
        cell[0] whack    Reset Shutdown
        cell[1] thunderbox       Reset Shutdown
        cell[2] thud     Fence Reset
        cell[3] thump    Reset Shutdown
        cell[4] terry    Reset Shutdown
        cell[5] leesa    Reset Shutdown
```

Heartbeat Timeout Status

You can use Performance Co-Pilot or the IRIX `icrash` command to monitor heartbeat timeouts. For example, the following command prints the CXFS kernel messaging statistics:

```
# icrash -e "load cxfs; mtcp_stats"
corefile = /dev/mem, namelist = /unix, outfile = stdout

Please wait............
Loading default Sial macros...........


>> load cxfs
>> mtcp_stats
STATS @ 0xc000000001beebb8
Max delays: discovery 500767 multicast 7486 hb monitor 0
hb generation histogram:(0:0)(1:0)(2:0)(3:0)(4:0)(5:0)
Improperly sized alive mesgs 0 small 0 big 0
Alive mesgs with: invalid cell 0 invalid cluster 0 wrong ipaddr 2
Alive mesgs from: unconfigured cells 100 cells that haven’t discovered us 6000
mtcp_config_cell_set 0x0000000000000007
cell 0:starting sequence # 77 skipped 0
hb stats init @ 15919:(0:1)(1:478301)(2:29733)(3:0)(4:0)
cell 1:starting sequence # 0 skipped 0
hb stats init @ 360049:(0:1)(1:483337)(2:21340)(3:0)(4:0)
cell 2:starting sequence # 0 skipped 0
```

The following fields contain information that is helpful to analyzing heartbeat timing:

- `discovery`: The maximum time in HZ that the discovery thread (that is, the thread that processes incoming heartbeats) has slept. Because nodes generate heartbeats once per second, this thread should never sleep substantially longer than 100 HZ.

  A value much larger than 100 suggests either that it was not receiving heartbeats or that something on the node prevented this thread from processing the heartbeats.

- `multicast`: The thread that generates heartbeats sleeps for 100 HZ after sending the last heartbeat and before starting on the next. This field contains the maximum time in HZ between the start and end of that sleep. A value substantially larger than 100 indicates a problem getting the thread scheduled; for example, when something else on the node is taking all CPU resources.

- `monitor`: The maximum time in HZ for the heartbeat thread to do its sleep and send its heartbeat. That is, it contains the value for `multicast` plus the time it takes to send the heartbeat. If this value is substantially higher than 100 but `multicast` is not, it suggests a problem in acquiring resources to send a heartbeat, such as a memory shortage.

- `gen_hist`: A histogram showing the number of heartbeats generated within each interval. There are 6 buckets tracking each of the first 5 seconds (anything over 5 seconds goes into the 6th bucket).

- `hb_stats`: Histograms for heartbeats received. There is one histogram for each node in the cluster.

- `seq_stats`: Number of consecutive incoming heartbeats that do not have consecutive sequence numbers. There is one field for each node. A nonzero value indicates a lost heartbeat message.
- `overdue`: Time when an overdue heartbeat is noticed. There is one field per node.
- `rescues`: Number of heartbeats from a node that are overdue but CXFS message traffic has been received within the timeout period.
- `alive_small`: Number of times a heartbeat message arrived that was too small, (that is, contained too few bytes).
- `alive_big`: Number of times a heartbeat arrived that was too large.
- `invalid_cell`: Number of heartbeats received from nodes that are not defined in the cluster
- `invalid_cluster`: Number of heartbeats received with the wrong cluster ID
- `wrong_ipaddr`: Number of heartbeats received with an IP address that does not match the IP address configured for the node ID
- `not_configured`: Number of heartbeats received from nodes that are not defined in the cluster
- `unknown`: Number of heartbeats from nodes that have not received the local node’s heartbeat

Chapter 16

Migration from an IRIX Cluster to a Linux Cluster

CXFS supports a running cluster with a single type of operating system for administration nodes: either all IRIX or all Linux. To migrate from an IRIX cluster to a Linux cluster, do the following:

Note: The following procedure assumes that the filesystems in the cluster you want to migrate do not have block sizes greater than the system page size and that they are not real-time filesystems. These types of filesystems are supported on IRIX but not on Linux.

The example in this chapter begins with a cluster named `performance` having a two IRIX server-capable nodes named `rum` and `snake` and a Solaris client-only node named `ray`:

```
rum # clconf_info

Event at [2004-02-13 07:57:17]

Membership since Thu Feb 12 15:15:26 2004
____________ ______ ________ ______ ______
Node         NodeID Status   Age    CellID
____________ ______ ________ ______ ______
snake             1 up            2      1
rum               2 up            2      2
ray               3 up            1      0
____________ ______ ________ ______ ______
1 CXFS FileSystems
/dev/cxvm/V9500 on /cxfs/V9500  enabled  server=(snake)  2 client(s)=(ray,rum)  status=UP
```

1. Unmount the CXFS filesystems. For example, on the IRIX node `rum`:

   ```
   cmgr> admin cxfs_unmount cxfs_filesystem V9500

   cxfs_unmount operation successful
   ```

2. Stop CXFS services on all nodes. For example on the IRIX node `rum`:

   ```
   cmgr> stop cx_services for cluster performance

   CXFS services have been deactivated in cluster performance
   ```

3. Define the administration node with the Linux operating system type. For example on the IRIX node `rum`:

```
cmgr> define node bang
Enter commands, you may enter "done" or "cancel" at any time to exit

Hostname[optional] ? bang
Is this a FailSafe node <true|false> ? false
Is this a CXFS node <true|false> ? true
Operating System <IRIX|Linux32|Linux64|AIX|HPUX|Solaris|MacOSX|Windows> ? Linux64
Node Function <server_admin|client_admin|client_only> ? server_admin
Node ID[optional] ? 64
Partition ID[optional] ? (0)
Do you wish to define failure hierarchy[y/n]:y
Hierarchy option 0 <System|FenceReset|Fence|Reset|Shutdown>[optional] ? Fence
Hierarchy option 1 <System|FenceReset|Fence|Reset|Shutdown>[optional] ?
Reset type <powerCycle|reset|nmi> ? (powerCycle)
Do you wish to define system controller info[y/n]:n
Number of Network Interfaces ? (1)
NIC 1 - IP Address ? bang-p
NIC 1 - Heartbeat HB (use network for heartbeats) <true|false> ? true
NIC 1 - (use network for control messages) <true|false> ? true
NIC 1 - Priority <1,2,...> ? 1

Successfully defined node bang
```

4. Add the Linux administration node to the cluster. For example on the IRIX node `rum`:

```
cmgr> modify cluster performance
Enter commands, you may enter "done" or "cancel" at any time to exit

Is this a FailSafe cluster <true|false> ? (false)
Is this a CXFS cluster <true|false> ? (true)
Cluster Notify Cmd [optional] ?
Cluster Notify Address [optional] ?
Cluster CXFS mode <normal|experimental>[optional] ? (normal)
```

```
Cluster ID ? (1)

Current nodes in cluster performance:
Node - 1: ray
Node - 2: snake
Node - 3: rum


No networks in cluster performance

Add nodes to or remove nodes/networks from cluster performance
Enter "done" when completed or "cancel" to abort

performance ? add node bang
performance ? done
Added node <bang> to cluster <performance>
Successfully modified cluster performance
```

5. Modify the CXFS filesystems to remove the IRIX administration nodes as metadata servers and add the new Linux administration node as metadata server. For example, on the IRIX node `rum`:

   ```
   cmgr> modify cxfs_filesystem V9500

   (Enter "cancel" at any time to abort)

   Device ? (/dev/cxvm/V9500)
   Mount Point ? (/cxfs/V9500)
   Mount Options[optional] ?
   Use Forced Unmount ? <true|false> ? (false)
   Grio Qualififed Bandwidth[optional] ?
   Grio managed filesystem ? <true|false>[optional] ?
   Default Local Status  ? (enabled)

   MODIFY CXFS FILESYSTEM OPTIONS

           0) Modify Server.
           1) Add Server.
           2) Remove Server.
           3) Add Enabled Node.
           4) Remove Enabled Node.
           5) Add Disabled Node.
   ```

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

Enter option:2

Current servers:
CXFS Server 1 - Rank: 0         Node: rum
CXFS Server 2 - Rank: 1         Node: snake

Server Node ? rum

        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:2

Current servers:
CXFS Server 1 - Rank: 1         Node: snake

Server Node ? snake

        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)
```

```
Enter option:1

No current servers

Server Node ? bang
Server Rank ? 1

        0) Modify Server.
        1) Add Server.
        2) Remove Server.
        3) Add Enabled Node.
        4) Remove Enabled Node.
        5) Add Disabled Node.
        6) Remove Disabled Node.
        7) Show Current Information.
        8) Cancel. (Aborts command)
        9) Done. (Exits and runs command)

Enter option:9
Successfully modified cxfs_filesystem V9500
```

After you complete this step, the filesystems would show the following information:

```
cmgr> show cxfs_filesystem V9500

Name: V9500
Device: /dev/cxvm/V9500
Mount Point: /cxfs/V9500
Forced Unmount: false
Global Status: disabled
Default Local Status: enabled

Server Name: bang
       Rank: 1
```

6. Remove the IRIX administration nodes from the cluster. For example, switching to the Linux node `bang`:

```
cmgr> modify cluster performance
Enter commands, you may enter "done" or "cancel" at any time to exit
```

```
Is this a FailSafe cluster <true|false> ? (false)
Is this a CXFS cluster <true|false> ? (true)
Cluster Notify Cmd [optional] ?
Cluster Notify Address [optional] ?
Cluster CXFS mode <normal|experimental>[optional] ? (normal)
Cluster ID ? (1)

Current nodes in cluster performance:
Node - 1: ray
Node - 2: snake
Node - 3: rum
Node - 4: bang


Add nodes to or remove nodes/networks from cluster performance
Enter "done" when completed or "cancel" to abort

performance ? remove node rum
performance ? remove node snake
performance ? done
Successfully modified cluster performance
```

7. Delete the IRIX administration nodes from the pool. For example, from the Linux node `bang`:

   ```
   cmgr> delete node rum
   Deleted node (rum).

   cmgr> delete node snake
   Deleted node (snake).
   ```

8. Start CXFS services for all nodes in the cluster. For example, from the Linux node `bang`:

```
cmgr> start cx_services for cluster performance

CXFS services have been activated in cluster performance
```

9. Mount the CXFS filesystems. For example, from the Linux node `bang`:

   ```
   cmgr> admin cxfs_mount cxfs_filesystem V9500

   cxfs_mount operation successful
   ```

After completing this procedure, the cluster information is as follows:

```
[root@bang root]# clconf_info

Event at [2004-02-13 08:44:18]

Membership since Fri Feb 13 08:44:13 2004
____________ ______ ________ ______ ______
Node         NodeID Status   Age    CellID
____________ ______ ________ ______ ______
ray               3 up            1      0
bang             64 up            1      3
____________ ______ ________ ______ ______
1 CXFS FileSystems
/dev/cxvm/V9500 on /cxfs/V9500  enabled  server=(bang)  1 client(s)=(ray)  status=UP
```

For more information about using the `cmgr` command to perform this procedure, see the following:

- "Unmount a CXFS Filesystem with `cmgr`" on page 267
- "Stop CXFS Services with `cmgr`" on page 254
- "Define a Node with `cmgr`" on page 224
- "Modify a Cluster with `cmgr`" on page 250
- "Modify a CXFS Filesystem with `cmgr`" on page 268
- "Modify a Cluster with `cmgr`" on page 250
- "Delete a Node with `cmgr`" on page 241
- "Start CXFS Services with `cmgr`" on page 254
- "Mount a CXFS Filesystem with `cmgr`" on page 266

For more information about using the GUI, see the following:

- "Unmount CXFS Filesystems with the GUI" on page 208
- "Stop CXFS Services with the GUI" on page 191
- "Define a Node with the GUI" on page 172
- "Add or Remove Nodes in the Cluster with the GUI" on page 181
- "Modify a CXFS Filesystem with the GUI" on page 207
- "Add or Remove Nodes in the Cluster with the GUI" on page 181
- "Delete a Node with the GUI" on page 186
- "Start CXFS Services with the GUI" on page 191
- "Mount CXFS Filesystems with the GUI" on page 208

Chapter 17

Troubleshooting

Configuring and administering a CXFS cluster can be a complex task. In general, most problems can be solved by rebooting a node. However, the topics in this chapter may help you avoid rebooting:

- "Troubleshooting Strategy"
- "Common Problems" on page 404
- "Understanding Error Messages" on page 412
- "Corrective Actions" on page 434
- "Reporting Problems to SGI" on page 441

You must perform administrative tasks with `cmgr` from a node that has the `cluster_admin` software package installed; you must connect the GUI to such a node. See the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage* for additional troubleshooting information.

Troubleshooting Strategy

To troubleshoot CXFS problems, do the following:

- "Know the Tools"
- "Avoid Problems" on page 391
- "Identify the Cluster Status" on page 400
- "Determine If a Node Is Fenced" on page 401
- "Locate the Problem" on page 402
- "Redirect Switch Logs" on page 403

Know the Tools

This section provides an **overview** of the tools required to troubleshoot CXFS:

Caution: Many of the commands listed are beyond the scope of this book and are provided here for quick reference only. See the other guides and man pages referenced for complete information before using these commands.

- "Physical Storage Tools" on page 382
- "Cluster Configuration Tools" on page 383
- "Cluster Control Tools" on page 384
- "Networking Tools" on page 385
- "Cluster/Node Status Tools" on page 386
- "Performance Monitoring Tools" on page 387
- "Kernel Status Tools" on page 387
- "Log Files" on page 389
- "Gather Cluster Configuration with `cxfsdump`" on page 389

Physical Storage Tools

Understand the following physical storage tools:

- To display the hardware inventory:
  - IRIX:

    ```
    irix# /sbin/hinv
    ```

  - Linux (assuming the `sgi-misc` RPM is installed):

    ```
    [root@linux root]# /usr/bin/hinv
    [root@linux root]# /usr/bin/topology
    ```

  If the output is not what you expected, do a probe for devices and perform a SCSI bus reset, using the following commands:

  - IRIX:

    ```
    irix# /usr/sbin/scsiha -pr bus_number
    ```

  - Linux:

    ```
    root@linux root]# /bin/xscsiha -pr  /dev/xscsi/busnumber/bus
    ```

- To configure I/O devices on an IRIX node, use the following command:

  ```
  irix# /sbin/ioconfig -f /hw
  ```

- To show the physical volumes, use the `xvm` command:

  ```
  # /sbin/xvm show -v phys/
  ```

  See the *XVM Volume Manager Administrator's Guide*.

Cluster Configuration Tools

Understand the following cluster configuration tools:

- To configure XVM volumes, use the `xvm` command:

  ```
  # /sbin/xvm
  ```

  See the *XVM Volume Manager Administrator's Guide*.

- To configure CXFS nodes and cluster, use either the GUI or the `cmgr` command:

  - The GUI:

    ```
    # /usr/sbin/cxfsmgr
    ```

    See "GUI Features" on page 156 and Chapter 9, "Reference to GUI Tasks for CXFS" on page 149.

  - The `cmgr` command line with prompting:

    ```
    # /usr/cluster/bin/cmgr -p
    ```

    See "`cmgr` Overview" on page 218, and Chapter 10, "Reference to `cmgr` Tasks for CXFS" on page 217.

- To reinitialize the database, use the `cdbreinit` command:

  ```
  # /usr/cluster/bin/cdbreinit
  ```

See "Recreating the Cluster Database" on page 440.

- To check the cluster configuration, use the following command from a server-capable administration node in the cluster:

  ```
  # /usr/cluster/bin/cxfs-config -all -check
  ```

  SGI recommends that you run this command after any significant configuration change or whenever problems occur. For more information, see "Checking the Cluster Configuration with `cxfs-config`" on page 342.

Cluster Control Tools

Understand the cluster control tools:

- "Cluster Administration Daemons" on page 24
- "CXFS Client Daemon" on page 26

  These commands are useful if you know that filesystems are available but are not indicated as such by the cluster status, or if cluster quorum is lost. However, note that `/etc/init.d/cxfs stop` and `/etc/init.d/cxfs_client stop` will cause CXFS to completely shut down on the local node.

  See the following:

  – "Cluster Database Membership Quorum Stability" on page 392

  – "Restarting CXFS Services" on page 434

  – "Clearing the Cluster Database" on page 435

  – "Stopping and Restarting Cluster Administration Daemons" on page 439

- "CXFS Services" on page 25

  Running this command on the metadata server will cause its filesystems to be recovered by another potential metadata server. See "Cluster Services Tasks with `cmgr`" on page 254, and "Cluster Services Tasks with the GUI" on page 191.

  Note: Relocation and recovery are supported only when using standby nodes. Relocation is disabled by default.

- To allow and revoke CXFS kernel membership on the local node, forcing recovery of the metadata server for the local node, use the GUI or the following `cmgr` commands:

  ```
  cmgr> admin cxfs_start
  cmgr> admin cxfs_stop
  ```

  Wait until recovery is complete before issuing a subsequent `admin cxfs_start`. The local node cannot rejoin the CXFS kernel membership until its recovery is complete.

  See the following:

  – "Revoke Membership of the Local Node with the GUI" on page 195

  – "Allow Membership of the Local Node with the GUI" on page 195

  – "Revoke Membership of the Local Node with `cmgr`" on page 259

  – "Allow Membership of the Local Node with `cmgr`" on page 259

Networking Tools

Understand the following networking tools:

- To send packets to network hosts:

  – IRIX:

  ```
  irix# /usr/etc/ping
  ```

  – Linux:

  ```
  [root@linux root]# /bin/ping
  ```

- To show network status:

  – IRIX:

  ```
  irix# /usr/etc/netstat
  ```

  – Linux:

  ```
  [root@linux root]# /bin/netstat
  ```

Cluster/Node Status Tools

Understand the following cluster/node status tools:

- To show which cluster daemons are running:

  ```
  # ps -ef | grep cluster
  ```

  See "Verify that the Cluster Daemons are Running" on page 126.

- To see cluster and filesystem status, use one of the following:

  – GUI:

  ```
  # /usr/sbin/cxfsmgr
  ```

  See "Display a Cluster with the GUI" on page 190.

  – `clconf_info` command:

  ```
  # /usr/cluster/bin/clconf_info
  ```

  – `cxfs_info` command on an IRIX or Linux client-only node:

  ```
  # /usr/cluster/bin/cxfs_info
  ```

- To see the mounted filesystems:

  – IRIX:

  ```
  irix# /sbin/mount
  irix# /usr/sbin/df
  ```

  – Linux:

  ```
  [root@linux root]# /bin/mount
  [root@linux root]# /bin/df
  ```

  You can also use the `df` command to report the number of free disk blocks

- To show volumes:

  ```
  # /sbin/xvm show vol/
  ```

  See the *XVM Volume Manager Administrator's Guide*.

Performance Monitoring Tools

Understand the following performance monitoring tools:

- To monitor system activity:

  ```
  # /usr/bin/sar
  ```

- To monitor file system buffer cache activity on IRIX nodes:

  ```
  irix# /usr/sbin/bufview
  ```

  Note: Do not use `bufview` interactively on a busy IRIX node; run it in batch mode.

- To monitor operating system activity data on an IRIX node::

  ```
  irix# /usr/sbin/osview
  ```

- To monitor the statistics for an XVM volume, use the `xvm` command:

  ```
  # /sbin/xvm change stat on {concatname|stripename|physname}
  ```

  See the *XVM Volume Manager Administrator's Guide*.

- To monitor system performance, use Performance Co-Pilot. See the *Performance Co-Pilot for IA-64 Linux User's and Administrator's Guide*, *Performance Co-Pilot for IRIX Advanced User's and Administrator's Guide*, the *Performance Co-Pilot Programmer's Guide*, and the `pmie` and `pmieconf` man pages.

Kernel Status Tools

Understand the following kernel status tools (this may require help from SGI service personnel):

- To determine IRIX kernel status, use the `icrash` commands:

  ```
  # /usr/bin/icrash
  ```

  - `cfs` to list CXFS commands
  - `dcvn` to list client vnodes
  - `dsvn` to list server vnodes
  - `mesglist` to trace messages to the receiver

  - `sinfo` to show clients/servers and filesystems
  - `sthread | grep cmsd` to determine the CXFS kernel membership state. You may see the following in the output:
    - `cms_dead()` indicates that the node is dead
    - `cms_follower()` indicates that the node is waiting for another node to create the CXFS kernel membership (the leader)
    - `cms_leader()` indicates that the node is leading the CXFS kernel membership creation
    - `cms_declare_membership()` indicates that the node is ready to declare the CXFS kernel membership but is waiting on resets
    - `cms_nascent()` indicates that the node has not joined the cluster since starting
    - `cms_shutdown()` indicates that the node is shutting down and is not in the CXFS kernel membership
    - `cms_stable()` indicates that the CXFS kernel membership is formed and stable
  - `tcp_channels` to determine the status of the connection with other nodes
  - `-t -a -w` *filename* to trace for CXFS
  - `-t` *cms_thread* to trace one of the above threads

- To determine Linux kernel status, use the KDB built-in kernel debugger.

  When kdb is enabled, a system panic will cause the debugger to be invoked and the keyboard LEDs will blink. The `kdb` prompt will display basic information. To obtain a stack trace, enter the `bt` command at the `kdb` prompt:

  ```
  kdb> bt
  ```

  To get a list of current processes, enter the following:

  ```
  kdb> ps
  ```

  To backtrace a particular process, enter the following, where *PID* is the process ID:

  ```
  kdb> btp PID
  ```

To exit the debugger, enter the following:

```
kdb> go
```

If the system will be run in graphical mode with `kdb` enabled, SGI highly recommends that you use kdb on a serial console so that the kdb prompt can be seen.

- To invoke internal kernel routines that provide useful debugging information, use the `idbg` command:

  ```
  # /usr/sbin/idbg
  ```

Log Files

Understand the log files discussed in "Status in Log Files" on page 360.

Gather Cluster Configuration with `cxfsdump`

Before reporting a problem to SGI, you should use the `cxfsdump` command to gather configuration information about the CXFS cluster, such as network interfaces, CXFS registry information, I/O, and cluster database contents. This will allow SGI support to solve the problem more quickly.

Note: In cluster mode (the default), the `cxfsdump` command requires `rsh`/`ssh` and `rcp`/`scp` access across all nodes in the cluster. You can use the `-secure` option to use secure remote connections.

You should run `cxfsdump` from a CXFS administration node in the cluster:

```
# /usr/cluster/bin/cxfsdump
```

The output will be placed in a file in the directory `/var/cluster/cxfsdump-data` directory on the CXFS administration node on which the `cxfsdump` command was run. The `cxfsdump` command will report the name and location of the file when it is finished.

To gather information about just the local node, use the `cxfsdump -local` option.

The `cxfsdump /?` command displays a help message on Windows nodes. The `cxfsdump -help` command displays a help message on other nodes.

Following is an example of gathering information for the entire cluster from an IRIX node:

```
adminnode# cxfsdump

Detecting cluster configuration

 Executing CXFSDUMP on CLUSTER testcluster NODE o200a
Gathering cluster information...
Determining OS level......
Getting versions info....
Obtaining CXFS database...
Checking for tie-breakers etc...
Obtaining hardware inventory...
Grabbing /etc/hosts.....
Grabbing /etc/resolv.conf...
Grabbing /ets/nsswitch.conf...
Obtaining physvol information using XVM...
ioctl() to xvm api node failed: Invalid argument
Could not get xvm subsystem info:  xvmlib_execute_ioctl: system call failed.
Obtaining Volume topology information using XVM...
ioctl() to xvm api node failed: Invalid argument
Could not get xvm subsystem info:  xvmlib_execute_ioctl: system call failed.
Copying failover configuration and scsifo paths ...
Gathering network information...
Checking for any installed Patches..
Monitoring file system buffer cache for 3 minutes...
Running Systune ...
Obtaining modified system tunable parameters...
Creating ICRASH CMD file...
Executing ICRASH commands...
Copying CXFS logs...
Copying /var/cluster/ha/log/cad_log...
Copying /var/cluster/ha/log/clconfd_o200a...
Copying /var/cluster/ha/log/cli_o200a...
Copying /var/cluster/ha/log/cmond_log...
Copying /var/cluster/ha/log/crsd_o200a...
Copying /var/cluster/ha/log/fs2d_log...
Copying /var/cluster/ha/log/fs2d_log.old...
Copying SYSLOG...
Distributing /usr/cluster/bin/cxfsdump.pl to node o200c ...
Distributing /usr/cluster/bin/cxfsdump.pl to node o200b ...
Creating the output directory : /var/cluster/cxfsdump-data
Gathering node information for the cluster testcluster ...
```

```
Running RSH to node o200c...
Running RSH to node o200b...
Waiting for other cluster nodes to gather data...
FINAL CXFSDUMP OUTPUT IN /var/cluster/cxfsdump-data/testcluster_cxfsdump20020903.tar.gz
```

On Windows systems, `cxfsdump` creates a directory called `cxfsdump-data` in the same directory where the the `passwd` file is kept. The `cxfsdump` command will report the location where the data is stored when it is complete. For example:

`FINAL CXFSDUMP output in` *output_filename*

Avoid Problems

This section covers the following:

- "Proper Start Up" on page 392
- "Eliminate a Residual Cluster" on page 392
- "Cluster Database Membership Quorum Stability" on page 392
- "Consistency in Configuration" on page 393
- "Define Node Function Appropriately" on page 393
- "GUI Use" on page 394
- "Log File Names and Sizes" on page 394
- "IRIX: Netscape and the Brocade Switch GUI" on page 394
- "Performance Problems with Unwritten Extent Tracking and Exclusive Write Tokens" on page 394
- "Avoid Excessive Filesystem Activity Caused by the `crontab` File" on page 396
- "Use System Capacity Wisely" on page 396
- "Reboot Before Changing Node ID or Cluster ID" on page 397
- "Remove Unused Nodes" on page 397
- "Restart CXFS after an Administrative CXFS Stop" on page 397

- "Remove Reset Lines" on page 398
- "Appropriate Use of `xfs_repair`" on page 398

Proper Start Up

Ensure that you follow the instructions in "Preliminary Cluster Configuration Steps" on page 125, before configuring the cluster.

Eliminate a Residual Cluster

Before you start configuring another new cluster, make sure no nodes are still in a CXFS membership from a previous cluster. Enter the following to check for a `cmsd` kernel thread:

- IRIX:

  ```
  irix# icrash -e 'sthread | grep cmsd'
  ```

- Linux:

  ```
  [root@linux root]# ps -ef | grep cmsd
  ```

If the output shows a `cmsd` kernel thread, perform a forced CXFS shutdown by entering the following:

```
# /usr/cluster/bin/cmgr -p
cmgr> admin cxfs_stop
```

Then check for a `cmsd` kernel thread again.

After waiting a few moments, if the `cmsd` kernel thread still exists, you must reboot the machine or leave it out of the new cluster definition. It will not be able to join a new cluster in this state and it may prevent the rest of the cluster from forming a new CXFS membership.

Cluster Database Membership Quorum Stability

The cluster database membership quorum must remain stable during the configuration process. If possible, use multiple windows to display the `fs2d_log` file for each CXFS administration node while performing configuration tasks. Enter the following:

```
# tail -f /var/cluster/ha/log/fs2d_log
```

Check the member count when it prints new quorums. Under normal circumstances, it should print a few messages when adding or deleting nodes, but it should stop within a few seconds after a new quorum is adopted.

If not enough machines respond, there will not be a quorum. In this case, the database will not be propagated.

If you detect cluster database membership quorum problems, fix them before making other changes to the database. Try restarting the cluster administration daemons on the node that does not have the correct cluster database membership quorum, or on all nodes at the same time.

Enter the following on administration nodes:

- IRIX:

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

- Linux:

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

Note: You could also use the `restart` option to stop and start.

Please provide the `fs2d` log files when reporting a cluster database membership quorum problem.

Consistency in Configuration

Be consistent in configuration files for nodes across the pool, and when configuring networks. Use the same names in the same order. See "Configuring System Files" on page 95.

Define Node Function Appropriately

Use the appropriate node function definition:

- Use an odd number of server-capable administration nodes. If you have an even number of server-capable administration nodes, define a CXFS tiebreaker node. SGI recommends making a client administration or client-only node the tiebreaker.

- Make unstable nodes CXFS client-only nodes.

GUI Use

The GUI provides a convenient display of a cluster and its components through the view area. You should use it to see your progress and to avoid adding or removing nodes too quickly. After defining a node, you should wait for it to appear in the view area before adding another node. After defining a cluster, you should wait for it to appear before you add nodes to it. If you make changes too quickly, errors can occur.

For more information, see "Starting the GUI" on page 150.

When running the GUI on IRIX, do not move to another IRIX desktop while GUI action is taking place; this can cause the GUI to crash.

Log File Names and Sizes

You should not change the names of the log files. If you change the names of the log files, errors can occur.

Periodically, you should rotate log files to avoid filling your disk space; see "Log File Management" on page 307. If you are having problems with disk space, you may want to choose a less verbose log level; see "Configure Log Groups with the GUI" on page 194, or "Configure Log Groups with `cmgr`" on page 257.

IRIX: Netscape and the Brocade Switch GUI

When accessing the Brocade Web Tools V2.0 through Netscape on an IRIX node, you must first enter one of the following before starting Netscape:

- For `sh` or `ksh` shells:

  ```
  $ NOJIT=1; export NOJIT
  ```

- For `csh` shell:

  ```
  % setenv NOJIT 1
  ```

If this is not done, Netscape will crash with a core dump.

Performance Problems with Unwritten Extent Tracking and Exclusive Write Tokens

This section discusses performance problems with unwritten extent tracking and exclusive write tokens.

Unwritten Extent Tracking

When you define a filesystem, you can specify whether unwritten extent tracking is on (`unwritten=1`) or off (`unwritten=0`); it is on by default.

In most cases, the use of unwritten extent tracking does not affect performance and you should use the default to provide better security.

However, unwritten extent tracking can affect performance when **both** of the following are true:

- A file has been preallocated
- These preallocated extents are written for the first time with records smaller than 4 MB

For optimal performance with CXFS when **both** of these conditions are true, it may be necessary to build filesystems with `unwritten=0` (off).

Note: There are security issues with using `unwritten=0`. For more information, see the *IRIX Admin: Disks and Filesystems*.

Exclusive Write Tokens

For proper performance, CXFS should not obtain exclusive write tokens. Therefore, use the following guidelines:

- Preallocate the file.
- Set the size of the file to the maximum size and do not allow it to be changed, such as through truncation.
- Do not append to the file. (That is, `O_APPEND` is not true on the open.)
- Do not mark an extent as `written`.
- Do not allow the application to do continual `preallocation` calls.

If the guidelines are followed and there are still performance problems, you may find useful information by running the `icrash stat` command before, halfway through, and after running the MPI job. For more information, see the `icrash` man page.

Avoid Excessive Filesystem Activity Caused by the `crontab` File

The default `root crontab` file contains the following entries (line breaks inserted here for readability):

```
0 5 * * *  find / -local -type f ’(’ -name core -o -name dead.letter ’)’ -atime +7
-mtime +7 -exec rm -f ’{}’ ’;’

 0 3 * * 0 if test -x /usr/etc/fsr; then (cd /usr/tmp; /usr/etc/fsr) fi
```

The first entry executes a `find` command that looks for and removes all files with the name `core` or `dead.letter` that have not been accessed in the past seven days.

The second entry executes an `fsr` command that improves the organization of mounted filesystems.

The `find` command will be run nightly on all local filesystems. Because CXFS filesystems are considered as local on all nodes in the cluster, the nodes may generate excessive filesystem activity if they try to access the same filesystems simultaneously. Therefore, you may wish use the following sequence to disable or modify the `find crontab` entries on all the CXFS administration nodes except for one:

1. Log in as `root`.
2. Define your editor of choice, such as `vi`:

   ```
   # setenv EDITOR vi
   ```

3. Edit the `crontab` file:

   ```
   # crontab -e
   ```

4. Comment out or delete the `find` line.

The `fsr` command can only be run on the metadata server, so it is not harmful to leave it in the `crontab` file for CXFS clients, but it will not be executed.

Use System Capacity Wisely

To avoid a loss of connectivity between the metadata server and the CXFS clients, do not oversubscribe the metadata server or the private network connecting the nodes in the cluster. Avoid unnecessary metadata traffic.

If the amount of free memory is insufficient, a node may experience delays in heartbeating and as a result will be kicked out of the CXFS membership. To observe the amount of free memory in your system, use the `osview` tool.

See also "Out of Logical Swap Space" on page 416.

Reboot Before Changing Node ID or Cluster ID

If you want redefine a node ID or the cluster ID, you must first reboot. The problem is that the kernel still has the old values, which prohibits a CXFS membership from forming. However, if you perform a reboot first, it will clear the original values and you can then redefine the node or cluster ID.

Therefore, if you use `cdbreinit` on a node to recreate the cluster database, you must reboot it before changing the node IDs or the cluster ID. See "Recreating the Cluster Database" on page 440.

Remove Unused Nodes

If a node is going to be down for a while, remove it from the cluster and the pool to avoid cluster database membership and CXFS membership quorum problems. See the following sections:

- "Modify a Cluster Definition with the GUI" on page 189
- "Modify a Cluster with `cmgr`" on page 250
- "Delete a Node with `cmgr`" on page 241

Restart CXFS after an Administrative CXFS Stop

If you perform an administrative CXFS stop (forced CXFS shutdown) on a node, you must perform an administrative CXFS start on that node before it can return to the cluster. If you do this while the database still shows that the node is in a cluster and is activated, the node will restart the CXFS membership daemon. Following a forced CXFS shutdown, the node can be prevented from restarting the CXFS membership daemon when CXFS is restarted by stopping CXFS services. (A forced CXFS shutdown alone does not stop CXFS services. A forced CXFS shutdown stops only the kernel membership daemon. Stopping CXFS services disables the node in the cluster database.)

For example, enter the following on the local node you wish to start:

```
# /usr/cluster/bin/cmgr -p
cmgr> stop cx_services on node localnode
cmgr> admin cxfs_start
```

See also "Forced CXFS Shutdown: Revoke Membership of Local Node" on page 304.

Remove Reset Lines

When reset is enabled, CXFS requires a `reset successful` message before it moves the metadata server. Therefore, if you have the reset capability enabled and you must remove the reset lines for some reason, you must also disable the reset capability. See "Modify a Node Definition with the GUI" on page 182, or "Modify a Node with `cmgr`" on page 234.

Note: System reset configuration is recommended for all potential metadata servers. See "Cluster Environment" on page 9.

Appropriate Use of `xfs_repair`

CXFS filesystems are really clustered XFS filesystems; therefore, in case of a file system corruption, you can use the `xfs_check` and `xfs_repair` commands. However, you must first ensure that you have an actual case of data corruption and retain valuable metadata information by replaying the XFS logs before running `xfs_repair`.

Caution: If you run `xfs_repair` without first replaying the XFS logs, you may introduce data corruption.

You should only run `xfs_repair` in case of an actual filesystem corruption; forced filesystem shutdown messages **do not** necessarily imply that `xfs_repair` should be run. Following is an example of a message that does indicate an XFS file corruption:

```
XFS read error in file system metadata block 106412416
```

When a filesystem is forcibly shut down, the log is not empty — it contains valuable metadata. You must replay it by mounting the filesystem. The log is only empty if the filesystem is unmounted cleanly (that is, not a forced CXFS shutdown, not a crash). You can use the following command line to see an example of the transactions captured in the log file:

```
# xfs_logprint -t device
```

If you run `xfs_repair` before mounting the filesystem, `xfs_repair` will delete all of this valuable metadata.

You should run `xfs_ncheck` and capture the output to a file before running `xfs_repair`. If running `xfs_repair` results in files being placed in the `lost+found` directory, the saved output from `xfs_ncheck` may help you to identify the original names of the files.

If you think you have a filesystem with real corruption, do the following:

1. Mount the device in order to replay the log:

   ```
   # mount device  any_mount_point
   ```

2. Unmount the filesystem:

   ```
   # unmount device
   ```

3. Check the filesystem:

   ```
   # xfs_check device
   ```

4. View the repairs that could be made, using `xfs_repair` in no-modify mode:

   ```
   # xfs_repair -n device
   ```

5. Capture filesystem file name and inode pairs:

   ```
   # xfs_ncheck device > xfs_ncheck.out
   ```

6. If you are certain that the repairs are appropriate, complete them:

   ```
   # xfs_repair device
   ```

For more information, see the *IRIX Admin: Disks and Filesystems*.

Identify the Cluster Status

When you encounter a problem, identify the cluster status by answering the following questions:

- Are the cluster daemons running? See "Verify that the Cluster Daemons are Running" on page 126.
- Is the cluster state consistent on each node? Run the `clconf_info` command on each CXFS administration node and compare.
- Which nodes are in the CXFS kernel membership? See "Check Cluster Status with `cmgr`" on page 364, and the following files:
  - IRIX: `/var/adm/SYSLOG`
  - Linux: `/var/log/messages`
- Which nodes are in the cluster database (`fs2d`) membership? See the `/var/cluster/ha/log/fs2d_log` files on each CXFS administration node.
- Is the database consistent on all CXFS administration nodes? Determine this logging in to each administration node and examining the `/var/cluster/ha/log/fs2d_log` file and database checksum.
- Log onto the various CXFS client nodes or use the GUI view area display with details showing to answer the following:
  - Are the devices available on all nodes? Use the following:
    - The `xvm` command to show the physical volumes:

      ```
      xvm:cluster> show -v phys/
      ```
    - Is the client-only node in the cluster? Use the `cxfs_info` command.
    - List the contents of the `/dev/cxvm` directory with the `ls` command:

      ```
      # ls /dev/cxvm
      ```
    - Use the `hinv` command to display the hardware inventory. See "Physical Storage Tools" on page 382.
  - Are the filesystems mounted on all nodes? Use `mount` and `clconf_info` commands.

- Which node is the metadata server for each filesystem? Use the `clconf_info` command.

  On the metadata server, use the `clconf_info` command.

- Is the metadata server in the process of recovery? Use the IRIX `icrash` command to search for messages and look at the following files:
  - IRIX: `/var/adm/SYSLOG`
  - Linux: `/var/log/messages`

  See "Kernel Status Tools" on page 387. Messages such as the following indicate that recovery status:

  - In process:

```
Mar 13 11:31:02 1A:p2 unix: ALERT: CXFS Recovery: Cell 1: Client Cell 0 Died, Recovering </scratch/p9/local>
```

  - Completed:

```
Mar 13 11:31:04 5A:p2 unix: NOTICE: Signaling end of recovery cell 1
```

- Are there any long running (>20 seconds) kernel messages? Use the `icrash mesglist` command to examine the situation. For example:

```
>> mesglist
Cell:7
THREAD ADDR         MSG ID TYPE CELL MESSAGE                           Time(Secs)
================== ======= ==== ==== ================================ ==========
0xa8000000d60a4800  5db537  Rcv    0                   I_dcvn_recall          0
0xa8000000d60a4800  5db541  Snt    0                 I_dsvn_notfound          0
0xa80000188fc51800  3b9b4f  Snt    0            I_dsxvn_inode_update  17:48:58
```

- If filesystems are not mounting, do they appear online in XVM? You can use the following `xvm` command:

```
xvm:cluster> show vol/*
```

Determine If a Node Is Fenced

To determine if a node is fenced, log in to a CXFS administration node and use the `hafence`(1M) command. For more details, see the *CXFS Administration Guide for SGI InfiniteStorage*.

The following messages are logged when fencing changes:

```
Raising fence on cell cellID (nodename)

Lowering fence on cell cellID (nodename)
```

Locate the Problem

To locate the problem, do the following:

- Examine the log files (see "Log Files" on page 389):
  - Search for errors in all log files. See "Status in Log Files" on page 360. Examine all messages within the timeframe in question.
  - Trace errors to the source. Try to find an event that triggered the error.
- Use the IRIX `icrash` commands. See "Kernel Status Tools" on page 387.
- Use detailed information from the view area in the GUI to drill down to specific configuration information.
- Run the **Test Connectivity** task in the GUI. See "Test Node Connectivity with the GUI" on page 187.
- Determine how the nodes of the cluster see the current CXFS kernel membership by entering the following command on each CXFS administration node:

  ```
  # /usr/cluster/bin/clconf_info
  ```

  For more information, see "Check Cluster Status with `clconf_info`" on page 362.
- Check the following file on each CXFS administration node to make sure the CXFS filesystems have been successfully mounted or unmounted:
  - IRIX: `/var/adm/SYSLOG`
  - Linux: `/var/log/messages`

  If a mount/unmount fails, the error will be logged and the operation will be retried after a short delay.

- Use the `sar` system activity reporter to show the disks that are active. For example, the following example for IRIX will show the disks that are active, put the disk name at the end of the line, and poll every second for 10 seconds:

  ```
  irix# sar -DF 1 10
  ```

  For more information, see the `sar` man page.

- Use the IRIX `bufview` filesystem buffer cache activity monitor to view the buffers that are in use. Within `bufview`, you can use the `help` subcommand to learn about available subcommands, such as the `f` subcommand to limit the display to only those with the specified flag. For example, to display the in-use (busy) buffers:

  ```
  # bufview
  f
  Buffer flags to display bsy
  ```

  For more information, see the `bufview` man page.

- Use the IRIX `icrash` command. For more information, see the `icrash` man page.
- Get a dump of the cluster database. You can extract such a dump with the following command:

  ```
  # /usr/cluster/bin/cdbutil -c 'gettree #' > dumpfile
  ```

Redirect Switch Logs

Brocade switch problems can cause CXFS to behave abnormally. For easier troubleshooting, use the `syslogdipadd` function on the switch to redirect its `syslogd` information to up to six potential metadata servers in the cluster. SGI recommends logging to at least two potential metadata servers on which you troubleshoot issues and look for error messages. The `syslogd` information is the same as that given by `errshow` command on the switch.

For example, on each switch, define the metadata server nodes MDS1 and MDS2 to which the switch can redirect its `syslogd` output:

```
switch:admin > syslogdipadd ipaddress_MDS1
switch:admin > syslogdipadd ipaddress_MDS2
```

The entries from the switch can be sorted because they are prefixed by the switch name, which is standard `syslogd` behavior.

Common Problems

The following are common problems and solutions:

- "Node is Permanently Fenced" on page 404
- "Cannot Access Filesystem" on page 405
- "GUI Will Not Run" on page 405
- "Log Files Consume Too Much Disk Space" on page 405
- "Unable to Define a Node" on page 406
- "System is Hung" on page 406
- "Node is Detected but Never Joins Membership" on page 406
- "Cell ID Count and “Membership Delivered” Messages" on page 406
- "You Cannot Log In" on page 407
- "I/O Error in Filesystem" on page 407
- "Cannot Mount Filesystems" on page 408
- "GUI Displays Invalid Filesystems" on page 408
- "Multiple `client_timeout` Values" on page 408
- "No HBA WWPNs are Detected" on page 409
- "XFS Internal Errors in System Log File" on page 411
- "Multiple Ethernet Interfaces on Altix Systems" on page 411
- "Clients Unable to Remount Filesystems" on page 412

Node is Permanently Fenced

If you are unable to raise the fence on a node, it may be that the switch ports are unable to determine the WWPN. See "Hardware Changes and I/O Fencing" on page 321.

Cannot Access Filesystem

If you cannot access a filesystem, check the following:

- Is the filesystem enabled? Check the GUI and `clconf_info` command.
- Were there mount errors?

GUI Will Not Run

If the GUI will not run, check the following:

- Is the license properly installed? See the following:
  - "Verify the License" on page 126
  - "License Error" on page 418
- Are the cluster daemons running? See "Verify that the Cluster Daemons are Running" on page 126.
- Are the `tcpmux` and `tcpmux/sgi_sysadm` services enabled in the following files?
  - IRIX: `/etc/inetd.conf`
  - Linux: `/etc/xinetd.d/tcpmux` and `/etc/tcpmux.conf`
- Are the `inetd` or `tcp` wrappers interfering? This may be indicated by `connection refused` or `login failed` messages.
- Are you connecting to a CXFS administration node? The `cxfsmgr` command can only be executed on a CXFS administration node. The GUI may be run from another system via the Web if you connect the GUI to a CXFS administration node.

Log Files Consume Too Much Disk Space

If the log files are consuming too much disk space, you should rotate them; see "Log File Management" on page 307. You may also want to consider choosing a less-verbose log level; see the following:

- "`cad.options` on CXFS Administration Nodes" on page 96
- "`fs2d.options` on CXFS Administration Nodes" on page 98
- "Configure Log Groups with the GUI" on page 194

Unable to Define a Node

If you are unable to define a node, it may be that there are hostname resolution problems. See "Hostname Resolution and Network Configuration Rules" on page 57.

System is Hung

The following may cause the system to hang:

- Overrun disk drives.
- Heartbeat was lost. In this case, you will see a message that mentions `withdrawl of node`.
- As a last resort, do a non-maskable interrupt (NMI) of the system and contact SGI. (The NMI tells the kernel to panic the node so that an image of memory is saved and can be analyzed later.) For more information, see the owner's guide for the node.

  Make the following files available:

  - System log file:
    - IRIX: `/var/adm/SYSLOG`
    - Linux: `/var/log/messages`
  - IRIX `vmcore.#.comp`
  - IRIX `unix.#`

Node is Detected but Never Joins Membership

If a node is detected in the system log file but it never receives a `Membership delivered` message, it is likely that there is a network problem.

See "Configuring System Files" on page 95.

Cell ID Count and “Membership Delivered” Messages

The `Membership delivered` messages in the system log file file include a list of cell IDs for nodes that are members in the new CXFS membership.

Following each cell ID is a number, the *membership version*, that indicates the number of times the membership has changed since the node joined the membership.

If the `Membership delivered` messages are appearing frequently in the system log file, it may indicate a network problem:

- Nodes that are stable and remain in the membership will have a large membership version number.
- Nodes that are having problems will be missing from the messages or have a small membership version number.

See "Configuring System Files" on page 95.

You Cannot Log In

If you cannot log in to a CXFS administration node, you can use one of the following commands, assuming the node you are on is listed in the other nodes' `.rhosts` files:

```
# rsh hostname ksh -i
# rsh hostname csh -i
```

I/O Error in Filesystem

The following message indicates a problem (output lines wrapped here for readability):

```
ALERT: I/O error in filesystem ("/mnt") metadata dev 0xbd block 0x41df03 ("xlog_iodone")
ALERT:     b_error 0 b_bcount 32768 b_resid 0
NOTICE: xfs_force_shutdown(/mnt,0x2) called from line 966 of file ../fs/xfs/xfs_log.c.
  Return address = 0xc0000000008626e8
ALERT: I/O Error Detected.  Shutting down filesystem: /mnt
ALERT: Please umount the filesystem, and rectify the problem(s)
```

You can fix this problem using `xfs_repair` only if there is no metadata in the XFS log. See "Appropriate Use of `xfs_repair`" on page 398, for the appropriate procedure.

I/O errors can also appear if the node is unable to access the storage. This can happen for several reasons:

- The node has been physically disconnected from the SAN

- A filesystem shutdown due to loss of membership
- A filesystem shutdown due to lost of the metadata server
- The node has been fenced out of the SAN

Cannot Mount Filesystems

If you are unable to raise the fence on a node, it may be that the switch ports are unable to determine the WWPN. See "Hardware Changes and I/O Fencing" on page 321.

If you have defined filesystems and then rename your cluster (by deleting the old cluster and defining a new cluster), CXFS will not be able to mount the existing filesystems. This happens because the clustered XVM volume on which your CXFS filesystem resides is not accessible to the new cluster, and the volumes are therefore considered as foreign.

In order to mount the filesystem on the new cluster, you must use the XVM `steal` command to bring the clustered XVM volume into the domain of the new cluster. For more information, see the *XVM Volume Manager Administrator's Guide*.

GUI Displays Invalid Filesystems

If you create new slices on a previously sliced disk that have the same starting blocks as slices already existing on the disk, and if the old slices had filesystems, then the GUI will display those old filesystems even though they may not be valid.

Multiple `client_timeout` Values

A `client_timeout` value is set by the `clconfd` and `cxfs_client` daemons. The value depends on the order in which filesystems are mounted on the various nodes. The value adapts to help ensure that all filesystems get mounted in a timely manner. The value has no effect on the filesystem operation after it is mounted.

The value for `client_timeout` may differ among nodes, and therefore having multiple values is not really a problem.

The `retry` value is forced to be 0 and you cannot change it.

Caution: You should not attempt to change the `client_timeout` value. Improperly setting the values for client_timeout and `retry` could cause the `mount` command to keep waiting for a server and could delay the availability of the CXFS filesystems.

No HBA WWPNs are Detected

On most platforms, the `cxfs_client` software automatically detects the world wide port names (WWPNs) of any supported host bus adapters (HBAs) in the system that are connected to a switch that is configured in the cluster database. These HBAs will then be available for fencing.

However, if no WWPNs are detected, there will be messages logged to the following file:

- IRIX: `/var/adm/cxfs_client`
- Linux: `/var/log/cxfs_client`

If no WWPNs are detected, you can manually specify the WWPNs in the `/etc/fencing.conf` fencing file for the Linux platform. This method does not work if the WWPNs are partially discovered.

The fencing file is not used on the IRIX platform.

The fencing file enumerates the worldwide port name for all of the HBAs that will be used to mount a CXFS filesystem. There must be a line for the HBA WWPN as a 64-bit hexadecimal number.

Note: The WWPN is that of the HBA itself, **not** any of the devices that are visible to that HBA in the fabric.

If used, the fencing file must contain a simple list of WWPNs, one per line.

If you use the fencing file, you must update it whenever the HBA configuration changes, including the replacement of an HBA.

Do the following:

1. Set up the switch and HBA.

2. Follow the Fibre Channel cable on the back of the node to determine the port to which it is connected in the switch. Ports are numbered beginning with 0. (For example, if there are 8 ports, they will be numbered 0 through 7.)

3. Use the `telnet` command to connect to the switch and log in as user `admin` (the password is `password` by default).

4. Execute the `switchshow` command to display the switches and their WWPN numbers.

   For example:

```
brocade04:admin> switchshow
switchName:     brocade04
switchType:     2.4
switchState:    Online
switchRole:     Principal
switchDomain:   6
switchId:       fffc06
switchWwn:      10:00:00:60:69:12:11:9e
switchBeacon:   OFF
port  0: sw  Online        F-Port  20:00:00:01:73:00:2c:0b
port  1: cu  Online        F-Port  21:00:00:e0:8b:02:36:49
port  2: cu  Online        F-Port  21:00:00:e0:8b:02:12:49
port  3: sw  Online        F-Port  20:00:00:01:73:00:2d:3e
port  4: cu  Online        F-Port  21:00:00:e0:8b:02:18:96
port  5: cu  Online        F-Port  21:00:00:e0:8b:00:90:8e
port  6: sw  Online        F-Port  20:00:00:01:73:00:3b:5f
port  7: sw  Online        F-Port  20:00:00:01:73:00:33:76
port  8: sw  Online        F-Port  21:00:00:e0:8b:01:d2:57
port  9: sw  Online        F-Port  21:00:00:e0:8b:01:0c:57
port 10: sw  Online        F-Port  20:08:00:a0:b8:0c:13:c9
port 11: sw  Online        F-Port  20:0a:00:a0:b8:0c:04:5a
port 12: sw  Online        F-Port  20:0c:00:a0:b8:0c:24:76
port 13: sw  Online        L-Port  1 public
port 14: sw  No_Light
port 15: cu  Online        F-Port  21:00:00:e0:8b:00:42:d8
```

   The WWPN is the hexadecimal string to the right of the port number. For example, the WWPN for port 0 is `2000000173002c0b` (you must remove the colons from the WWPN reported in the `switchshow` output to produce the string to be used in the fencing file).

5. Create the `/etc/fencing.conf` fencing file and add the WWPN for the port determined in step 2. (Comment lines begin with #.)

   For dual-ported HBAs, you must include the WWPNs of any ports that are used to access cluster disks. This may result in multiple WWPNs per HBA in the file; the numbers will probably differ by a single digit.

   For example, if you determined that port 0 is the port connected to the switch, your fencing file should contain the following:

   ```
   # WWPN of the HBA installed on this system
   #
   2000000173002c0b
   ```

6. After the node is added to the cluster, enable the fencing feature by using the CXFS GUI or `cmgr` command on a CXFS administration node.

XFS Internal Errors in System Log File

After a filesystem has been defined in CXFS, running `mkfs` on it (or using "Make Filesystems with the GUI" on page 200) will cause XFS internal errors to appear in the system log file. For example (line breaks added for readability):

```
Aug 17 09:25:52 1A:yokohama-mds1 unix: ALERT: Filesystem "(NULL)": XFS internal error
xfs_mount_validate_sb(4) at line 237 of file ../fs/xfs/xfs_mount.c.
Caller 0xc000000000326ef4

Aug 17 09:14:52 6X:yokohama-mds1 clconfd[360]: < E clconf 11> CI_FAILURE, fsinfo_update(/dev/cxvm/work)
kernel returned 1010 (Filesystem is corrupted)
```

To avoid these errors, run `mkfs` before defining the filesystem in CXFS, or delete the CXFS filesystem before running `mkfs`. See "Delete a CXFS Filesystem with the GUI" on page 210, and "Delete a CXFS Filesystem with `cmgr`" on page 272.

Multiple Ethernet Interfaces on Altix Systems

In Altix systems with multiple Ethernet interfaces, the default behavior of the operating system is to dynamically assign interface names (such as `eth0`, `eth1`, and so on) at boot time. Therefore, the physical interface associated with the `eth0` device may change after a system reboot; if this occurs, it will cause a networking problem for CXFS. To avoid this problem, provide persistent device naming by using the `/etc/sysconfig/networking/eth0_persist` file to map specific Ethernet device names to specific MAC addresses. Adding lines of the format to the `eth0_persist` file:

eth*N MAC_ID*

For example:

```
eth0 08:00:69:13:dc:ec
eth1 08:00:69:13:72:e8
```

For more information about persistent naming, see *SGI ProPack for Linux Start Here*.

Clients Unable to Remount Filesystems

If you have multiple metadata servers in the cluster but only one potential metadata server defined for a given filesystem and that server goes down, the now server-less filesystem goes into a shutdown state. Although the clients maintain membership in the cluster, they will not remount the filesystem automatically when the potential metadata server comes back up. You must manually unmount the filesystem.

If there had been only one potential metadata server in the cluster, the filesystem's clients would have lost membership and gone through a forced shutdown, which automatically unmounts the filesystems.

Understanding Error Messages

This section describes some of the error messages you may see. In general, the example messages are listed first by type and then in alphabetical order, starting with the message identifier or text.

Sections are as follows:

- "Normal Messages" on page 413
- "Relocation Error" on page 415
- "Controller Disable Messages" on page 415
- "CMS Error Messages" on page 415
- "`clconfd` Daemon Death" on page 416
- "Out of Logical Swap Space" on page 416

- "No Cluster Name ID Error" on page 417
- "Lost CXFS Membership" on page 418
- "License Error" on page 418
- "IP Address Error" on page 419
- "System Log File Errors" on page 419
- "Log File Error Messages" on page 428

Normal Messages

You can expect to see the following messages. They are normal and do not indicate a problem.

```
NOTICE: Error reading mesg header 4 channel 1 cell 2
```

Error number 4 (`EINTR`) on `MEMBERSHIP` message channel (channel 1; channel 0 is the main channel for CXFS and XVM data) for connection with node 2. The `EINTR` indicates that this message channel is purposely being torn down and does not indicate an error in itself. (Any other error number is a real error that will cause the local node to declare the other node failed.) This is an informative message; no corrective action is required.

```
NOTICE: Membership delivered.  Membership contains 0(21) 1(12)
cells
```

The format `x(y)` shows the cell ID and membership age: in this example, cell 0 and cell 1 are in the CXFS membership; cell 0 has been in the last 21 CXFS memberships, cell 1 has been in the last 12.

```
NOTICE: Resetting cells 0x4
```

The number here is a bitmask of node numbers on which a reset is being requested. In this case, `0x4` equates to node 2. This is an informative message; no corrective action is required.

```
CI_FAILURE, Cell 1 Machine cxfs1:  server has no information
about a machine that has reset capabilities for this machine
```

A reset mechanism was not provided for this node. The node will not be automatically reset if it fails. If you do not have reset capability, this message can be ignored. System reset configuration is recommended for all potential metadata servers.

```
NOTICE: Error reading mesg header 4 channel 1 cell 2
```

The `mesg header 4` text indicates that this is just an informative message.

```
clconfd[16574]:  <<CI> E config 2> CI_ERR_NOTFOUND, Error
reading CMS status for machine tango, assuming machine is
FailSafe-disabled in cluster twango.
```

This indicates that the cluster is CXFS only and that you are not using FailSafe.

```
CI_CLCONFERR_INIT in ep_name() not binding socket
```

This message appears before the daemons start.

```
clconfd[16574]:  <<CI> E clconf 0> CI_CLCONFERR_INIT, in
ep_name():  not binding socket
```

This `clconfd` message appears when daemons are starting up.

```
date <I0 clconfd clconf 610:0 clconfd_client.c:84> client
registration:  clconfinfo, id 9119
date<I0 clconfd clconf 610:0 clconfd_service.c:781> sending reply
configuration and membership msg to client:  clconfinfo, id 9119
date <I0 clconfd clconf 610:0 clconfd_client.c:96> client
un-registration:  clconfinfo, id 9119
```

These messages are issued if you run the `clcon_info` command. The `clconf_info` command first registers as a CXFS client with `clconfd`; it then gets a reply message to its request for configuration and membership status; finally, it unregisters when it is done.

```
date <I0 clconfd clconf 610:0 clconfd_service.c:781 sending reply
configuration and membership msg to client:  cad, id 602
```

This message indicates that the `cad` daemon is polling `clconfd` for status regularly. `cad` does not register and unregister each time like `clconf_info` because it is a daemon and it does not exit after each request. You will see register/unregister messages for `cad` only when `cad` or `clconfd` restarts.

```
dcvn_import_force:  error 1502 from invk_dsvn_obtain_exist
```

This is a normal message sent during the recovery process.

Relocation Error

If you try to relocate a filesystem and see an error similar to the following example, it means that relocation has not been enabled:

```
[root@node1 bin]# /usr/cluster/bin/cmgr -p
Welcome to SGI Cluster Manager Command-Line Interface

cmgr> set cluster linuxsan
cmgr> admin cxfs_relocate cxfs_filesystem lsan1 to node node1
CMD(/bin/mount -n -o remount,set_server=node1 /lsan1): exited with status
32 (0x20)

Failed to admin:
        cxfs_relocate

admin command failed
```

To allow the relocation to occur, you must enable relocation as specified in "Relocation" on page 20.

Controller Disable Messages

If you see messages such as the following on the console or in a message log, it means that the Fibre Channel switch is misconfigured:

```
controller disable is not supported on loop
```

CXFS fencing recovery operations do not support loop mode. Verify that all Fibre Channel switches are configured correctly. See the switch documentation for configuration information.

CMS Error Messages

The following messages may be logged by CMS.

`CMS excluded cells` *0xXXX* `with incomplete connectivity`

Generated when CMS delivers a membership that excluded some **new** cells that had not established connections with enough cells yet to be admitted. *0xXXX* is a bitmask of excluded cells.

`CMS calculation limited to last membership:configuration change incomplete on cells` *0xXXX*

Generated when the leader is attempting to make a configuration change current (that is, actually use the change on all nodes), but some cells in the cluster have not yet gotten the configuration change staged (uploaded and ready to be made current). *0xXXX* is a bitmask of cells that do not yet have the change in their configuration. Changes make their way through the cluster asynchronously, so this situation is expected. It can take a few attempts by the CMS leader before all nodes have the change staged. As long as this situation resolves eventually, there is no problem. For more information, use `idbg cms_info`.

`CMS calculation limited to last membership:recovery incomplete`

Generated when new members were disallowed due to recovery from the last cell failure that is still being processed.

`clconfd` Daemon Death

If the `clconfd` daemon exits immediately after it starts up, it means that the CXFS license has not been properly installed. For information about the associated error message, see "License Error" on page 418.

You must install the license on each node before you can use CXFS. If you increase the number of CPUs in your system, you may need a new license. See Chapter 2, "CXFS and XVM FLEXlm Licenses" on page 51.

Out of Logical Swap Space

The following example system log file message indicates an oversubscribed system:

```
ALERT: inetd [164] - out of logical swap space during fork while
allocating uarea - see swap(1M)
Availsmem 8207 availrmem 427 rlx freemem 10, real freemem 9
```

See "Use System Capacity Wisely" on page 396.

The cluster daemons could also be leaking memory in this case. You may need to restart them:

- On administration nodes:
  - IRIX:

    ```
    # /etc/init.d/cluster restart
    ```

  - Linux:

    ```
    # /etc/init.d/cxfs_cluster restart
    ```

- On client-only nodes:

  ```
  # killall cxfs_client
  # /etc/init.d/cxfs_client start
  ```

No Cluster Name ID Error

For example:

```
Mar  1 15:06:18 5A:nt-test-07 unix: NOTICE: Physvol (name cip4) has no
CLUSTER name id: set to ""
```

This message means the following:

- The disk labeled as an XVM physvol was probably labeled under IRIX 6.5.6f and the system was subsequently upgraded to a newer version that uses a new version of XVM label format. This does not indicate a problem.
- The cluster name had not yet been set when XVM encountered these disks with an XVM cluster physvol label on them. This is normal output when XVM performs the initial scan of the disk inventory, before node/cluster initialization has completed on this host.

  The message indicates that XVM sees a disk with an XVM cluster physvol label, but that this node has not yet joined a CXFS membership; therefore, the cluster name is empty ("").

  When a node or cluster initializes, XVM rescans the disk inventory, searching for XVM cluster physvol labels. At that point, the cluster name should be set for this host. An empty cluster name after node/cluster initialization indicates a problem with cluster initialization.

  The first time any configuration change is made to any XVM element on this disk, the label will be updated and converted to the new label format, and these notices will go away.

For more information about XVM, see the *XVM Volume Manager Administrator's Guide*.

Lost CXFS Membership

The following message in the system log file indicates a kernel-triggered revocation of CXFS membership:

```
Membership lost - withdrawing from cluster
```

You must actively allow CXFS membership for the local node in this situation. See "Allow Membership of the Local Node with the GUI" on page 195, or "Allow Membership of the Local Node with `cmgr`" on page 259.

License Error

If you see the following message in the `/var/cluster/ha/log/clconf_hostname` logfile, it means that the CXFS license was not properly installed:

```
CXFS not properly licensed for this host.  Run
              '/usr/cluster/bin/cxfslicense -d'
       for detailed failure information.
```

If you do not have the CXFS license properly installed, you will see an error on the console when trying to run CXFS. For example, on a Linux node:

```
Cluster services:CXFS not properly licensed for this host.  Run
        '/usr/cluster/bin/cxfslicense -d'
for detailed failure information.  After fixing the
license, please run '/etc/init.d/cxfs_cluster restart'.
```

An error such as the following example will appear in the system log file:

```
Mar  4 12:58:05 6X:typhoon-q32 crsd[533]: <<CI> N crs 0> Crsd restarted.
Mar  4 12:58:05 6X:typhoon-q32 clconfd[537]: <<CI> N clconf 0>
Mar  4 12:58:05 5B:typhoon-q32 CLCONFD failed the CXFS license check.Use the
Mar  4 12:58:05 5B:typhoon-q32    '/usr/cluster/bin/cxfslicense -d'
Mar  4 12:58:05 5B:typhoon-q32 command to diagnose the license problem.
```

If the `clconfd` daemon dies right after it starts up, this error is present.

You must install the license on each node before you can use CXFS. See Chapter 2, "CXFS and XVM FLEXlm Licenses" on page 51.

IP Address Error

If you have conflicting cluster ID numbers at your site, you will see errors such as the following:

```
WARNING: mtcp ignoring  alive message from 1 with wrong ip addr 128.162.89.34
WARNING: mtcp ignoring  alive message from 0 with wrong ip addr 128.162.89.33
```

A cluster ID number must be unique. To solve this problem, make the cluster ID numbers unique.

This error can occur if you redefine the cluster configuration and start CXFS services while some nodes have stale information from a previous configuration.

To solve the problem, first try the steps in "Eliminate a Residual Cluster" on page 392. If that does not work, reboot the nodes that have stale information. You can determine which nodes have stale information as follows: stale nodes will complain about all of the nodes, but the up-to-date nodes will complain only about the stale nodes. The `/var/cluster/ha/log/clconfd_` log file on the stale nodes will also show error messages about `SGI_CMS_CONFIG_ID` failures.

If there are too many error messages to recognize the stale nodes, reboot every node.

System Log File Errors

CXFS logs both normal operations and critical errors to the system log file, as well as to individual log files for each log group.

The system log files are:

- IRIX: `/var/adm/SYSLOG`
- Linux: `/var/log/messages`

In general, errors in the system log file file take the following form:

timestamp priority_&_facility : hostname process[ID]: <internal_info> CODE message_text

For example:

```
Sep  7 11:12:59 6X:cxfs0 cli[5830]: < E clconf 0> CI_IPCERR_NOSERVER, clconf
ipc: ipcclnt_connect() failed, file /var/cluster/ha/comm/clconfd-ipc_cxfs0
```

Table 17-1 shows the parts of the preceding message.

Table 17-1 System Log File Error Message Format

Content	Part	Meaning
`Sep 7 11:12:59`	Time Stamp	September 7 at 11:12 AM.
`6X`	Facility and level	6X indicates an informational message. See `syslogd` and the file `/usr/include/sys/syslog.h`.
`cxfs0`	Node name	The node whose logical name is `cxfs0` is the node on which the process is running.
`cli[5830]`	Process[ID]	The process sending the message is `cli` and its process ID number is `5830`.
`<CI>E clconf 0`	Internal information: message source, logging subsystem, and thread ID	The message is from the cluster infrastructure (CI). `E` indicates that it is an error. The `clconf` command is the logging subsystem. `0` indicates that it is not multithreaded.
`CI_IPCERR_NOSERVER, clconf ipc`	Internal error code	Information about the type of message; in this case, a message indicating that the server is missing. No error code is printed if it is a normal message.
`ipcclnt_connect() failed, file /var/cluster/ha/comm/clconfd-ipc_cxfs0`	Message text	A connection failed for the `clconfd-ipc_cxfs0` file.

The following sections present only the message identifiers and text.

`cli` Error Messages

For all `cli` messages, only the last message from the command (which begins with `CLI private command failed`) is meaningful. You can ignore all other `cli` messages.

The following are example errors from the `cli` daemon.

`CI_ERR_INVAL, CLI private command:  failed (Machine (cxfs0) exists.)`

You tried to create a new node definition with logical name `cxfs0`; however, that node name already exists in the cluster database. Choose a different name.

`CI_ERR_INVAL, CLI private command:  failed (IP address (128.162.89.33) specified for control network is cxfs0 is assigned to control network of machine (cxfs0).)`

You specified the same IP address for two different control networks of node `cxfs0`. Use a different IP address.

`CI_FAILURE, CLI private command:  failed (Unable to validate hostname of machine (cxfs0) being modified.)`

The DNS resolution of the `cxfs0` name failed. To solve this problem, add an entry for `cxfs0` in `/etc/hosts` on all nodes.

`CI_IPCERR_NOPULSE, CLI private command:  failed (Cluster state is UNKNOWN.)`

The cluster state is `UNKNOWN` and the command could not complete. This is a transient error. However, if it persists, stop and restart the cluster daemons; see "Stopping and Restarting Cluster Administration Daemons" on page 439.

`clconfd` Error Messages

The following errors are sent by the `clconfd` daemon.

`CI_CONFERR_NOTFOUND, Could not access root node.`

The cluster database is either non-existent or corrupted, or the database daemons are not responding. Check that the database does exist.

If you get an error or the dump is empty, re-create the database; for more information, see "Clearing the Cluster Database" on page 435.

If the database exists, restart the cluster daemons; see "Stopping and Restarting Cluster Administration Daemons" on page 439.

`CI_ERR_NOTFOUND, Could not get Cellular status for local machine (cxfs1)`

The database is corrupted or cannot be accessed. Same actions as above.

`CI_FAILURE, Call to open cdb for logging configuration when it is already open.`

This indicates a software problem requiring you to restart the daemons; see "Stopping and Restarting Cluster Administration Daemons" on page 439.

`CI_FAILURE, Cell 1 Machine cxfs1:  server has no information about a machine that has reset capabilities for this machine`

A reset mechanism was not provided for this node. The node will not be automatically reset if it fails. To ensure proper failure handling, use the GUI or the `cmgr` command to modify the node's definition and add reset information. System reset configuration is recommended for all potential metadata servers. See "Define a Node with the GUI" on page 172, or "Modify a Node with `cmgr`" on page 234.

`CI_FAILURE, CMD(/sbin/umount -k /dev/xvm/bob1):  exited with status 1 (0x1)`

An error occurred when trying to unmount the `/dev/xvm/bob1` filesystem. Messages from the `umount` command are usually issued just before this message and provide more information about the reason for the failure.

`CI_FAILURE, CMD(/sbin/clmount -o 'server_list=(cxfs0,cxfs1)' /dev/xvm/bob2 /bob2):  exited with status 1 (0x1)`

An error occurred when trying to mount the `/dev/xvm/bob2` filesystem. Messages from the `mount` command are usually issued just before this message and provide more information about the reason of the failure.

`CI_FAILURE, CMD(/sbin/clmount -o 'server_list=(cxfs2,cxfs0)' /dev/xvm/stripe4 /xvm/stripe4):  exited with status 1 (0x1)`

You have tried to mount a filesystem without first running `mkfs`. You must use `mkfs` to construct the filesystem before mounting it. For more information, see the `mkfs` man page.

`CI_FAILURE, Could not write newincarnation number to CDB, error = 9.`

There was a problem accessing the cluster database. Retry the operation. If the error persists, stop and restart the cluster daemons; see "Stopping and Restarting Cluster Administration Daemons" on page 439.

If the problem persists, clear the database, reboot, and re-create the database. See "Clearing the Cluster Database" on page 435.

`CI_FAILURE, Exiting, monitoring agent should revive me.`

The daemon requires fresh data. It will be automatically restarted.

`CI_FAILURE, No node for client (3) of filesystem (/dev/xvm/bob1) on (/bob1).`

(There may be many repetitions of this message.) The filesystem appears to still be mounted on a CXFS client node that is no longer in the cluster database. If you can identify the CXFS client node that used to be in the cluster and still has the filesystem mounted, reboot that node. Otherwise, reboot the entire cluster.

`CI_FAILURE, No node for server (-1) of filesystem (/dev/xvm/bob1) on (/bob1).`

(There may be many repetitions of this message.) The filesystem appears to still be mounted on a server node that is no longer in the cluster database. If you can identify the server node that used to be in the cluster and still has the filesystem mounted, reboot that node. Otherwise, reboot the entire cluster.

`CI_ FAILURE, Node cxfs0:  SGI_CMS_HOST_ID(tcp,128.162.8 >9.33) error 149 (Operation already in progress)`

The kernel already had this information; you can ignore this message.

`CI_FAILURE, Unregistered from crs.`

The `clconfd` daemon is no longer connected to the reset daemon and will not be able to handle resets of failed nodes. There is no corrective action.

```
CI_IPCERR_NOSERVER, Crs_register failed,will retry later.
Resetting not possible yet.
```

The `clconfd` daemon cannot connect to the reset daemon. It will not be able to handle resets of failed nodes. Check the reset daemon's log file (`/var/cluster/ha/log/crsd_`) for more error messages.

```
Clconfd is out of membership, will restart after notifying
clients.
```

The `clconfd` daemon does not have enough information about the current state of the cluster. It will exit and be automatically restarted with fresh data.

```
CMD(/sbin/clmount -o 'server_list=(cxfs2,cxfs0)'
/dev/xvm/stripe4 /xvm/stripe4):  /dev/xvm/stripe4:  Invalid
argument
```

You have tried to mount a filesystem without first running `mkfs`. You must use `mkfs` to construct the filesystem before mounting it. For more information, see the `mkfs` man page.

```
CMD(/sbin/clmount -o 'server_list=(cxfs0,cxfs1)' /dev/xvm/bob2
/bob2):  /dev/xvm/bob2:  Invalid argumentSep 9 14:12:43 6X:cxfs0
clconfd[345]:  < E clconf 3> CI_FAILURE, CMD(/sbin/clmount -o
'server_list=(cxfs0,cxfs1)' /dev/xvm/bob2 /bob2):  exited with
status 1 (0x1)
```

The first message comes from the `clmount` command (the internal CXFS mount command) and explains the error (an invalid argument was issued). The second message says that the mount failed.

`crsd` Error Messages

The following errors are sent by the `crsd` daemon.

```
CI_ERR_NOTFOUND, No logging entries found for group crsd, no
logging will take place - Database entry #global#logging#crsd
not found.
```

No `crsd` logging definition was found in the cluster database. This can happen if you start cluster processes without creating the database. See "Recreating the Cluster Database" on page 440.

`CI_ERR_RETRY, Could not find machine listing.`

The `crsd` daemon could not find the local node in the cluster database. You can ignore this message if the local node definition has not yet been created.

`CI_ERR_SYS:125, bind() failed.`

The `sgi-crsd` port number in the `/etc/services` file is not unique, or there is no `sgi-crsd` entry in the file. For information about adding this entry, see "`/etc/services` on CXFS Administration Nodes" on page 96.

`CI_FAILURE, Entry for sgi-crsd is missing in /etc/services.`

The `sgi-crsd` entry is missing from the `/etc/services` file. For information about adding this entry, see "`/etc/services` on CXFS Administration Nodes" on page 96.

`CI_FAILURE, Initialization failed, exiting.`

A sequence of messages will be ended with this message; see the messages prior to this one in order to determine the cause of the failure.

`cmond` Error Messages

The following errors are sent by the `cmond` daemon.

`Could not register for notification.cdb_error = 7`

An error number of `7` indicates that the cluster database was not initialized when the cluster process was started.

This may be caused if you execute the `cdbreinit` on one CXFS administration node while some other CXFS administration nodes in the pool are still running `fs2d` and already have the node listed in the database.

Do the following:

1. Execute the following command on the nodes that show the error:

```
# /usr/cluster/bin/cdb-init-std-nodes
```

This command will recreate the missing nodes without disrupting the rest of the database.

2. If the error persists, force the daemons to restart by executing the following command on IRIX:

   ```
   # /etc/init.d/cluster restart
   ```

   On Linux:

   ```
   # /etc/init.d/cxfs_cluster restart
   ```

   Verify that `cmond` is restarted.

3. If the error persists, reinitialize the database on just the node that is having problems.

4. If the error still persists, reinitialize all nodes in the cluster.

See "Recreating the Cluster Database" on page 440.

```
Process clconfd:343 of group cluster_cx exited, status = 3.
```

The `clconfd` process exited with status 3, meaning that the process will not be restarted by `cmond`. No corrective action is needed.

```
Process crsd:1790 of group cluster_control exited, status = 127
```

The `crsd` process exited with an error (nonzero) status. Look at the corresponding daemon logs for error messages.

`cxfs_client` Error Messages

The following errors are sent by the `cxfs_client` daemon.

```
cxfs_client:  cis_get_hba_wwns warning:  fencing configuration
file "fencing.conf" not found
```

The fencing file was not found, therefore the fencing configuration will not be updated on the server.

```
cxfs_client:op_failed ERROR: Mount failed for concat0
```

A filesystem mount has failed and will be retried.

`fs2d` Error Messages

The following errors are sent by the `fs2d` daemon.

```
Error 9 writing CDB info attribute for node
#cluster#elaine#machines#cxfs2#Cellular#status
```

An internal error occurred when writing to the cluster database. Retry the operation. If the error persists, stop and restart the cluster daemons; see "Stopping and Restarting Cluster Administration Daemons" on page 439.

If the problem persists, clear the database, reboot, and re-create the database. See "Clearing the Cluster Database" on page 435.

```
Error 9 writing CDB string value for node
#cluster#elaine#machines#cxfs2#Cellular#status
```

An internal error occurred when writing to the cluster database. Retry the operation. If the error persists, stop and restart the cluster daemons; see "Stopping and Restarting Cluster Administration Daemons" on page 439.

If the problem persists, clear the database, reboot, and re-create the database. See "Clearing the Cluster Database" on page 435.

```
Failed to update CDB for node
#cluster#elaine#Cellular#FileSystems#fs1#FSStatus
```

An internal error occurred when writing to the cluster database. Retry the operation. If the error persists, stop and restart the cluster daemons; see "Stopping and Restarting Cluster Administration Daemons" on page 439.

If the problem persists, clear the database, reboot, and re-create the database. See "Clearing the Cluster Database" on page 435.

```
Failed to update CDB for node
#cluster#elaine#machines#cxfs2#Cellular#status
```

An internal error occurred when writing to the cluster database. Retry the operation. If the error persists, stop and restart the cluster daemons; see "Stopping and Restarting Cluster Administration Daemons" on page 439.

If the problem persists, clear the database, reboot, and re-create the database. See "Clearing the Cluster Database" on page 435.

```
Machine 101 machine_sync failed with lock_timeout error
```

The `fs2d` daemon was not able to synchronize the cluster database and the `sync` process timed out. This operation will be retried automatically by `fs2d`.

```
ALERT: CXFS Recovery:  Cell 0:  Server Cell 2 Died, Recovering
```

The server (cell 2) died and the system is now recovering a filesystem.

General Messages

```
CI_CONFERR_NOTFOUND, Logging configuration error:  could not
read cluster database /var/cluster/cdb/cdb.db, cdb error = 3.
```

The cluster database has not been initialized. See "Recreating the Cluster Database" on page 440.

```
WARNING: Error receiving messages from cell 2 tcpchannel 1
```

There has been an error on the CXFS membership channel (channel 1; channel 0 is the main message channel for CXFS and XVM data). This may be a result of tearing down the channel or may be an error of the node (node with an ID of 2 in this case). There is no corrective action.

Log File Error Messages

CXFS maintains logs for each of the CXFS daemons. For information about customizing these logs, see "Set Log Configuration with the GUI" on page 193.

Log file messages take the following form:

*daemon*_`log` *timestamp internal_process: message_text*

For example:

```
cad_log:Thu Sep  2 17:25:06.092  cclconf_poll_clconfd: clconf_poll failed with error CI_IPCERR_NOPULSE
```

Table 17-2 on page 429, shows the parts in the preceding message.

Table 17-2 Log File Error Message Format

Content	Part	Meaning
`cad_log`	Daemon identifier	The message pertains to the `cad` daemon
`Sep 2 17:25:06.092`	Time stamp and process ID	September 2 at 5:25 PM, process ID 92.
`cclconf_poll_clconfd`	Internal process information	Internal process information
`clconf_poll failed with error CI_IPCERR_NOPULSE`	Message text	The `clconfd` daemon could not be contacted to get an update on the cluster's status.

cad Messages

The following are examples of messages from `/var/cluster/ha/log/cad_log`:

```
ccacdb_cam_open:  failed to open connection to CAM
server error 4
```

Internal message that can be ignored because the `cad` operation is automatically retried.

```
ccamail_cam_open:  failed to open connection to CAM
server error 4
```

Internal message that can be ignored because the `cad` operation is automatically retried.

```
ccicdb_cam_open:  failed to open connection to CAM
server error 4
```

Internal message that can be ignored because the `cad` operation is automatically retried.

```
cclconf_cam_open:  failed to open connection to CAM
server error 4
```

Internal message that can be ignored because the `cad` operation is automatically retried.

```
cclconf_poll_clconfd:  clconf_poll failed with error
CI_IPCERR_NOCONN
```

The `clconfd` daemon is not running or is not responding to external requests. If the error persists, stop and restart the cluster daemons; see "Stopping and Restarting Cluster Administration Daemons" on page 439.

```
cclconf_poll_clconfd:  clconf_poll failed with error
CI_IPCERR_NOPULSE
```

The `clconfd` daemon could not be contacted to get an update on the cluster's status. If the error persists, stop and restart the cluster daemons; see "Stopping and Restarting Cluster Administration Daemons" on page 439.

```
cclconf_poll_clconfd:  clconf_poll failed with error
CI_CLCONFERR_LONELY
```

The `clconfd` daemon does not have enough information to provide an accurate status of the cluster. It will automatically restart with fresh data and resume its service.

```
csrm_cam_open:  failed to open connection to CAM server error 4
```

Internal message that can be ignored because the `cad` operation is automatically retried.

```
Could not execute notification cmd.  system() failed.  Error:
No child processes
```

No mail message was sent because `cad` could not fork processes. Stop and restart the cluster daemons; see "Stopping and Restarting Cluster Administration Daemons" on page 439.

```
error 3 sending event notification to client 0x000000021010f078
```

GUI process exited without cleaning up.

`error 8 sending event notification to client 0x000000031010f138`

GUI process exited without cleaning up.

`cli` Messages

The following are examples of messages from `/var/cluster/ha/log/cli_`*hostname*:

`CI_CONFERR_NOTFOUND, No machines found in the CDB.`

The local node is not defined in the cluster database.

`CI_ERR_INVAL, Cluster (bob) not defined`

The cluster called `bob` is not present in the cluster database.

```
CI_ERR_INVAL, CLI private command:  failed (Cluster (bob) not
defined)
```

The cluster called `bob` is not present in the cluster database.

```
CI_IPCERR_AGAIN, ipcclnt_connect():  file
/var/cluster/ha/comm/clconfd-ipc_cxfs0 lock failed - Permission
denied
```

The underlying command line interface (CLI) was invoked by a login other than `root`. You should only use `cmgr` when you are logged in as `root`.

```
CI_IPCERR_NOPULSE, CLI private command:  failed (Cluster state
is UNKNOWN.)
```

The cluster state could not be determined. Check if the `clconfd` daemon is running.

```
CI_IPCERR_NOPULSE, ipcclnt_pulse_internal():  server failed to
pulse
```

The cluster state could not be determined. Check if the `clconfd` daemon is running.

```
CI_IPCERR_NOSERVER, clconf ipc:  ipcclnt_connect() failed, file
/var/cluster/ha/comm/clconfd-ipc_cxfs0
```

The local node (`cxfs0`) is not defined in the cluster database.

```
CI_IPCERR_NOSERVER, Connection file
/var/cluster/ha/comm/clconfd-ipc_cxfs0 not present.
```

The local node (`cxfs0`) is not defined in the cluster database.

`crsd` Errors

The following are examples of messages from `/var/cluster/ha/log/crsd_`*hostname*:

```
CI_CONFERR_INVAL, Nodeid -1 is invalid.
I_CONFERR_INVAL, Error from ci_security_init().
CI_ERR_SYS:125, bind() failed.
CI_ERR_SYS:125, Initialization failed, exiting.
CI_ERR_NOTFOUND, Nodeid does not have a value.
CI_CONFERR_INVAL, Nodeid -1 is invalid.
```

For each of these messages, either the node ID was not provided in the node definition or the cluster processes were not running in that node when node definition was created in the cluster database. This is a warning that optional information is not available when expected.

```
CI_ERR_NOTFOUND, SystemController information for node cxfs2 not
found, requests will be ignored.
```

System controller information (optional information) was not provided for node `cxfs2`. Provide system controller information for node `cxfs2` by modifying node definition. This is a warning that optional information is not available when expected. Without this information, the node will not be reset if it fails, which might prevent the cluster from properly recovering from the failure.

```
CI_ERR_NOTFOUND, SystemController information for node cxfs0 not
found, requests will be ignored.
```

The owner node specified in the node definition for the node with a node ID of 101 has not been defined. You must define the owner node.

`CI_CRSERR_NOTFOUND, Reset request 0x10087d48 received for node 101, but its owner node does not exist.`

The owner node specified in the node definition for the node with a node ID of 101 has not been defined. You must define the owner node.

`fs2d` Errors

The following are examples of messages from `/var/cluster/ha/log/fs2d_`*hostname*:

`Failed to copy global CDB to node cxfs1 (1), error 4`

There are communication problems between the local node and node `cxfs2`. Check the control networks of the two nodes.

`Communication failure send new quorum to machine cxfs2 (102) (error 6003)`

There are communication problems between the local node and node `cxfs2`. Check the control networks of the two nodes.

`Failed to copy CDB transaction to node cxfs2 (1)`

There are communication problems between the local node and node `cxfs2`. Check the control networks of the two nodes.

`Outgoing RPC to` *hostname* `:  NULL`

If you see this message, check your Remote Procedure Call (RPC) setup. For more information, see the `rpcinfo`, `rpcinfo`, and `portmap` man pages.

`cdbreinit` Error Messages

```
Thu Jun  3 16:20:45.431 cxfsopus1.americas.sgi.com cbe_fs2   - cbe_create_node: cannot create new node (RPC error = 9
  libcdb       - cdb_create_node: error 9 creating child of node 0x0x60000000000135c0 with subkey "ifd1"
```

This error means that some nodes have not been created in the cluster database. Error 9 usually means that `fs2d` is has encountered an internal error while creating that node. To fix the problem, make sure that `fs2d` is not running on any administration-capable node and rerun `cdbreinit`.

Messages During Remote Installation

If you are performing a remote IRIX installation, you may see informational messages such as the following:

```
cdb-exitop: can’t run remotely - scheduling to run later
```

When you perform a remote or miniroot install, the `exitop` commands are deferred until cluster administration daemons are started, at which time the `exitop` commands will be run. For more information, see "Differences When Performing a Remote or Miniroot Install" on page 74.

Corrective Actions

This section covers the following corrective actions:

- "Restarting CXFS Services" on page 434
- "Clearing the Cluster Database" on page 435
- "Rebooting" on page 436
- "Recovering a Two-Node Cluster" on page 436
- "Rebooting without Rejoining the Cluster" on page 438
- "Stopping and Restarting Cluster Administration Daemons" on page 439
- "Recreating the Cluster Database" on page 440
- "Verifying Connectivity in a Multicast Environment" on page 440

Restarting CXFS Services

If CXFS services to do not restart after a reboot, it may be that the node was marked as `INACTIVE` in the cluster data base using the **Stop CXFS Services** function of the GUI or the `stop cx_services` command in `cmgr`. In this case, issuing a `/etc/init.d/cluster start` (IRIX) `/etc/init.d/cxfs_cluster start` (Linux) or will not restart the services.

You must manually start CXFS services. If you use the GUI or `cmgr` to restart the services, the configuration will be set so that future reboots will also restart CXFS services.

For information, see "Start CXFS Services with the GUI" on page 191, or "Start CXFS Services with `cmgr`" on page 254.

Clearing the Cluster Database

To clear the cluster database on all of the administration nodes of the cluster, do the following, completing each step on each administration node before moving to the next step:

Caution: This procedure deletes all configuration information.

1. Enter the following on all administration nodes:

   ```
   # /usr/cluster/bin/cmgr -c 'admin cxfs_stop'
   ```

2. Enter the following on all administration nodes:

   - IRIX:

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

   - Linux:

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

Caution: Complete steps 1 and 2 on each node before moving to step 3 for any node.

3. Enter the following on all administration nodes:

   ```
   # /usr/cluster/bin/cdbreinit
   ```

   See also "Reboot Before Changing Node ID or Cluster ID" on page 397.

4. Enter the following on all administration nodes:

   - IRIX:

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

- Linux:

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

5. Enter the following on all administration nodes:

   ```
   # /usr/cluster/bin/cmgr -c 'admin cxfs_start'
   ```

See "Eliminate a Residual Cluster" on page 392, to get rid of possible stale cluster configuration in the kernel. If needed, reboot the nodes.

Rebooting

Enter the following individually on every node to reboot the cluster:

```
# reboot
```

For information about nodes running operating systems other than IRIX or Linux, see the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

If you want CXFS services to restart whenever the node is rebooted, use the GUI or `cmgr` to start CXFS services. For information, see "Start CXFS Services with the GUI" on page 191, and "Start CXFS Services with `cmgr`" on page 254.

The following are situations that may require a rebooting:

- If some CXFS clients are unable to unmount a filesystem because of a busy vnode and a reset of the node does not fix the problem, you may need to reboot every node in the cluster
- If there is no recovery activity within 10 minutes, you may need to reboot the node

Recovering a Two-Node Cluster

Suppose the following:

1. You have cluster named `clusterA` that has two server-capable nodes and there is no CXFS tiebreaker:
   - `node1`
   - `node2`
2. `node1` goes down and will remain down for a while.

3. node2 recovers and clusterA remains up.

Note: An existing cluster can drop down to 50% of the remaining server-capable nodes **after** the initial CXFS kernel membership is formed. For more information, see "CXFS Kernel Membership, Quorum, and Tiebreaker" on page 462.

---

4. node2 goes down and therefore clusterA fails.
5. node2 comes back up. However, clusterA cannot form because the initialization of a cluster requires either:
   - **More than 50%** of the server-capable nodes
   - 50% of the server-capable nodes, **one of which is the CXFS tiebreaker**

To allow node2 to form a cluster by itself, you must do the following:

1. Set node2 to be the CXFS tiebreaker node, using either the GUI or cmgr:
   - See "Set Tiebreaker Node with the GUI" on page 192.
   - See "Set the Tiebreaker Node with cmgr" on page 255.
2. Revoke the CXFS kernel membership of node2:
   - See "Revoke Membership of the Local Node with the GUI" on page 195.
   - In cmgr, enter:

     ```
     cmgr> admin cxfs_stop
     ```

     See "Revoke Membership of the Local Node with cmgr" on page 259.
3. Allow CXFS kernel membership of node2:
   - See "Allow Membership of the Local Node with the GUI" on page 195.
   - In cmgr, enter:

     ```
     cmgr> admin cxfs_start
     ```

     See "Allow Membership of the Local Node with cmgr" on page 259.
4. Unset the CXFS tiebreaker node capability.

Caution: All two-server-capable node clusters without a tiebreaker set must have fencing or reset configured. SGI recommends reset.

Use either the GUI or `cmgr`:

- "Set Tiebreaker Node with the GUI" on page 192
- "Set the Tiebreaker Node with `cmgr`" on page 255

The cluster will attempt to communicate with the `node1` because it is still configured in the cluster, even though it is down. Therefore, it may take some time for the CXFS kernel membership to form and for filesystems to mount.

Rebooting without Rejoining the Cluster

The following arguments to `chkconfig` control the other cluster administration daemons and the replicated cluster database:

- IRIX: `cluster`
- Linux: `cxfs_cluster`

If they are turned off, the database daemons will not be started at the next reboot and the local copy of the database will not be updated if you make changes to the cluster configuration on the other nodes. This could cause problems later, especially if a majority of nodes are not running the database daemons.

If the cluster daemons are causing serious trouble and prevent the machine from booting, you can recover the node by booting in single-user mode, turning the argument off and booting in multiuser mode:

- IRIX:

```
irix# init 1
irix# /etc/chkconfig cluster off
irix# init 2
```

- Linux:

```
[root@linux root]# init 1
[root@linux root]# /bin/chkconfig cxfs_cluster off
[root@linux root]# init 3
```

For more information, see "CXFS `chkconfig` Arguments" on page 289.

Stopping and Restarting Cluster Administration Daemons

The commands to stop and restart cluster administration daemons depends upon the platform. See also "Restarting CXFS Services" on page 434. For general information about the daemons, see "Daemons" on page 445.

To stop and restart cluster administration daemons, enter the following:

- On administration nodes:
  - IRIX:

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

  - Linux:

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

- On client-only nodes:

  ```
  # killall cxfs_client
  # /etc/init.d/cxfs_client start
  ```

Note: You could also use the `restart` option to stop and start.

These commands affect the cluster administration daemons only.

Caution: When the cluster administration daemons are stopped, the node will not receive database updates and will not update the kernel configuration. This can have very unpleasant side effects. Under most circumstances, the administration daemons should remain running at all times. Use these commands only as directed.

Recreating the Cluster Database

To recreate the initial cluster database, do the following:

1. Ensure that the database membership quorum is held by nodes with a good database, in order to avoid propagating a bad database.
2. Enter the following:

```
# /usr/cluster/bin/cdbreinit
```

Note: See also "Reboot Before Changing Node ID or Cluster ID" on page 397.

Verifying Connectivity in a Multicast Environment

To verify general connectivity in a multicast environment, you can execute a `ping` command on the `224.0.0.1` IP address.

To verify the CXFS heartbeat, use the `224.0.0.250` IP address, which is the default CXFS heartbeat multicast address (because it is the default, this address does not have to appear in the `/etc/hosts` file).

Note: A node is capable of responding only when the administration daemons (`fs2d`, `cmond`, `cad`, and `crsd`) or the `cxfs_client` daemon is running.

For example, to see the response for two packets sent from IRIX IP address `163.154.17.49` to the multicast address for CXFS heartbeat and ignore loopback, enter the following:

```
irixnodeA# ping -c 2 -I 163.154.17.49 -L 224.0.0.250
PING 224.0.0.250 (224.0.0.250): 56 data bytes
64 bytes from 163.154.17.140: icmp_seq=0 ttl=64 time=1.146 ms
64 bytes from 163.154.17.55: icmp_seq=0 DUP! ttl=255 time=1.460 ms
64 bytes from 163.154.17.52: icmp_seq=0 DUP! ttl=255 time=4.607 ms
64 bytes from 163.154.17.50: icmp_seq=0 DUP! ttl=255 time=4.942 ms
64 bytes from 163.154.17.140: icmp_seq=1 ttl=64 time=2.692 ms

----224.0.0.250 PING Statistics----
2 packets transmitted, 2 packets received, +3 duplicates, 0.0% packet
loss
round-trip min/avg/max = 1.146/2.969/4.942 ms
```

The above output indicates that there is a response from the following addresses:

```
163.154.17.140
163.154.17.55
163.154.17.52
163.154.17.50
```

To override the default address, you can use the `-c` and `-m` options or make the name `cluster_mcast` resolvable on all nodes (such as in the `/etc/hosts` file). For more information, see the `cxfs_client` man page.

Reporting Problems to SGI

When reporting a problem about a CXFS node to SGI, you should retain the information discussed in this section, depending upon the circumstances you experience.

Reporting IRIX Problems

Retain the following information for IRIX nodes:

- If a panic has occurred on an IRIX node, retain the system core files in `/var/adm/crash`, including the following:

  ```
  analysis.number
  unix.number
  vmcore.number.comp
  ```

- For any type of problem, run the `/usr/cluster/bin/cxfsdump` utility on an IRIX node and retain the output. You can run this utility immediately after noticing a problem. The `cxfsdump` utility attempts to collect information from all nodes in the cluster by using the `rsh` command, including the following:

  – Information from the following files:

    ```
    /var/adm/SYSLOG
    /var/adm/cxfs_client        (for client-only nodes)
    /var/cluster/ha/log/*       (for administration nodes)
    /etc/failover.conf
    /etc/failover2.conf
    /var/sysgen/stune
    ```

```
/etc/hosts
```

– Output from the following commands:

```
/usr/cluster/bin/cdbutil gettree ’#’
/usr/sbin/versions -n
/usr/sbin/systune
/sbin/hinv -vm
/sbin/xvm show -v phys
/sbin/xvm show -top -v vol
/usr/sbin/scsifo -d
/usr/etc/netstat -ia
```

Reporting Linux Problems

Retain the following information for Linux nodes:

- The kernel you are running:

```
[root@linux root]# uname -a
```

- The CXFS packages you are running:

```
[root@linux root]# rpm -q cxfs_client cxfs-modules cxfs_utils xvm-cmds
```

- The number and types of processors in your machine:

```
[root@linux root]# cat /proc/cpuinfo
```

- The hardware installed on your machine:

```
[root@linux root]# /sbin/lspci
```

- Modules that are loaded on your machine:

```
[root@linux root]# /sbin/lsmod
```

- The `/var/log/cxfs_client` log file
- Any messages that appeared in the system logs immediately before the system exhibited the problem.
- Output about the cluster obtained from the `cxfsdump` utility run on an administration node.
- After a system kernel panic, the debugger information from the KDB built-in kernel debugger. See "Kernel Status Tools" on page 387

- Output from the following commands:
  - Information from the following files:

    ```
    /var/log/messages
    /var/adm/cxfs_client         (for client-only nodes)
    /var/cluster/ha/log/*        (for administration nodes)
    /etc/failover.conf
    /etc/hosts
    ```

  - Output from the following commands:

    ```
    /usr/cluster/bin/cdbutil gettree '#'
    /usr/bin/hinv
    /usr/bin/topology
    /sbin/xvm show -v phys
    /sbin/xvm show -top -v vol
    /bin/netstat -ia
    ```

Appendix A

CXFS Software Architecture

This appendix discusses the following for administration nodes:

- "Daemons"
- "Communication Paths" on page 448
- "Communication Paths in a Coexecution Cluster" on page 453
- "Flow of Metadata for Reads and Writes" on page 454

Also see the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

Daemons

The following table lists the CXFS daemons and threads. CXFS shares with XFS the IRIX `xfsd` and Linux `xfsdatad` kernel threads to push buffered writes to disk.

If you are using a coexecution (of type `CXFS and FailSafe`) cluster, see the *FailSafe Administrator's Guide for SGI InfiniteStorage*, for information about FailSafe daemons.

Note: On Linux, the process names begin with a `*` (such as `[*mtcp_notify]`).

Table A-1 CXFS Daemons and Threads

Layer	Subsystem	Process	Description
CXFS daemons	`cluster_services`	`clconfd`	CXFS control daemon for administration nodes. Reads the cluster configuration from the CDB database and manages the local kernel's CXFS kernel membership services accordingly.
	`cxfs_client`	`cxfs_client`	CXFS client daemon for client-only nodes. Manages the local kernel's CXFS kernel membership services accordingly.
Cluster software infrastructure (cluster administrative processes)	`cluster_admin`	`cad`	Cluster administration daemon. Provides administration services.
	`cluster_control`	`crsd`	Node control daemon. Monitors the serial connection to other nodes. Has the ability to reset other nodes.
		`cmond`	Daemon that manages all other daemons. This process starts other processes in all nodes in the cluster and restarts them on failures.
		`fs2d`	Manages the database and keeps each copy in synchronization on all nodes in the pool.
Kernel Threads	IRIX `sthreads`	`cmsd`	Manages CXFS kernel membership and heartbeating. (The CXFS `cmsd` resides in the kernel; it differs from the IRIS FailSafe `cmsd` that resides in user space.)
		`Recovery`	Manages recovery protocol for node.
		`corpseleader`	Coordinates recovery between nodes.

Layer	Subsystem	Process	Description
		`dcshake`	Purges idle CXFS vnodes on the CXFS client.
		`cxfsd`	Manages sending extent and size updates from the client to the server. This daemon (which runs on the CXFS client) takes modified inodes on the client and ships back any size and unwritten extent changes to the server.
	`xthreads`	`mesgtcprcv`	Reads messages (one per open message channel).
		`mesgtcpaccept`	Responsible for accepting new connections.
		`mesgtcpdiscovery`	Responsible for monitoring and discovering other nodes.
		`mesgtcpmulticast`	Responsible for supplying heartbeat.

The `fs2d`, `clconfd`, and `crsd` daemons run at real-time priority. However, the `mount` and `umount` commands and scripts executed by `clconfd` are run at normal, time-shared priority.

Communication Paths

The following figures show communication paths in CXFS.

Note: The following figures do not represent the `cmond` cluster manager daemon. The purpose of this daemon is to keep the other daemons running.

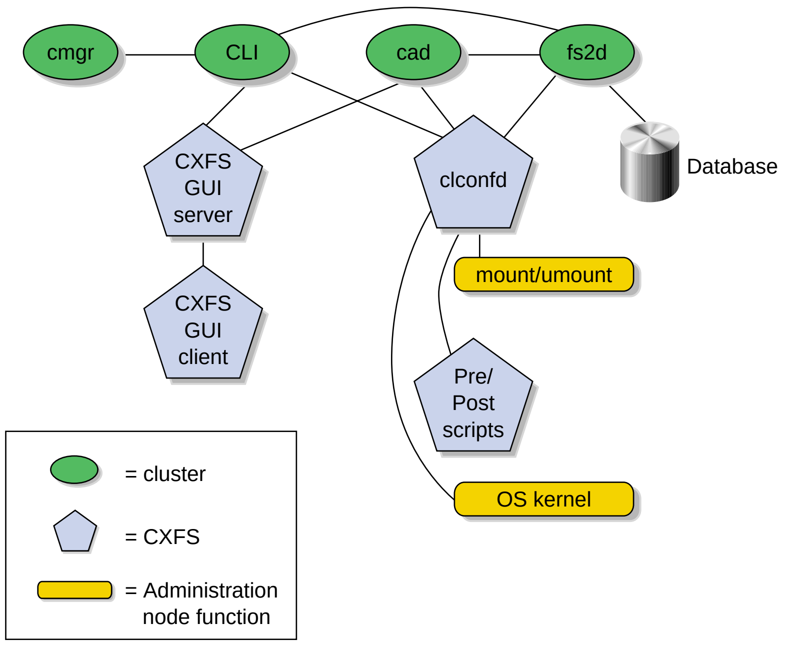


Figure A-1 Communication within One Administration Node

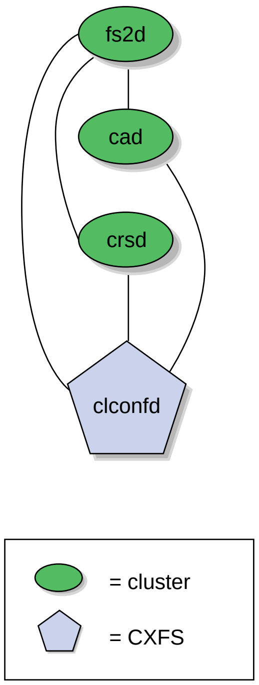


Figure A-2 Daemon Communication within One Administration Node

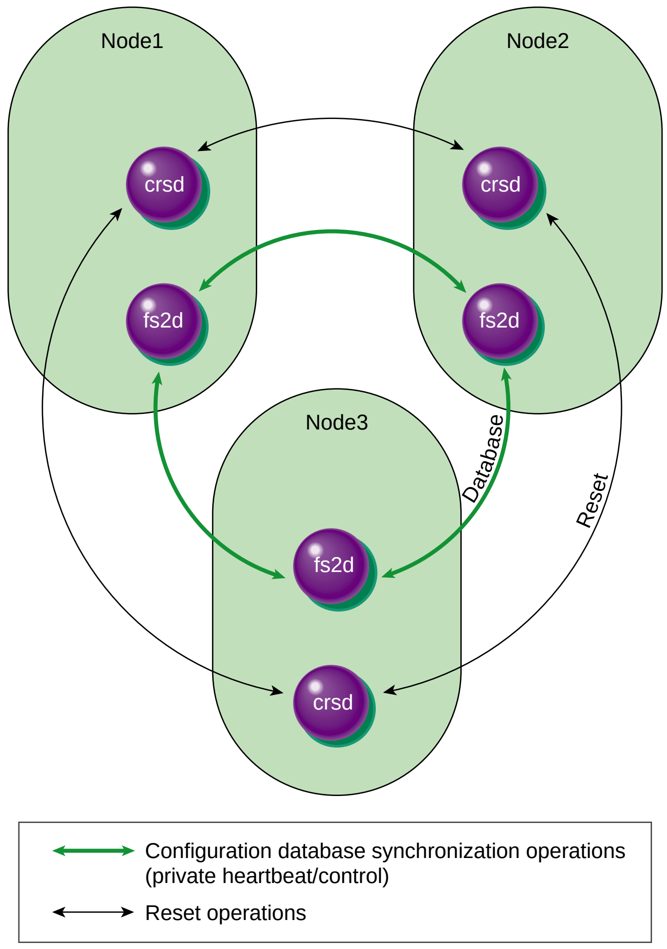

Node1
Node2
Node3
crsd
fs2d
crsd
fs2d
fs2d
crsd
Database
Reset
Configuration database synchronization operations (private heartbeat/control)
Reset operations

Figure A-3 Communication between Nodes in the Pool

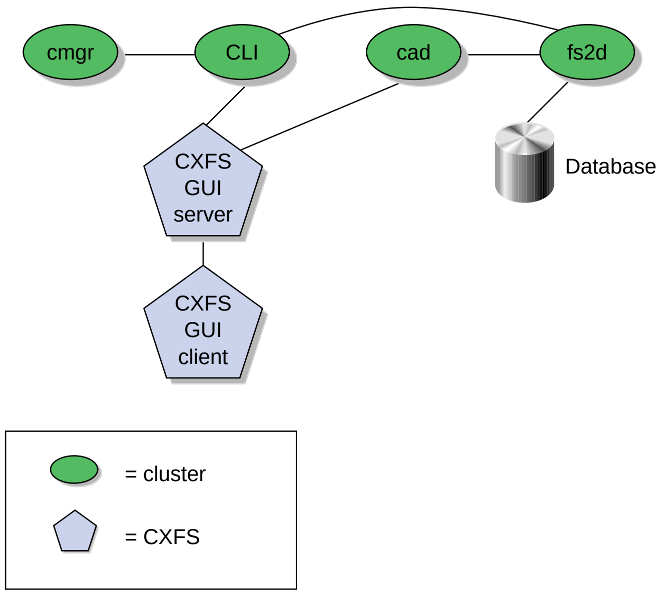


Figure A-4 Communication for an Administration Node Not in a Cluster

One of the administration nodes running the `fs2d` daemon is chosen to periodically multicasts its IP address and the generation number of the cluster database to each of the client-only nodes. Each time the database is changed, a new generation number is formed and multicast. The following figure describes the communication among nodes, using a Solaris client-only node as an example.

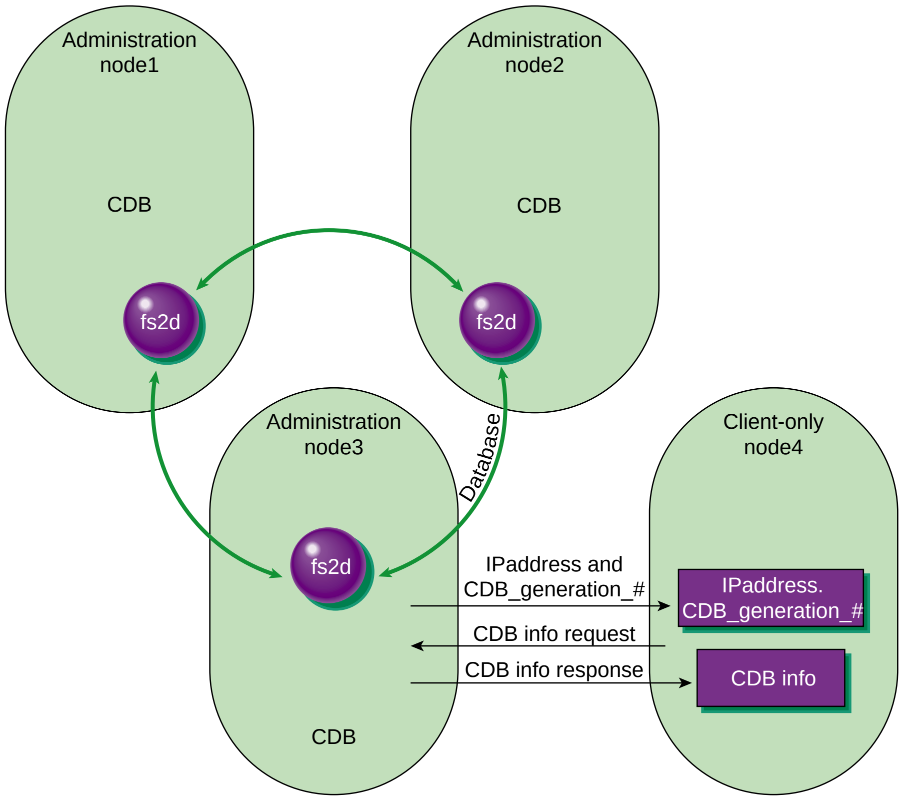


Figure A-5 Communication Among Administration Nodes and Client-Only Nodes

Communication Paths in a Coexecution Cluster

The following figures show the communication paths within one node in a coexecution cluster running CXFS and IRIS FailSafe.

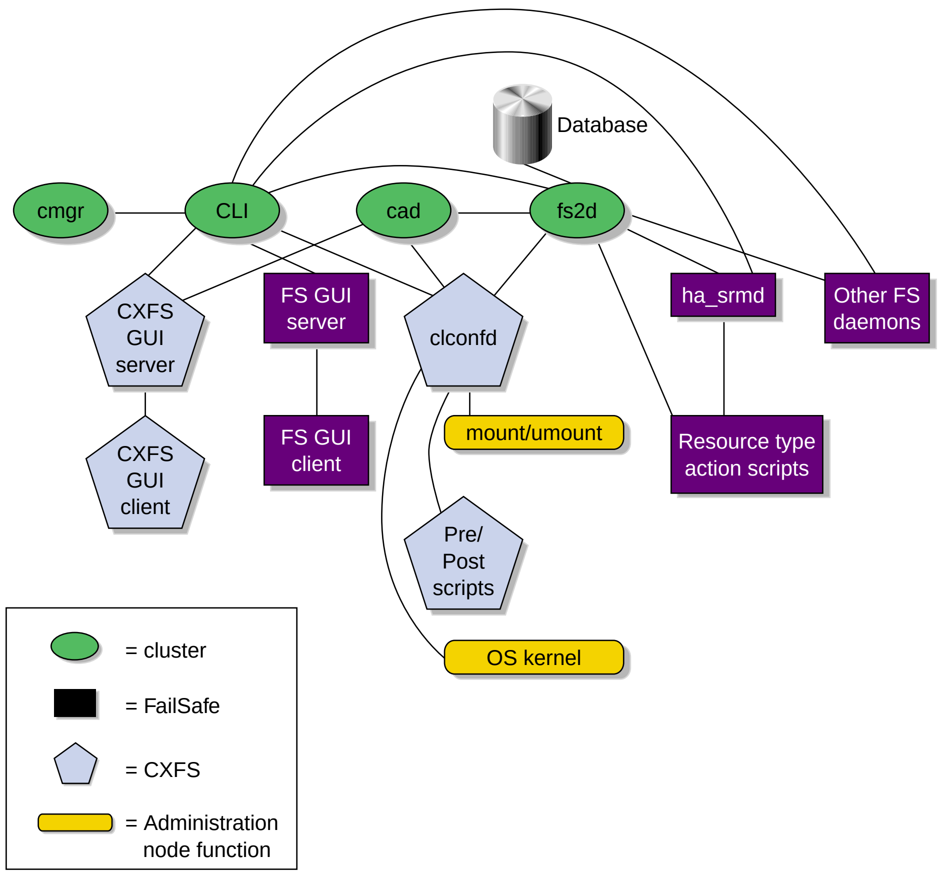


Figure A-6 Administrative Communication within One Administration Node under Coexecution

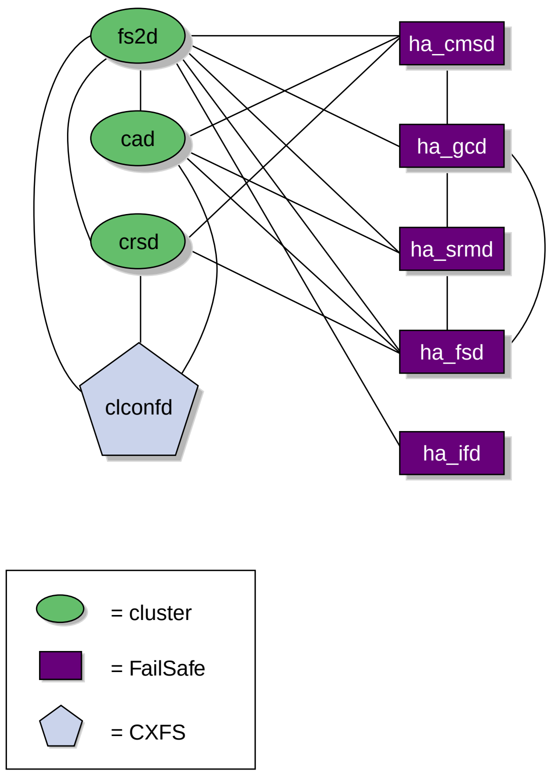


Figure A-7 Daemon Communication within One Administration Node under Coexecution

Flow of Metadata for Reads and Writes

The following figures show examples of metadata flow.

Note: A token protects a file. There can be multiple read tokens for a file at any given time, but only one write token.

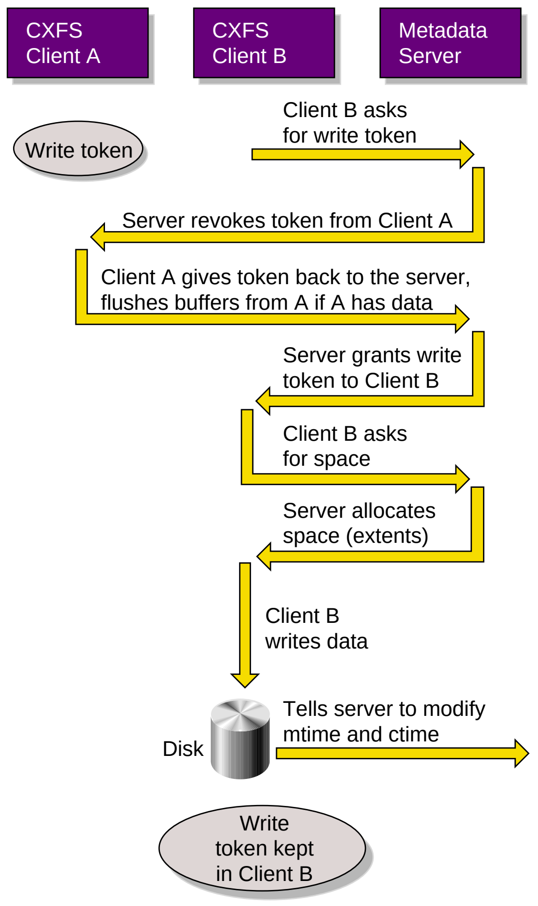


Figure A-8 Metadata Flow on a Write

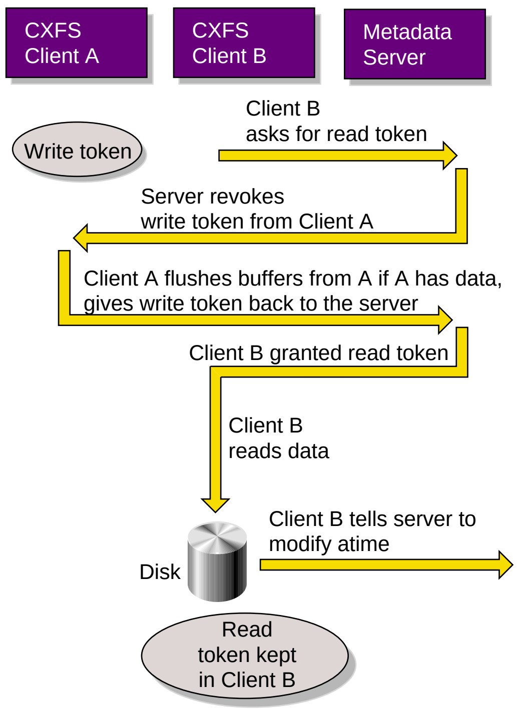

CXFS Client A
CXFS Client B
Metadata Server
Write token
Client B asks for read token
Server revokes write token from Client A
Client A flushes buffers from A if A has data, gives write token back to the server
Client B granted read token
Client B reads data
Client B tells server to modify atime
Disk
Read token kept in Client B

Figure A-9 Metadata Flow on a Read on Client B Following a Write on Client A

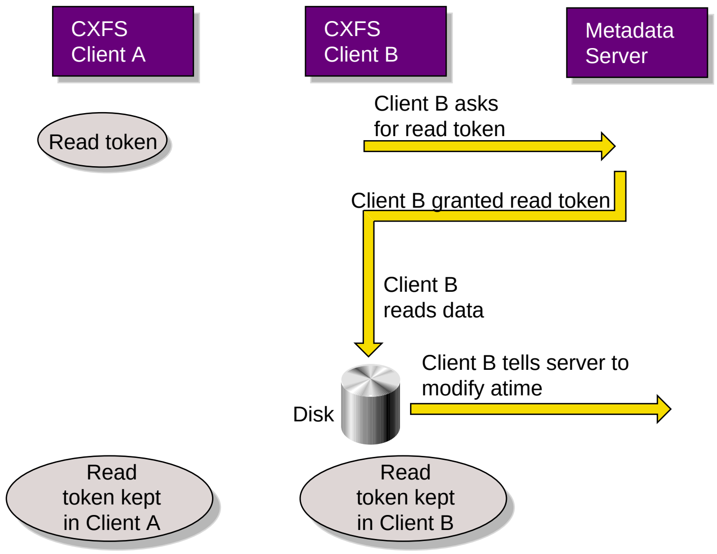

CXFS Client A
CXFS Client B
Metadata Server
Read token
Client B asks for read token
Client B granted read token
Client B reads data
Client B tells server to modify atime
Disk
Read token kept in Client A
Read token kept in Client B

Figure A-10 Metadata Flow on a Read on Client B Following a Read on Client A

Memberships and Quorums

The nodes in a FailSafe or CXFS cluster must act together to provide a service. To act in a coordinated fashion, each node must know about all the other nodes currently active and providing the service. The set of nodes that are currently working together to provide a service is called a *membership*. Cluster activity is coordinated by a configuration database that is replicated or at least accessible on all nodes in the cluster. The cluster software sends heartbeat messages between the nodes to indicate that a node is up and running. Heartbeat messages for each membership type are exchanged via a private network so that each node can verify each membership.

Nodes within the cluster must have the correct memberships in order to provide services. This appendix discusses the different types of membership and the effect they have on the operation of your cluster.

Nodes might not be able to communicate for reasons such as the following:

- They are down
- The communication daemons have failed or have been turned off
- Software has not been configured, or has been misconfigured
- The network is misconfigured (in this case, some heartbeat messages may fail while others succeed)
- The network router or cable fails (in this case, all heartbeat messages will fail)

Nodes that cannot communicate must be excluded from the membership because the other nodes will not be able to verify their status.

It is critical that only one membership of each type exist at any one time, as confusion and corruption will result if two sets of nodes operate simultaneously but independently. There is a risk of this happening whenever a segmentation of the private network occurs, or any other network problem occurs that causes the nodes eligible for membership to be divided into two or more sets, where the nodes in each set can communicate with themselves, but not with nodes outside of the set. Thus, in order to form a membership, the nodes must have a quorum, the minimum number of nodes required to form a membership. The quorum is typically set at half the total eligible members.

For example, consider the case of six nodes eligible for a membership:

- If all six nodes can communicate with each other, they will form a membership of six and begin offering the membership’s services.
- If a network segmentation occurs that causes four nodes to be in one set and two in another set, the two-node set will try to form its own membership but will be unable to do so because it does not have enough nodes to form a quorum; these nodes will therefore stop offering services. The four-node set will be able to form a new membership of four nodes and will continue to offer the membership’s services.
- If a network segmentation occurs that divides the nodes into three sets of two nodes each, no set will be able to form a membership because none contains enough nodes to form a quorum. In this case, the membership services will be unavailable; this situation is unavoidable, as each set of two nodes thinks that the four other nodes may have formed a quorum, and so no set may safely offer the membership’s services.
- If a network segmentation occurs that divides the nodes into two sets of three, then both could have a quorum, which could cause problems. To prevent this situation from occurring, some memberships may require a majority (>50%) of nodes or a tiebreaker node to form or maintain a membership. Tiebreaker nodes are used when exactly half of the server-capable administration nodes can communicate with each other.

The following sections provide more information about the specific requirements for membership.

Note: Because the nodes are unable to distinguish between a network segmentation and the failure of one or more nodes, the quorum must always be met, regardless of whether a partition has actually occurred or not.

Membership Types

There are three types of membership:

- "Cluster Database Membership and Quorum" on page 461
- "CXFS Kernel Membership, Quorum, and Tiebreaker" on page 462

- "FailSafe Membership, Quorum, and Tiebreaker" on page 464

Each provides a different service using a different heartbeat. Nodes are usually part of more than one membership.

Cluster Database Membership and Quorum

The nodes that are part of the the cluster database membership (also known as `fs2d` *membership*) work together to coordinate configuration changes to the cluster database:

- The *potential* cluster database membership is all of the administration nodes (installed with `cluster_admin` and running `fs2d`) that are defined using the GUI or the cmgr command as nodes in the pool. (CXFS client-only nodes are not eligible for cluster database membership.)
- The *actual* membership is the subset of eligible nodes that are up and running and accessible to each other, as determined by heartbeats on the private network. If the primary private network is unavailable, the cluster database heartbeat will failover to the next available heartbeat network defined for the node, if any.

The cluster database heartbeat messages use remote procedure calls (RPCs). Heartbeats are performed among all nodes in the pool. You cannot change the heartbeat timeout or interval.

If a node loses its cluster database membership, the cluster database write-operations from the node will fail; therefore, FailSafe and CXFS configuration changes cannot be made from that node.

The *cluster database membership quorum* ensures atomic write-operations to the cluster database that `fs2d` replicates in all administration nodes in the pool.

The cluster database membership quorum allows an initial membership to be formed when at least half (>=50%) of the eligible members are present. If there is a difference in the membership log between members, the cluster database tiebreaker node is used to determine which database is replicated. (See "Cluster Database Membership Logs" on page 465.) The cluster database tiebreaker node is always the administration node in the membership with the lowest node ID; you cannot reconfigure the tiebreaker for cluster database membership.

When the quorum is lost, the cluster database cannot be updated. This means that FailSafe and CXFS configuration changes cannot be made; although FailSafe and CXFS may continue to run, the loss of the cluster database quorum usually results in the loss of quorum for FailSafe and/or CXFS, because the nodes that drop from the cluster database membership will probably also drop from other memberships.

CXFS Kernel Membership, Quorum, and Tiebreaker

The nodes that are part of the CXFS kernel membership can share CXFS filesystems:

- The *potential* CXFS kernel membership is the group of all CXFS nodes defined in the cluster and on which CXFS services have been enabled. Nodes are enabled when CXFS services are started. The enabled status is stored in the cluster database; if an enabled node goes down, its status will remain enabled to indicate that it is supposed to be in the membership.
- The *actual* membership consists of the eligible nodes on which CXFS services have been enabled and that are communicating with other nodes using the heartbeat/control network. CXFS supports only one private network, and that network is the only network used for CXFS kernel membership heartbeats (but remember that the CXFS nodes may use multiple networks for the cluster database membership heartbeats).

  Note: CXFS metadata also uses the private network. The multiple heartbeats on the private network therefore reduce the bandwidth available for CXFS metadata.

  During the boot process, a CXFS node applies for CXFS kernel membership. Once accepted, the node can actively share the filesystems in the cluster.

The CXFS heartbeat uses multicast. Heartbeats are performed among all CXFS-enabled nodes in the cluster.

If a node loses its CXFS kernel membership, it can no longer share CXFS filesystems.

The *CXFS kernel membership quorum* ensures that only one metadata server is writing the metadata portion of the CXFS filesystem over the storage area network:

- For the *initial* CXFS kernel membership quorum, a majority (>50%) of the server-capable administration nodes with CXFS services enabled must be available to form a membership. (*Server-capable administration* nodes are those that are installed with the `cluster_admin` product and are also defined with the GUI or `cmgr` as capable of serving metadata. Client administration nodes are those that are installed with the `cluster_admin` product but are not defined as server-capable.)

Note: Client administration nodes and client-only nodes can be part of the CXFS kernel membership, but they are not considered when forming a CXFS kernel membership quorum. Only server-capable nodes are counted when forming the quorum.

- To *maintain* the existing CXFS kernel membership quorum requires at least half (50%) of the server-capable nodes that are eligible for membership. If CXFS kernel quorum is lost, the shared CXFS filesystems are no longer available.

No matter what the cluster components are, SGI recommends a system reset configuration on potential metadata servers to protect data integrity and improve server reliability. I/O fencing (or system reset when available) must be used on client-only nodes. See "Use a Client-Only Tiebreaker" on page 116. In clusters with an even number of potential metadata servers, a tiebreaker should be used in addition to I/O fencing or system reset; see "Isolating Failed Nodes" on page 27.

You can set the CXFS tiebreaker node by using the GUI's **Set Tiebreaker Node** task or by using the `modify` command in `cmgr`. See "Set Tiebreaker Node with the GUI" on page 192 and "Set the Tiebreaker Node with `cmgr`" on page 255.

Note: If one of the server-capable nodes is the CXFS tiebreaker in a two server-capable cluster, failure of that node or stopping the CXFS services on that node will result in a cluster-wide forced shutdown. SGI recommends making a client administration or client-only node the tiebreaker to avoid losing the cluster if the tiebreaker node fails.

If I/O fencing or reset is used, the quorum is maintained by whichever side wins the reset/fence race.

If a tiebreaker node is set and the network being used for heartbeat/control is divided in half, only the group that has the CXFS tiebreaker node will remain in the CXFS kernel membership. Nodes on any portion of the heartbeat/control network that are not in the group with the tiebreaker node will exit from the membership. Therefore, if the heartbeat/control network is cut in half, you will not have an active metadata server on each half of the heartbeat/control network trying to access the same CXFS metadata over the storage area network at the same time.

Note: A tiebreaker node must be configured individually for CXFS and for FailSafe. In a coexecution cluster, these could be different nodes.

FailSafe Membership, Quorum, and Tiebreaker

The nodes that are part of the FailSafe membership provide highly available (HA) resources for the cluster:

- The *potential* FailSafe membership is the set of all FailSafe nodes that are defined in the cluster and on which HA services have been enabled. Nodes are enabled when HA services are started. The enabled status is stored in the cluster database; if an enabled node goes down, its status will remain enabled to indicate that it is supposed to be in the membership.
- The *actual* membership consists of the eligible nodes whose state is known and that are communicating with other FailSafe nodes using heartbeat and control networks. If the primary private network is unavailable, the FailSafe heartbeat will failover to the next available heartbeat network defined for the node.

The FailSafe heartbeat uses user datagram protocol (UDP). Heartbeats are performed among all FailSafe-enabled nodes in the cluster. You can change the FailSafe heartbeat timing with the GUI Set FailSafe HA Parameters task or the `cmgr` command `modify ha_parameters` (the `node_timeout` parameter is the heartbeat timeout and the heartbeat is the heartbeat interval).

If a node loses its FailSafe membership, FailSafe will fail over its HA resources to another node in the cluster.

The *FailSafe membership quorum* ensures that a FailSafe resource is available only on one node in the cluster. The quorum requires that the state of a majority (>50%) of eligible nodes to be known and that half (50%) of the eligible nodes be present to form or maintain membership.

If a network partition results in a tied membership, in which there are two sets of nodes (each consisting of 50% of the potential FailSafe membership), then a node from the set containing the *FailSafe tiebreaker node* will attempt to perform a reset on a node in the other set. *Reset* is the failure action that performs a system reset via a serial line connected to the system controller.

If the node can verify that the other node was reset, then the membership will continue on the set with the tiebreaker. However, containing the tiebreaker is not a guarantee of membership; for more information, see the *FailSafe Administrator's Guide for SGI InfiniteStorage*. The default FailSafe tiebreaker is the node with the lowest node ID in the cluster.

When FailSafe membership quorum is lost, the resources will continue to run but they are no longer highly available.

Cluster Database Membership Logs

Each `fs2d` daemon keeps a *membership log* that contains a history of each database change (write transaction), along with a list of nodes that were part of the membership when the write transaction was performed. All nodes that are part of the cluster database membership will have identical membership logs.

When a node is defined in the database, it must obtain a current copy of the cluster database and the membership log from a node that is already in the cluster database membership. The method used to choose which node's database is replicated follows a hierarchy:

1. If the membership logs in the pool share a common transaction history, but one log does not have the most recent transactions and is therefore incomplete, the database from a node that has the complete log will be chosen to be replicated.
2. If there are two different sets of membership logs, the database from the set with the most number of nodes will be chosen.
3. If there are two different sets of membership logs, and each set has an equal number of nodes, then the set containing the node with the lowest node ID will be chosen.

To ensure that the complete transaction history is maintained, do not make configuration changes on two different administration nodes in the pool simultaneously. You should connect the CXFS or FailSafe GUI to (or run the `cmgr` command on) a single administration node in the pool when making changes. However, you can use any node in the pool when requesting status or configuration information.

The following figures describe potential scenarios using the hierarchies.

Figure B-1 on page 467, shows:

- Time 1: An established pool of three administration nodes sharing heartbeats, with node IDs 1-3, represented by the node names N1-N3. The `fs2d` database tiebreaker node is the node in the membership with the lowest node ID. Each successive database write is identified by a letter in the membership log.
- Time 2: A new node, N4, is defined using `cmgr` or the GUI connected to node N1. Node N4 (node ID = 4) joins the pool. Its membership log is empty.
- Time 3: Because N1/N2/N3 have identical membership logs, the database is replicated from one of them. In this case, N2 is randomly chosen.

- Time 4: All nodes in the pool have identical membership logs.
- Time 5: A network partition occurs that isolates N1. Therefore, N1 can no longer receive database updates. Configuration changes are made by connecting the GUI to N2 (or running `cmgr` on node N2); this results in updates to the membership logs in N2, N3, and N4, but not to N1 because it is isolated.
- Time 6: The partition is resolved and N1 is no longer isolated. Because N2/N3/N4 have identical membership logs, and share the beginning history with N1, the database is replicated from one of them. N4 is chosen at random.
- Time 7: All nodes in the pool have identical membership logs.

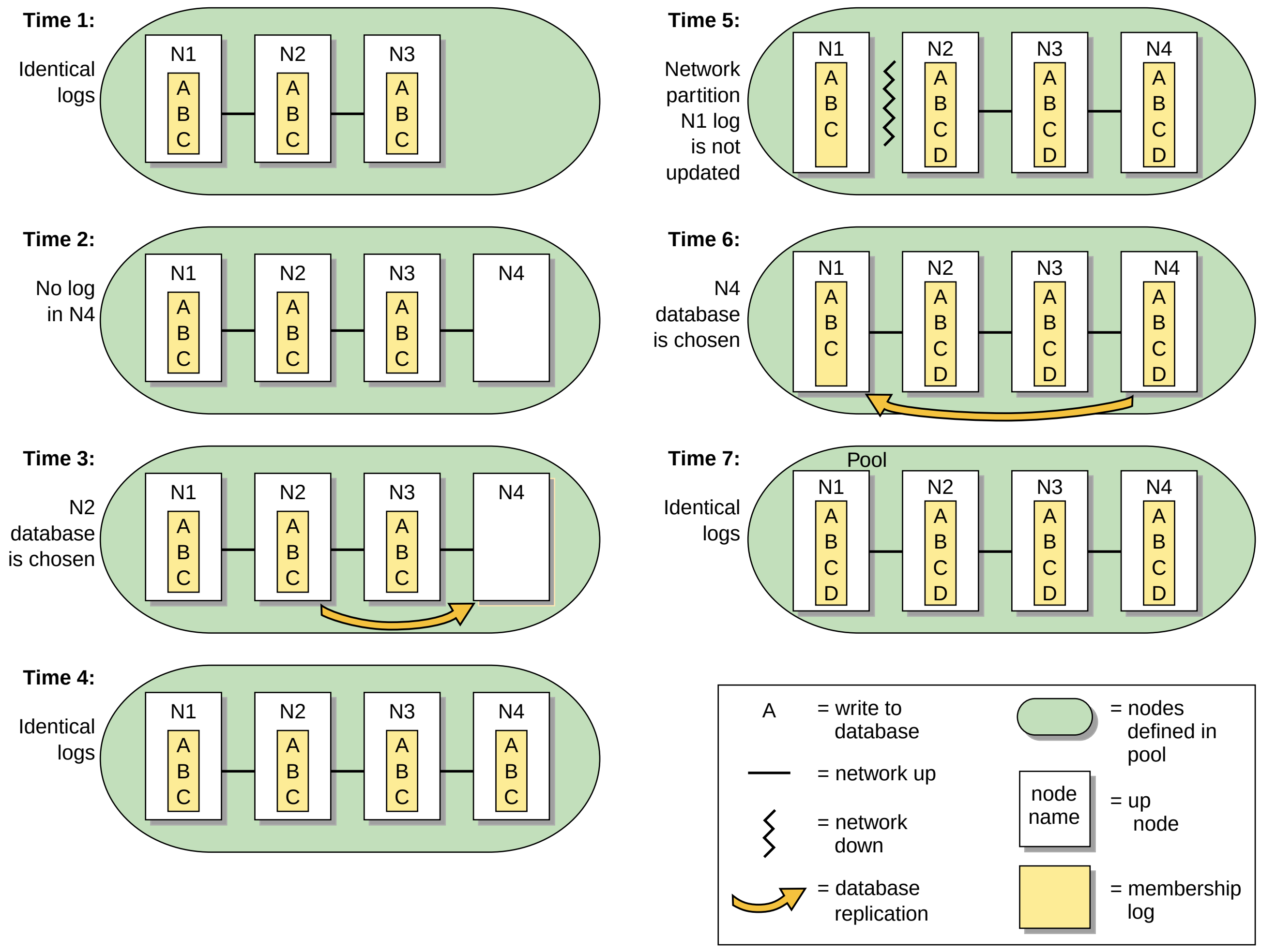


Figure B-1 One Node is Out of Date: Most Recent Log is Replicated

Recall that a node can be in only one pool at a time. If there are two separate pools, and from a node in one pool you define one or more nodes that are already in the other pool, the result will be that nodes from one of the pools will move into the other pool. **This operation is not recommended**, and determining which nodes will move into which other pool can be difficult. Figure B-2 on page 468 illustrates what to expect in this situation.

- Time 1: There are two pools that do not share membership log contents. One pool has two nodes (N1/N2), the other has three (N3/N4/N5).
- Time 2: N1 and N2 are defined as part of the second pool by running `cmgr` or connecting the GUI to node N3, N4, or N5. This results in a new pool with five nodes with different membership logs.
- Time 3: The database from the larger set of nodes is the one that must be replicated. N3 is chosen at random from the N3/N4/N5 set.
- Time 4: All nodes in the pool have identical membership logs.

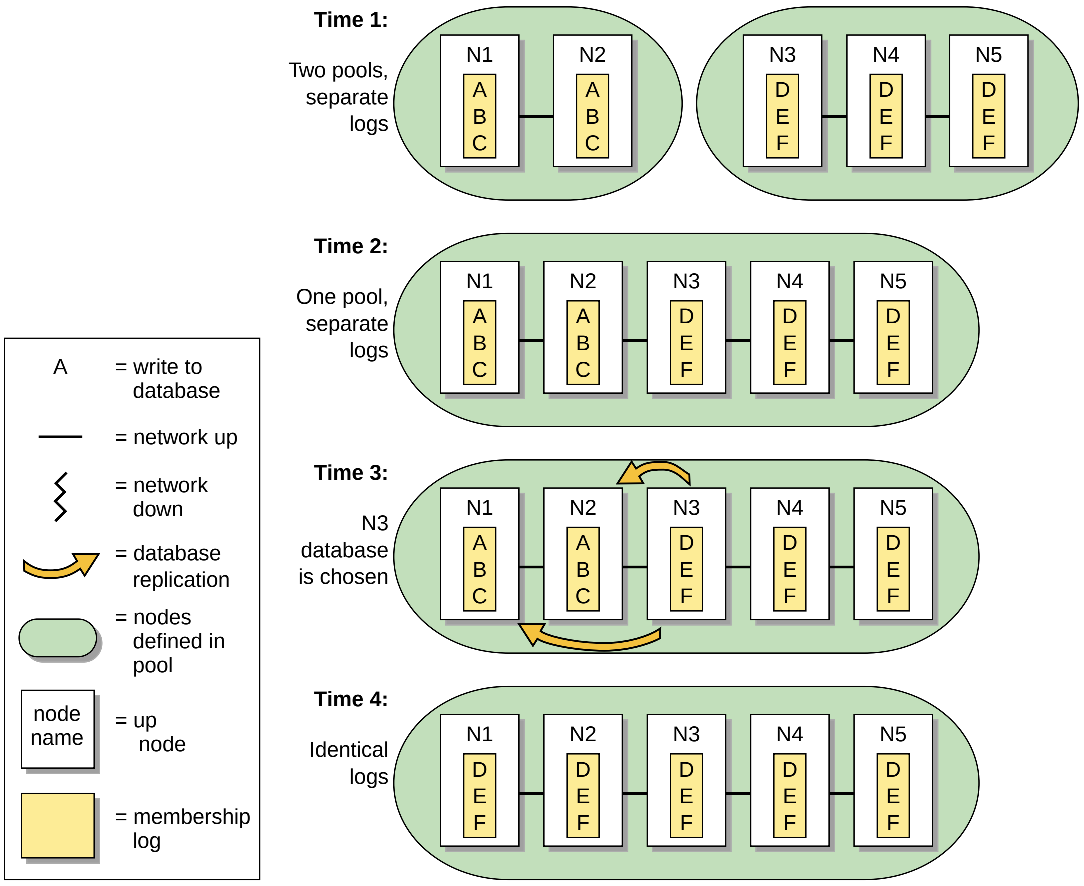


Figure B-2 Unequally Sized Pools are Joined: Log from Larger Pool is Replicated

Figure B-3 on page 470, shows a similar situation in which two nodes are defined in two pools, but the pools are of equal size:

- Time 1: There are two pools that do not share membership log contents. Each pool has two nodes (N1/N2 in pool 1, and N3/N4 in pool 2).
- Time 2: N1 and N2 are defined as part of the second pool by connecting the GUI or running cmgr on node N3 or N4. This results in a new pool with four nodes with different membership logs.
- Time 3: Because each set has the same number of nodes, the tiebreaker node (the node with the lowest node ID in the membership) must be used to determine whose database will be chosen. Because node N1 is the lowest node ID (node ID=1), the database from N1 is chosen.
- Time 4: All nodes in the pool have identical membership logs.

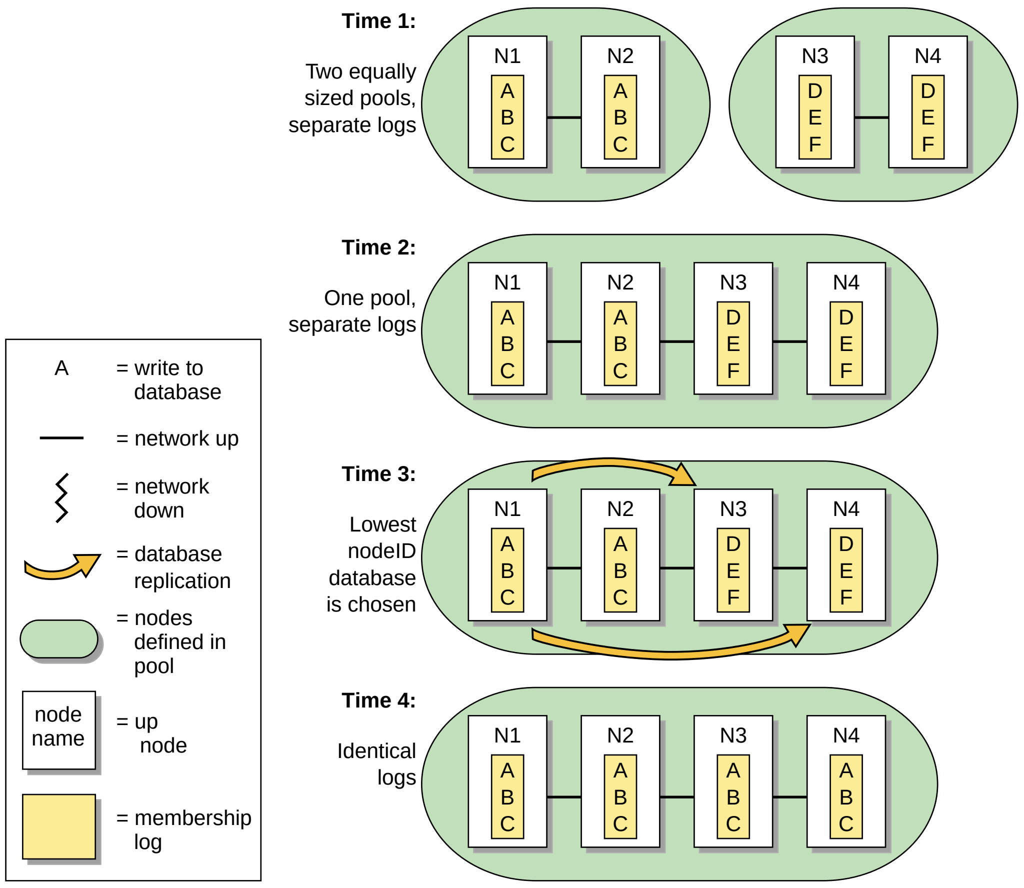


Figure B-3 Equally Sized Pools are Joined: Log from Node with Lowest Node ID is Replicated

Quorum and Tiebreaker Examples

Changing CXFS Kernel Membership Quorum Example

Figure B-4 on page 472, shows an example of a changing CXFS kernel membership quorum. It shows a pool of:

- Five CXFS server-capable administration nodes (A, B, C, D, and E)

- Two client-only nodes (F and G)
- One client admin node (H)

All nodes except E are defined as part of the cluster. Assume that CXFS services have been enabled on A, B, C, D, F, G, and H.

Of the seven nodes eligible for CXFS kernel membership, four are server-capable nodes (A, B, C, and D). Therefore, at least three of these four nodes must be able to communicate with each other to form an initial CXFS kernel quorum (>50% of the eligible server-capable nodes). Once the quorum has been reached, a membership will form with the nodes in the quorum plus all other eligible nodes that can communicate with the nodes in the quorum.

Figure B-4 on page 472, shows the following:

- Time 1: The CXFS kernel membership quorum is formed with three server-capable nodes, A, B, and C. The membership is A, B, C, F, G, and H.
- Time 2: Node B shuts down and leaves the membership. The remaining nodes in the quorum are A and C. The membership is still be available in this case because it satisfies the quorum requirement to maintain 50% of the eligible server-capable nodes (that is, two of the four server-capable nodes). The membership is A, C, F, G, and H.
- Time 3: Node A also shuts down and leaves the membership. Therefore, the quorum requirement is no longer met because quorum cannot be maintained with fewer than 50% of the eligible server-capable nodes. Without a quorum, the membership cannot continue, and so the CXFS filesystems in the cluster would not be available.

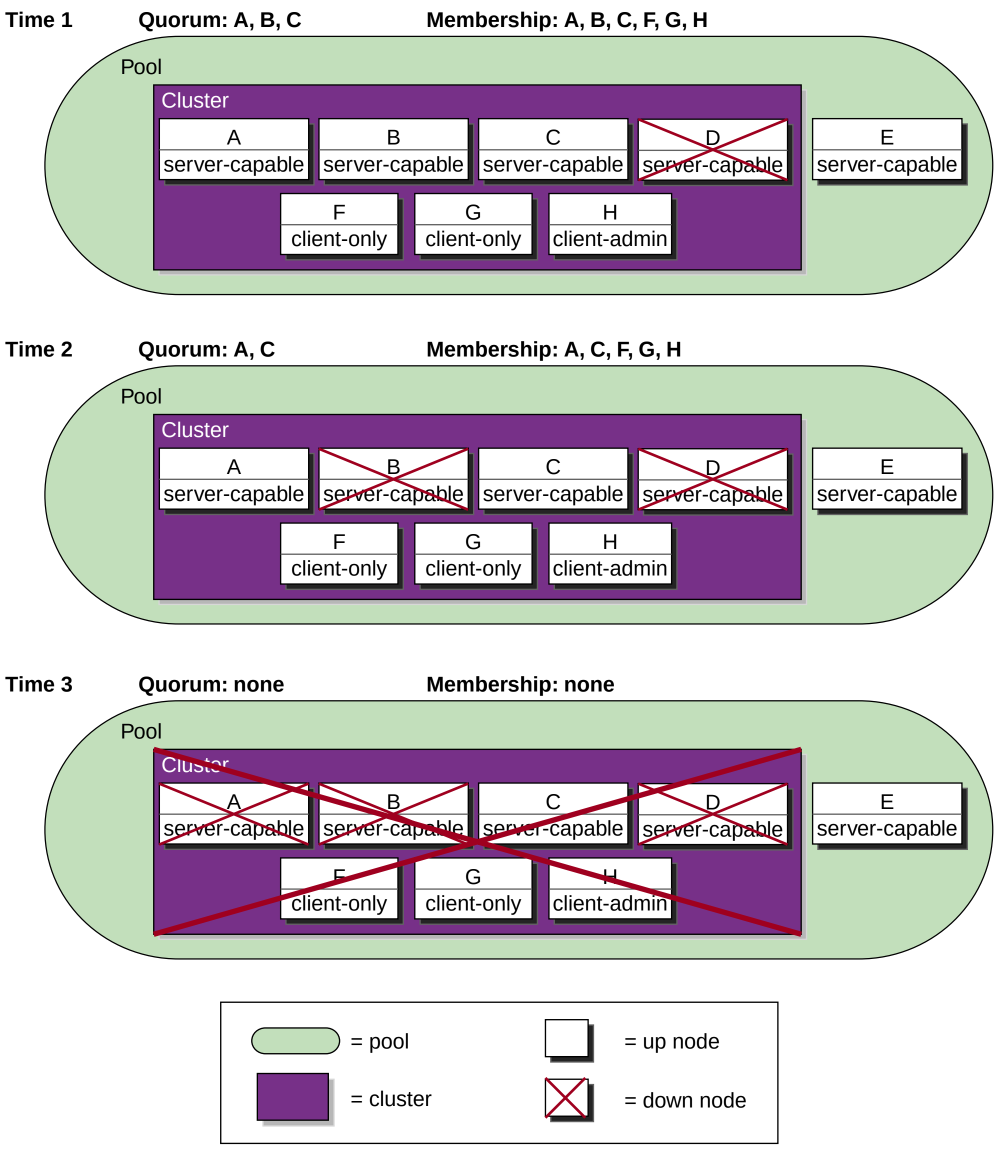


Figure B-4 Changing Quorum for CXFS Kernel Membership

Coexecution Example

Figure B-5 on page 474, shows an example of the different memberships in a cluster running CXFS and FailSafe. The pool contains 15 nodes (named N1 through N15). N1 has the lowest node ID number. There are CXFS nodes running IRIX, Solaris, and Windows; only the nodes running IRIX are administration nodes containing the cluster database. The FailSafe nodes are those where HA services are enabled; each of these is an administration node.

- Cluster database membership:
  - Eligible: N1, N2, N3, N5, N9, and N10 (that is, all nodes containing the cluster database)
  - Actual: N1, N2, N3, and N10 (because N5 and N9 are down)
  - Quorum: N1, N2, N3, and N10 (>50% of eligible nodes)
- FailSafe membership:
  - Eligible: N1, N2, and N3 (that is, those nodes with HA services enabled and defined as part of the cluster)
  - Actual: N1, N2, N3
  - Quorum: N1, N2, N3 (>50% of eligible nodes)
- CXFS kernel membership:
  - Eligible: N1-N8 and N11-N15 (N9 and N10 are not defined as part of the cluster)
  - Actual: N1, N2, N3, N4, N6, and N11-N15 (because N5, N7, and N8 are down)
  - Quorum: N1, N2 (>50% of server-capable eligible nodes)

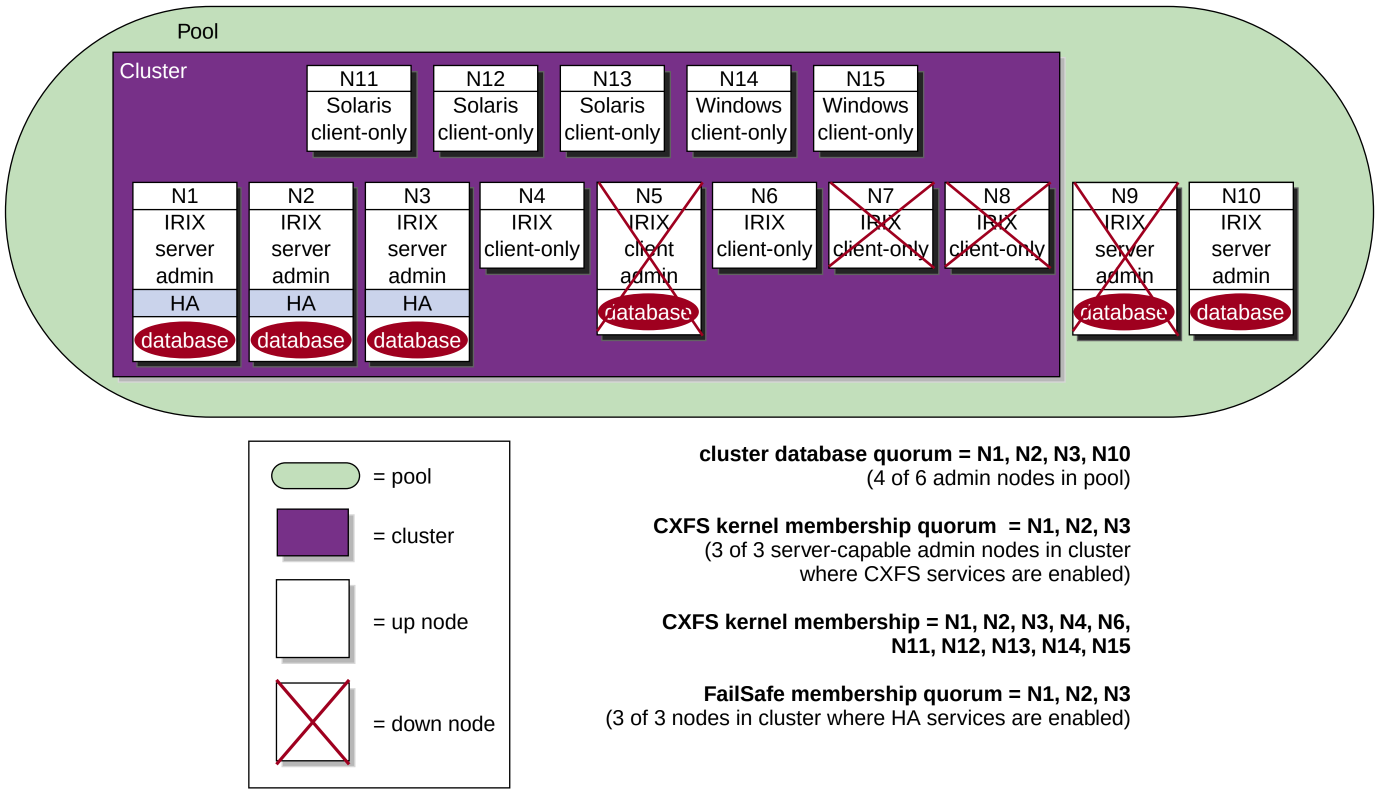

Pool
Cluster
N11 Solaris client-only
N12 Solaris client-only
N13 Solaris client-only
N14 Windows client-only
N15 Windows client-only
N1 IRIX server admin HA database
N2 IRIX server admin HA database
N3 IRIX server admin HA database
N4 IRIX client-only
N5 IRIX client admin database
N6 IRIX client-only
N7 IRIX client-only
N8 IRIX client-only
N9 IRIX server admin database
N10 IRIX server admin database
= pool
= cluster
= up node
= down node
cluster database quorum = N1, N2, N3, N10
(4 of 6 admin nodes in pool)
CXFS kernel membership quorum = N1, N2, N3
(3 of 3 server-capable admin nodes in cluster where CXFS services are enabled)
CXFS kernel membership = N1, N2, N3, N4, N6, N11, N12, N13, N14, N15
FailSafe membership quorum = N1, N2, N3
(3 of 3 nodes in cluster where HA services are enabled)

Figure B-5 Example Memberships in a Coexecution Cluster

CXFS Tiebreaker Node Example

Figure B-6 on page 475, displays a situation in which a router dies and the heartbeat/control network is effectively split in two. The potential CXFS kernel membership is defined to be nodes A, B, C, and D. The nodes on network segment 2 (nodes C and D) will leave the CXFS kernel membership because they do not contain the CXFS tiebreaker node, and therefore do not have a quorum. On network segment 1, one of the other two potential metadata servers will become active and the membership will only include the systems on network segment 1. The nodes that were on network segment 2 will remain out of the membership until CXFS services are restarted on them and the router is repaired.

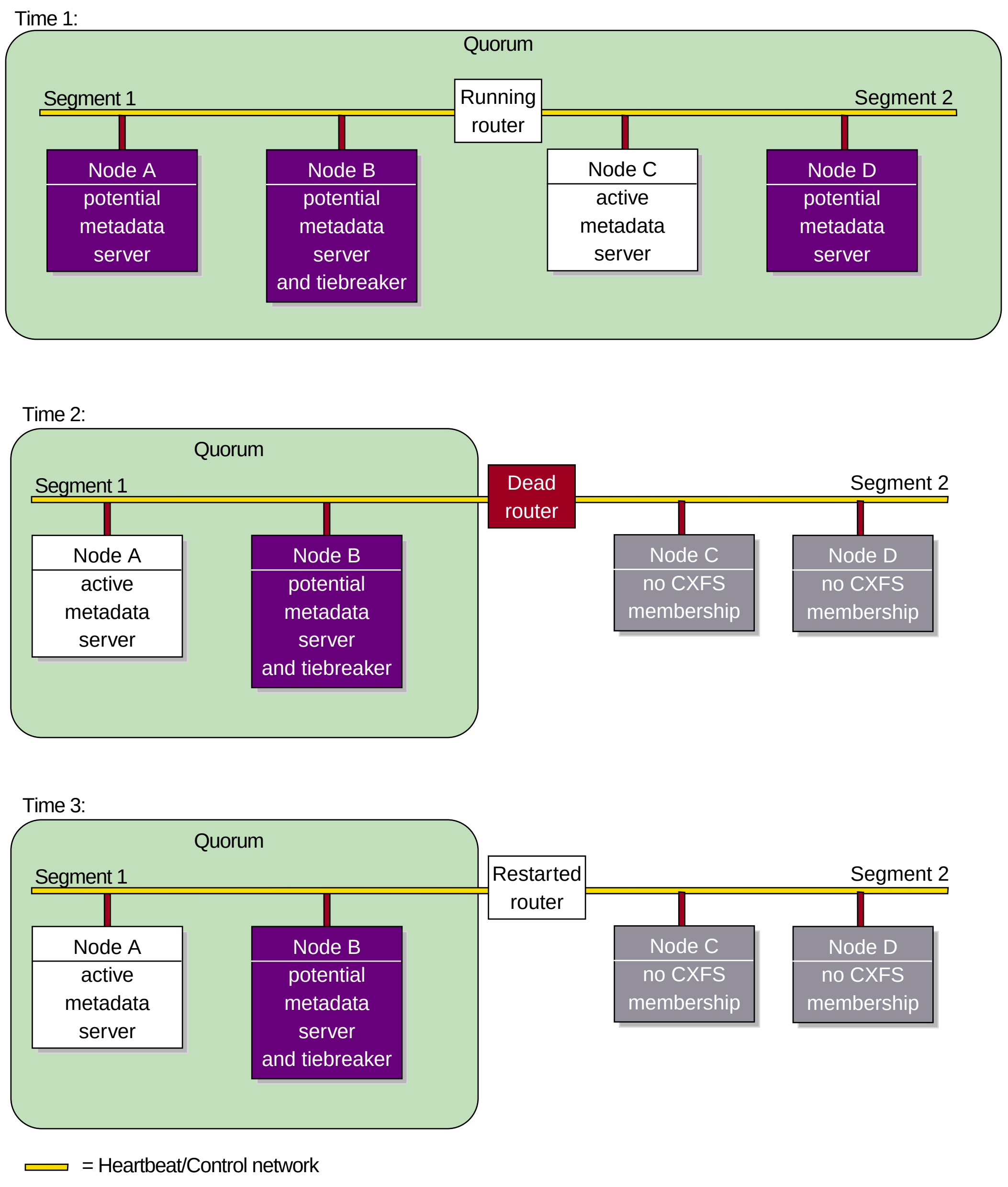

Time 1:
Quorum
Segment 1
Running router
Segment 2
Node A potential metadata server
Node B potential metadata server and tiebreaker
Node C active metadata server
Node D potential metadata server
Time 2:
Quorum
Segment 1
Dead router
Segment 2
Node A active metadata server
Node B potential metadata server and tiebreaker
Node C no CXFS membership
Node D no CXFS membership
Time 3:
Quorum
Segment 1
Restarted router
Segment 2
Node A active metadata server
Node B potential metadata server and tiebreaker
Node C no CXFS membership
Node D no CXFS membership
= Heartbeat/Control network

Figure B-6 CXFS Tiebreaker Node

Heartbeat Considerations

There are different heartbeats for each membership type, and each uses a different networking method. Therefore, certain network misconfiguration can cause one heartbeat to fail while another succeeds.

At least two networks should be designated as FailSafe heartbeat networks. FailSafe uses only the highest priority working network for heartbeats; the other network is for heartbeat failover. Usually the private network is used as the highest priority heartbeat network.

In a coexecution cluster, there must be two networks as required by FailSafe; at least one private network is recommended for FailSafe and a private network is required by CXFS.

In a coexecution cluster, CXFS meta data, CXFS heartbeat, and FailSafe heartbeat can use the same network. The heartbeat intervals and timeouts should be appropriately adjusted, if possible, so that all network traffic has sufficient bandwidth. You cannot change the heartbeat timeout or interval for the cluster database membership. Before you adjust the heartbeat settings for the FailSafe membership or CXFS kernel membership, you should consider the impact on the other heartbeats.

If the highest priority network fails, the FailSafe, cluster and CXFS kernel memberships will continue using the next priority network. (However, if private network failover has not been defined for CXFS, then the CXFS kernel membership will fail.)

CXFS Recovery Issues in a Cluster with Only Two Server-Capable Nodes

A cluster with an odd number of server-capable nodes is recommended for a production environment. However, if you use a production cluster with an even number of server-capable nodes (especially only two server-capable nodes), you must do one of the following:

- Use system reset configuration on potential metadata servers to protect data integrity and improve server reliability. I/O fencing (or system reset when available) must be used on client-only nodes. Clusters should have an odd number of server-capable nodes or use a client-only tiebreaker node.
- Set a CXFS tiebreaker node. If the tiebreaker node is a server-capable administration node, there will be a loss of CXFS kernel membership, and

therefore CXFS filesystems, if the tiebreaker node goes down. If the tiebreaker is an administration node, the cluster database membership may also be lost.

However, even with these methods, there are recovery and relocation issues inherent to a cluster with only two server-capable nodes.

IP Filtering Example for the CXFS Private Network

This appendix contains an example `/etc/ipfilterd.conf` file that can be used to provide IP filtering for the CXFS private network.

Note the following:

- If you use I/O fencing and `ipfilterd` on a node, the `ipfilterd` configuration must allow communication between the node and the `telnet` port on the switch.
- There must be an `/etc/ipfilterd.conf` file configured on each node on which you want to filter IP traffic. The files will be similar except for the first set of lines, which are node-dependent; that is, the lines in the file for NodeA must match the networking interfaces on which the network traffic may pass for NodeA.
- The `systune` variable `ipfilterd_inactive_behavior` must be set to 0, which means that the filter will be disabled as soon as `ipfilterd` is terminated using the `killall` command.
- The `ipfilterd` argument to `chkconfig` must be turned on for each node where `ipfilterd` will run. For example:

  ```
  nodeA# chkconfig ipfilterd on
  ```

- If any network interface name is changed on a system, you must update the `/etc/ipfilterd.conf` file to include the change in the appropriate `accept` line. That is:

  accept -i *changed_or_new_interface*

- For debugging purposes, each dropped packet will log a message similar to the following in the `syslog` file:

```
May 24 16:44:44 5A:rodin unix: NOTICE: ipfilter(cache) - packet dropped:
10.1.1.5 SPT=137 DPT=137 UDP
```

If you want to disable the filtering, such as in the case where it is blocking wanted traffic, do the following:

1. Kill the `ipfilterd` daemon:

   ```
   nodeA# killall ipfilterd
   ```

2. Turn off the `ipfilterflag` argument:

   ```
   nodeA# chkconfig ipfilterd off
   ```

Following is a sample file for nodeA:

```
nodeA# cat ipfilterd.conf
#
# ipfilterd.conf for nodeA
#
#
# Filters follow:
#
# Do not restrict traffic on any of the interfaces for NodeA,
# except from ef1 (CXFS heartbeat)
#
accept -i lo0
accept -i ef0
accept -i eg0
accept -i eg1
accept -i lb0

#
# Restrict access over the CXFS heartbeat network
# Interface ef1
#

# Accept any fragment, reassembly won’t work if first fragment filtered out.
accept -i ef1 ip.off>0

# CXFS is using RPC, need portmapper.
accept -i ef1 udp.port 111
accept -i ef1 tcp.port 111


# fs2d daemon is dynamically assigning ports in range 600-1023.
# We need port definition (sport + dport for both directions).
accept -i ef1 tcp.sport>=600 and tcp.sport<=1023
accept -i ef1 tcp.dport>=600 and tcp.dport<=1023


# sgi-cad defaults to 5435/tcp
```

```
accept -i ef1 tcp.port 5435

# sgi-crsd
# Each node opens 7500/udp, both directions needed
accept -i ef1 udp.port 7500

# Uncomment the line below for CXFS client-only node.
# accept -i ef1 udp.port 5449


# CXFS kernel ports 5450-5453
# Connections in both directions so open dport and sport.
accept -i ef1 tcp.port 5450
accept -i ef1 tcp.port 5451
accept -i ef1 udp.port 5452
accept -i ef1 udp.port 5453

# fs2d client are using ports in range 7000-8500
accept -i ef1 tcp.dport>7000
accept -i ef1 udp.dport>7000

# Uncomment the line below for IO fencing only if switches are on CXFS private network
#  (ip.src is the switch address)
# accept -i ef1 tcp.sport=23 and ip.src=10.1.1.6

# Let icmp traffic pass, especially 'PORT UNREACHABLE ICMP packet'
accept -i ef1 icmp

# Reject the rest (-l will log any rejected packet to the SYSLOG)
reject -i ef1 -l
```

Operating System Path Differences

This appendix lists the locations for commonly used commands. For information about other client-only operating systems, see the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

Table D-1 IRIX Paths

Command/File	IRIX
`chkconfig`	`/etc/chkconfig`
`cxfs_client` log	`/var/adm/cxfs_client`
`cxfs_client.options`	`/etc/config/cxfs_client.options`
`df`	`/usr/sbin/df`
`grioadmin`	`/usr/sbin/grioadmin`
`grioqos`	`/usr/sbin/grioqos`
`hinv`	`/sbin/hinv`
`hostname`	`/usr/bsd/hostname`
`mount`	`/sbin/mount`
`netstat`	`/usr/etc/netstat`
`ping`	`/usr/etc/ping`
`ps`	`/usr/bin/ps`
`scsiha`	`/usr/sbin/scsiha`
`xvm`	`/sbin/xvm`
Cluster daemon configuration files	`/etc/config/`
System log	`/var/adm/SYSLOG`
CXFS/cluster daemon initialization	`/etc/init.d/cluster`

Table D-2 Linux Paths

Command/File	Linux
`chkconfig`	`/bin/chkconfig`
`cxfs_client` log	`/var/log/cxfs_client`
`cxfs_client.options`	`/etc/cluster/config/cxfs_client.options`
`df`	`/bin/df`
`grioadmin`	`/usr/sbin/grioadmin`
`grioqos`	`/usr/sbin/grioqos`
`hinv`	`/usr/bin/hinv`
`hostname`	`/bin/hostname`
`mount`	`/bin/mount`
`netstat`	`/bin/netstat`
`ping`	`/bin/ping`
`ps`	`/bin/ps`
`scsiha`	`/bin/scsiha`
`xvm`	`/sbin/xvm`
Cluster daemon configuration files	`/etc/cluster/config/`
System log	`/var/log/messages`
CXFS/cluster daemon initialization	`/etc/init.d/cxfs_cluster`

Filesystem Specifications

Item	IRIX	Linux
Maximum filesystem size	2^{64} bytes (about 18 million terabytes). The maximum offset within a CXFS file is (about	2^{64} bytes
Maximum files size/offset	2^{63}-1 bytes (about 9 million terabytes)	2^{63}-1 bytes
Block size (in bytes)[1]	512, 1024, 2048, 4096, 8192, 16384, 32768, or 65536	512, 1024, 2048, or 4096

[1] If the filesystem is to be accessible by other platforms in a multiOS cluster, its block size must be supported on all platforms in the cluster

System Reset Configuration

This appendix discusses system controllers that can be used in CXFS system reset configurations:

- "L2 System Controller"
- "L1 System Controller" on page 492
- "MSC System Controller" on page 494
- "MMSC System Controller" on page 497

Note: Serial cables are provided with SAN server configurations. Other configurations require that you purchase serial cables if you want to use system reset.

L2 System Controller

The L2 system controller and the required USB cables are optional equipment available for purchase. The L2 method is recommended because the system console remains available.

Use the modem port on the L2 system controller as shown in Figure F-2. Use DB9 serial ports on an IX-brick on Altix 3000 and Origin 3000. Connect the serial cable to the modem port on one end and the serial port on the IX-brick (for example, serial port connector 0), as shown in Figure F-3.

Figure F-4, Figure F-5, and Figure F-6 show serial connections for two machines with an L2 system controller. (These figure shows direct attached storage. Serial connections for other storage configurations will be the same.)

In Altix 350, use IO10 and a *multiport serial adapter cable*, which is a device that provides four DB9 serial ports from a 36-pin connector; see Figure F-1.

In Altix systems with an integrated L2 (such as a NUMAlink 4 R-brick) or SGI Altix 3000 Bx2 systems, use the L2 over Ethernet. See "Define a Node with `cmgr`" on page 224.

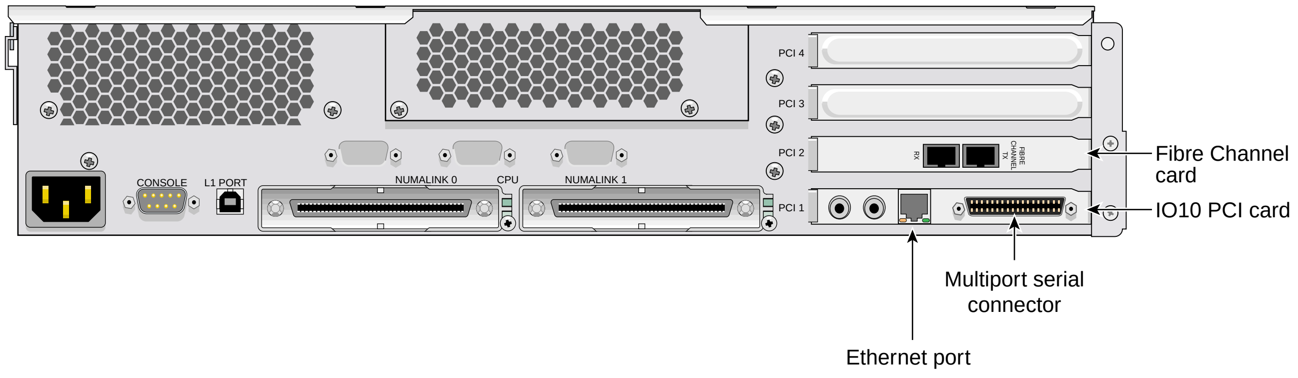


Figure F-1 Altix 350 Rear Panel

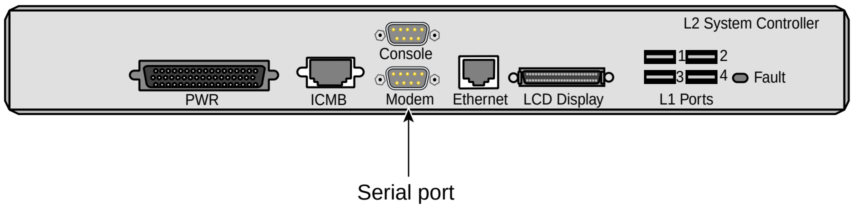


Figure F-2 L2 Rear Panel

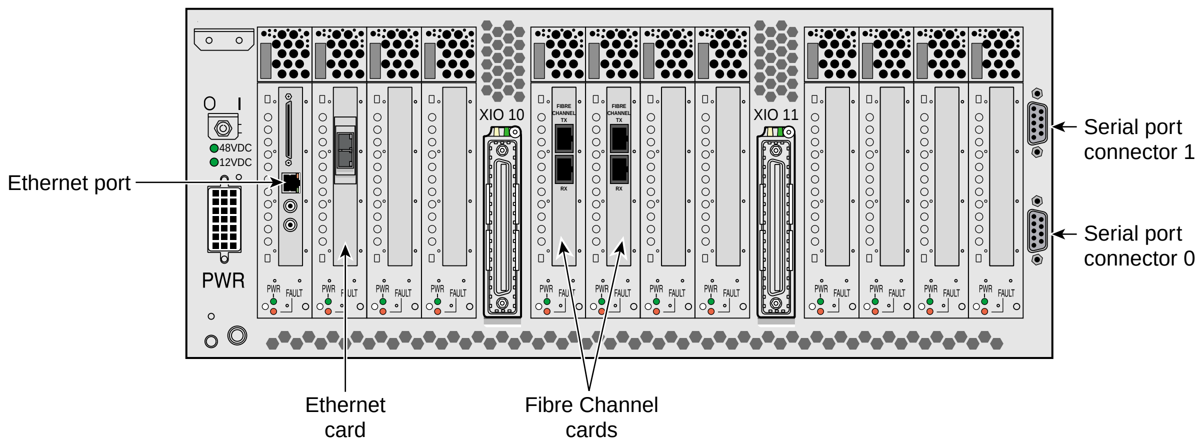


Figure F-3 IX-brick Rear Panel

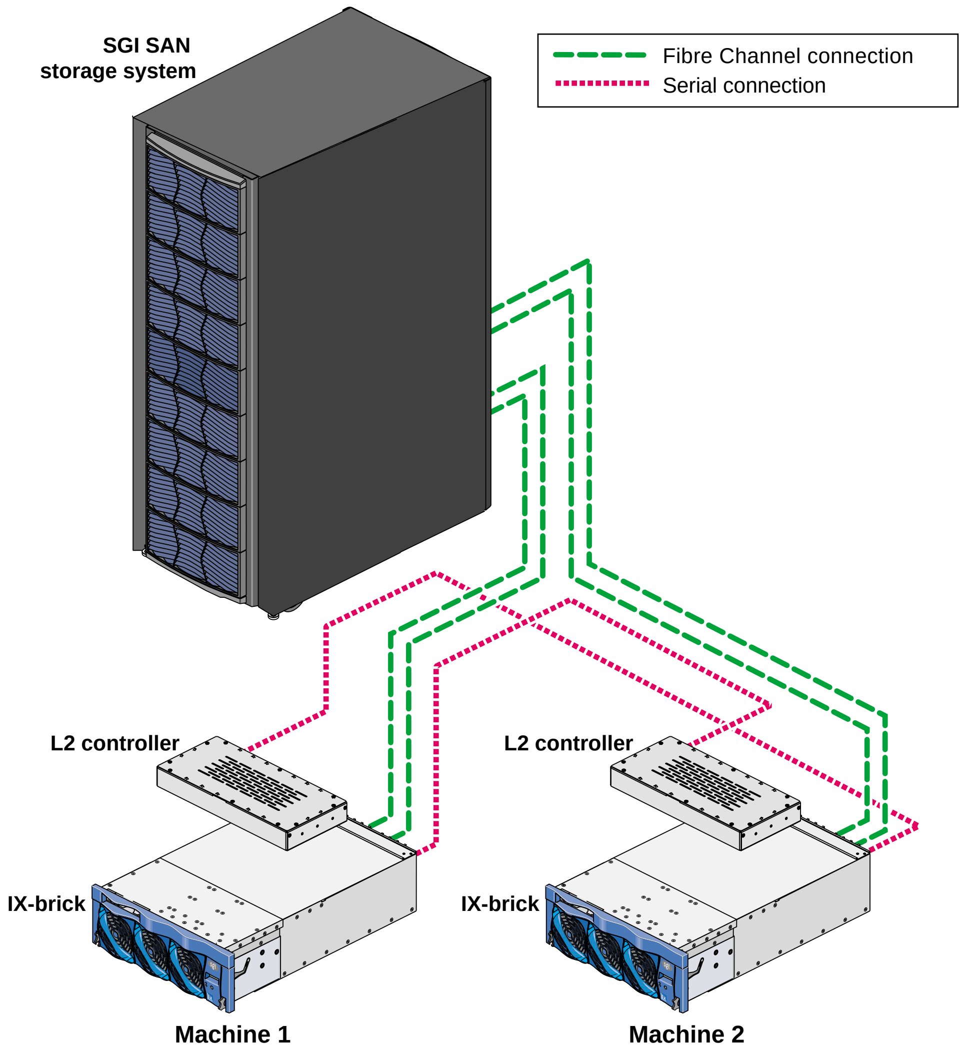


Figure F-4 Altix 3000 and Origin 3000 Serial Connections

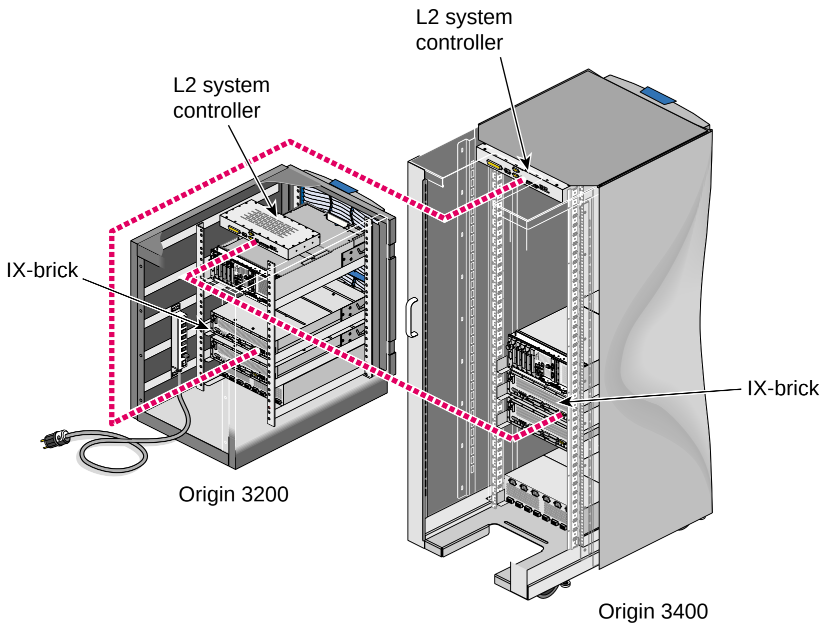


Figure F-5 Serial Connection Between SGI Origin 3200 and Origin 3400/3800 Servers

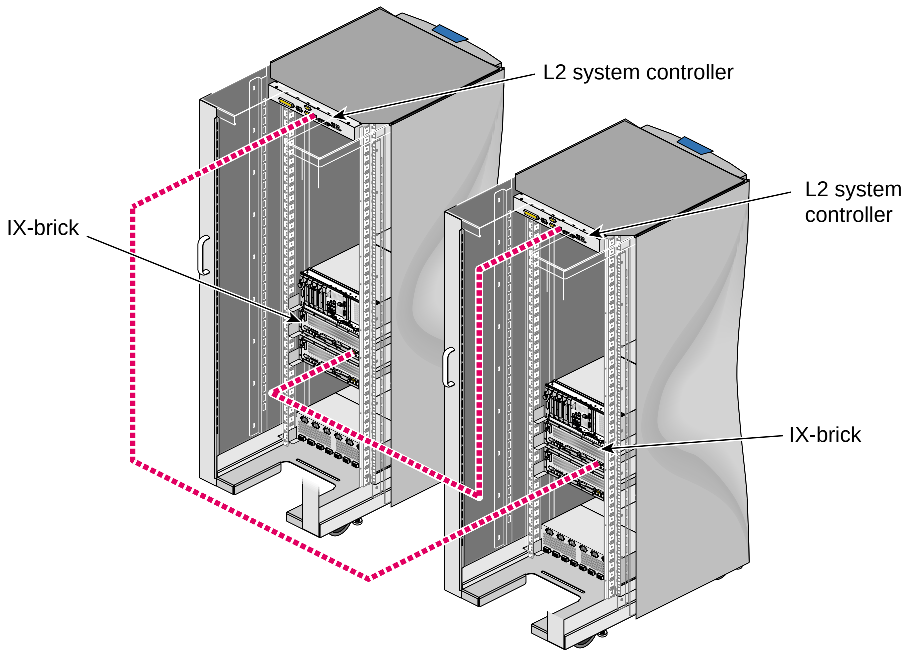


Figure F-6 Serial Connection Between Two SGI Origin 3400 or SGI Origin 3800 Servers

L1 System Controller

L1 system controller can be used for reset, however, SGI recommends the use of L2 because using L1 for system reset makes the L1 unavailable for system console).

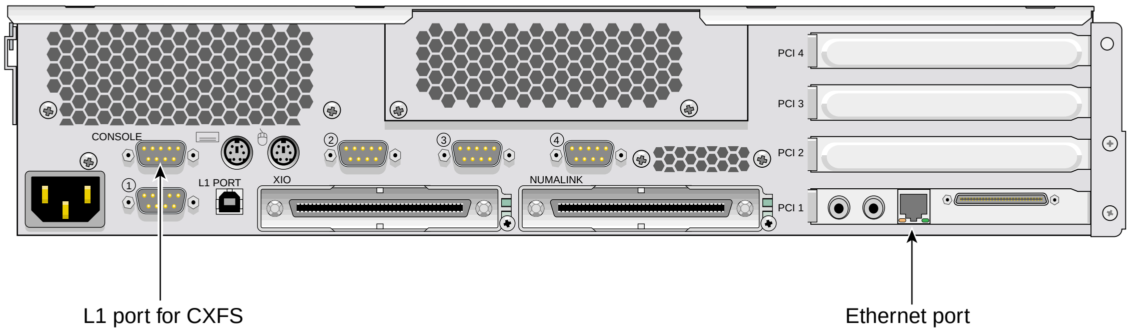


Figure F-7 Origin 350 Rear Panel

Connect the serial cable to the console port (port labeled **CONSOLE**) on one end and the serial port of other node on the other end. The serial ports on Origin 350 are labeled as **1**, **2**, **3**, and **4**; see Figure F-7.

Note: The USB port on the Origin 350 is labeled **L1 PORT**. Do not use this port for system reset. Use the port labeled **CONSOLE** as shown in Figure F-7.

Redirecting the Console for Origin 300, Origin 350, Origin 3200C, Onyx 300, Onyx 350, and Onyx 3200C

On Origin 300, Origin 350, Origin 3200C, Onyx 300, Onyx 350, and Onyx 3200C systems, there is only one serial/USB port that provides both L1 system controller and console support for the machine. In a CXFS configuration, this port (the DB9 connector) is used for system reset. It is connected to a serial port in another node or to the Ethernet multiplexer.

To get access to console input and output, you must redirect the console to another serial port in the machine.

Use the following procedure to redirect the console:

1. Edit the `/etc/inittab` file to use an alternate serial port.
2. Either issue an `init q` command or reboot.

For example, suppose you had the following in the `/etc/inittab` file (line breaks added for readability):

```
# on-board ports or on Challenge/Onyx MP machines, first IO4 board ports
t1:23:respawn:/sbin/suattr -C CAP_FOWNER,CAP_DEVICE_MGT,CAP_DAC_WRITE+ip
-c "exec /sbin/getty ttyd1 console"    # alt console
t2:23:off:/sbin/suattr -C CAP_FOWNER,CAP_DEVICE_MGT,CAP_DAC_WRITE+ip
-c "exec /sbin/getty -N ttyd2 co_9600"     # port 2
```

You could change it to the following:

```
# on-board ports or on Challenge/Onyx MP machines, first IO4 board ports
t1:23:off:/sbin/suattr -C CAP_FOWNER,CAP_DEVICE_MGT,CAP_DAC_WRITE+ip
-c "exec /sbin/getty ttyd1 co_9600"       # port 1
t2:23:respawn:/sbin/suattr -C CAP_FOWNER,CAP_DEVICE_MGT,CAP_DAC_WRITE+ip
-c "exec /sbin/getty -N ttyd2 console" # alt console
```

Caution: Redirecting the console by using the above method works only when the IRIX operating system is running. To access the console when the operating system is not running (miniroot) , you must physically reconnect the machine: unplug the serial hardware reset cable from the console/L1 port and then connect the console cable.

MSC System Controller

Figure F-8 and Figure F-9 show the serial connection between two deskside servers.

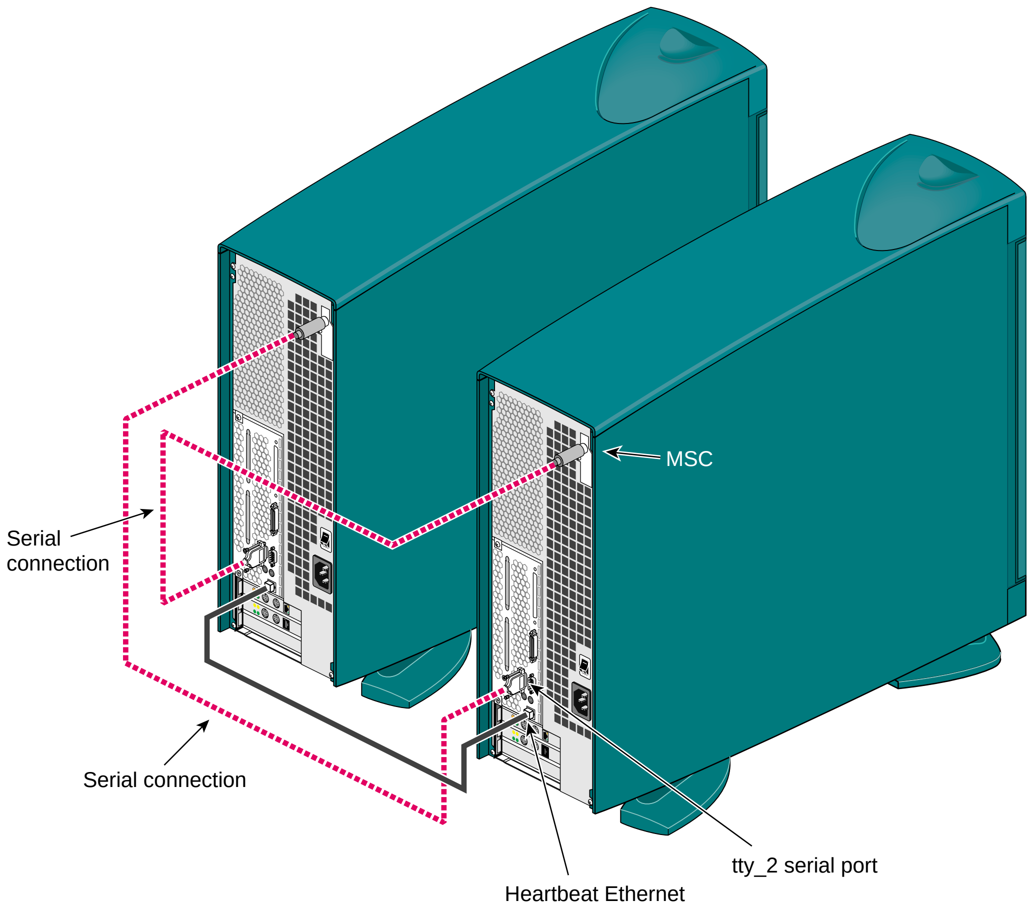


Figure F-8 Serial Connection Between Two Origin 200 Deskside Servers

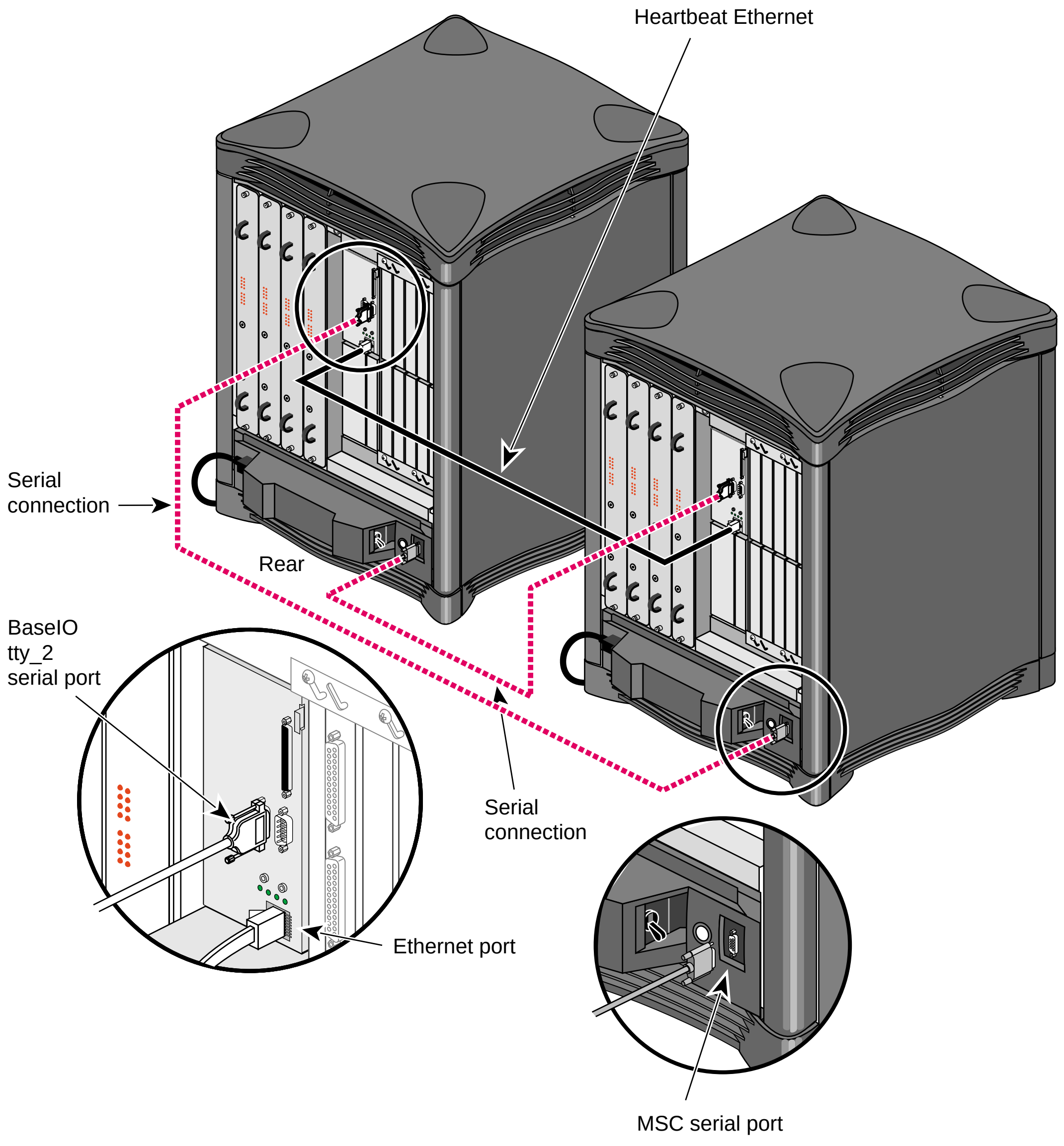


Figure F-9 Serial Connection Between Two SGI 2200 Deskside Servers

MMSC System Controller

Figure F-10 shows the MMSC. The alternate console port should be connected to the serial port on another machine using a serial cable.

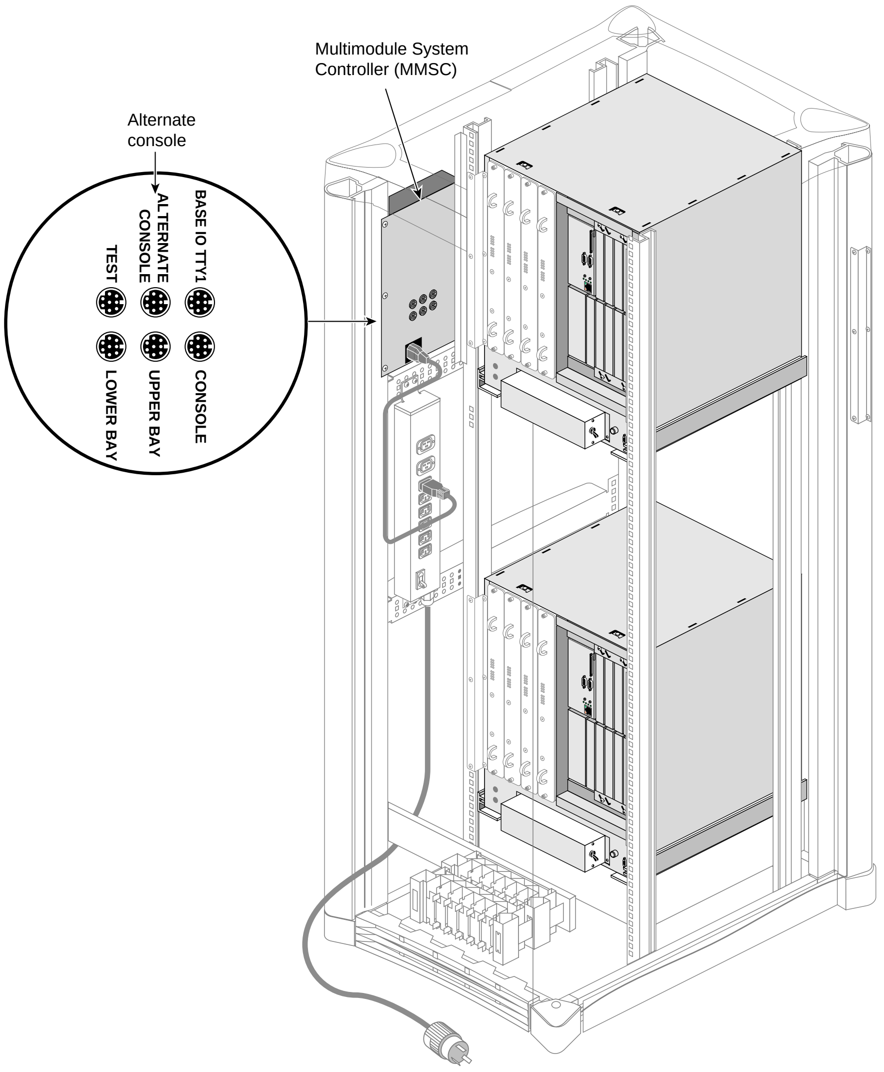


Figure F-10 MMSC Serial Port

Mount Options Support

The table in this appendix lists the mount options that are supported by CXFS, depending upon the server platform. Some of these mount options affect only server behavior and are ignored by client-only nodes.

The table also lists those options that are not supported, especially where that support varies from one platform to another. Both the IRIX and the SGI ProPack for Linux `mount` commands support many additional options, but these options may be silently ignored by the clients, or cause the mount to fail and should be avoided.

For more information, see the IRIX `mount`(1M) and Linux `mount`(8) man pages.

Note: The following are mandatory, internal CXFS mount options that cannot be modified and are set by `clconfd` and `cxfs_client`:

```
client_timeout
server_list
```

The table uses the following abbreviations:

MDS = Metadata server
S = Supported
n = Not supported
D = Determined by the CXFS administration tools (not user-configurable)

A blank space within the table means that the option has not been verified.

Table G-1 Mount Options Support

Mount Option	IRIX Client-only Node		ProPack for Linux Client-only Node	
	IRIX MDS	ProPack MDS	IRIX MDS	ProPack MDS
`biosize`	S	S	S	S
`client_timeout`	D	D	D	D
`dmapi`	n	n	n	n
`dmi`[1]	S	S	S	S
`gqnoenforce`	S	S	S	S
`gquota`	S	S	S	S
`grpid`	S	n	n	n
`inode64`	S	S	S	S
`logbufs`	S			
`noatime`	S	S	S	S
`noauto`	n	n	n	n
`nodev`	S	S	S	S
`noquota`	S	S	S	S
`nosuid`	S	S	S	S
`osyncisdsync`	S			
`pqnoenforce`	S			
`pquota`	S			
`qnoenforce`	S	S	S	S
`quota`	S	S	S	S
`ro`	S	S	S	S
`rw`	S	S	S	S
`server_list`	D	D	D	D

1 You must install `eoe.sw.dmi` on each potential CXFS metadata server. See "Using Hierarchical Storage Management (HSM) Products" on page 296.

Mount Option	IRIX Client-only Node		ProPack for Linux Client-only Node	
	IRIX MDS	ProPack MDS	IRIX MDS	ProPack MDS
`sunit`	S			
`swalloc`	S			
`swidth`	S			
`uqnoenforce`	S	S	S	S
`uquota`	S	S	S	S
`wsync`	S			

Initial Configuration Checklist

Following is a checklist of the steps you must perform when installing and configuring a CXFS system.

Caution: CXFS is a complex product. To ensure that it is installed and configured in an optimal manner, you **must** purchase initial setup services from SGI.

This checklist is not intended to be used directly by the customer, but is provided for reference. It is intended to be used only as an aid; you must be certain to read this entire manual, especially "Configuration Best Practices" on page 113 and Chapter 17, "Troubleshooting" on page 381, before attempting to complete these procedures.

- [ ] Understand the application use, storage configuration, and I/O patterns at the site. (Not all applications benefit from CXFS; see "Comparison of XFS and CXFS" on page 2.)
- [ ] Connect the SAN hardware. See the RAID documents.
- [ ] Is there a private network? This is a requirement.
- [ ] Is system reset configured for all potential metadata servers, as recommended by SGI? Is system reset or I/O fencing configured for all client-only nodes? One of these solutions is required to ensure data integrity for **all** nodes. See "Isolating Failed Nodes" on page 27, and "System Reset" on page 34.
- [ ] Are there an odd number of server-capable nodes and/or is there a client-only tiebreaker node? See "Use an Odd Number of Server-Capable Nodes" on page 116 and "Use a Client-Only Tiebreaker" on page 116.
- [ ] Read this book and any `README` files and release notes provided with the release. If you have clients running operating systems other than IRIX or SGI ProPack for Linux, also read the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.
- [ ] Verify that the network is usable. See Chapter 4, "IRIX CXFS Installation" on page 65.

[ ] Install the CXFS software. See Chapter 4, "IRIX CXFS Installation" on page 65, and Chapter 5, "Linux CXFS Installation" on page 83. If you have clients running operating systems other than IRIX or SGI ProPack for Linux, also see the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*.

[ ] Modify the configuration files and perform other tasks as needed. See Chapter 6, "Postinstallation Steps" on page 95.

[ ] Completely configure a **small** cluster (3 nodes). See Chapter 8, "Initial Configuration of the Cluster" on page 125.

[ ] Look for errors in the daemon log files in the `/var/cluster/ha/logs` directory.

[ ] If all is well, add the rest of the nodes. If there are problems, see Chapter 17, "Troubleshooting" on page 381.

[ ] Set up the filesystems. See Chapter 8, "Initial Configuration of the Cluster" on page 125.

Summary of New Features from Previous Releases

This appendix contains a summary of the new features for each version of this guide.

CXFS Version 1: Original Implementation

CXFS version 1 is the original implementation of CXFS.

IRIX 6.5.6f

Original publication (007–4016–001).

IRIX 6.5.6f

The 007–4016–002 update contains additional troubleshooting information and instructions for unmounting and remounting filesystems with the command line interface. It was reorganized to make the tasks of installation and configuration clearer.

IRIX 6.5.7f

The 007–4016–003 update contains the following:

- Metadata server recovery information
- Administrative shutdown procedures
- Additional troubleshooting information
- Instructions for unmounting and remounting filesystems with the CLI
- Reorganized installation and configuration information

IRIX 6.5.8f

The 007–4016–004 update contains the following:

- Support for hierarchical storage management (HSM) through data management application programming interface (DMAPI), also know as X/Open data storage management specification (XDSM)
- Changes to administrative shutdown, including two new `cmgr` subcommands to stop CXFS services on the local nodes: `admin cxfs_stop` and `admin cxfs_stop`
- Quorum changes without panics

IRIX 6.5.9f

The 007–4016–005 update contains the following:

- Coexecution of CXFS and IRIS FailSafe 2.1, including commands to convert nodes and clusters to apply to both utilities
- Ability to use the `cmgr` command without extra prompting (`-p`), permitting the use of scripts
- New tasks to revoke and allow membership of the local node
- Ability to specify the tie-breaker node, which is used in the process of computing node membership for the cluster when exactly half the nodes in the cluster are up and can communicate with each other
- Clarification that a single subnet should be used

IRIX 6.5.10f

The 007–4016–006 update contains the following:

- Clarifications about CXFS shutdown and database shutdown
- Additional information about CXFS daemons
- Clarifications to the comparison of XFS and CXFS

IRIX 6.5.11f

The 007–4016–007 update contains the following:

- Addition of the Origin 3000 partition ID to node configuration
- Troubleshooting information for a two-node cluster when both nodes go down
- Information about editing the `/etc/hosts` file to use an alternate interface for heartbeat and control.
- Clarification about the use of hardware reset and tie-breaker nodes
- Ability to unset the tie-breaker node
- Use of `fsr`

CXFS Version 2: MultiOS Cluster

CXFS version 2 includes client-only nodes on operating system platforms other than IRIX (*multiOS cluster*, or *heterogeneous clients*).

IRIX 6.5.12f

The 007–4016–008 update contains the following:

- A cluster of at least **three** weighted nodes is recommended for a production environment (that is, one requiring relocation and recovery of the metadata server).

  If you use a two-weighted-node cluster for production, you must do one of the following:

  – Use reset lines to avoid data corruption and ensure that only one node is running in error conditions (reset lines are recommended for all CXFS clusters and required for use with IRIS FailSafe).

  – Weight one node as 1 and the other as 0.

  – Set a tie-breaker node.

  However, there are issues with a two-weighted-node cluster.

- The new `cluster_status` command, which provides a `curses` interface to display status information gathered by the `cad` daemon (this information is also displayed by the `cxdetail` command).
- Cluster nodes can run adjacent levels of the IRIX operating system (OS); for example, 6.5.11f and 6.5.12f (this applies as of 6.5.12f).
- The ability to execute your own custom scripts around mounting operations.
- Partition ID information.
- Clarification about the following:
  - Hostname resolution rules; it is **critical** that you understand these rules and have files configured poperly before attempting to configure a cluster.
  - The difference between *CXFS membership* and `fs2d` *membership*.
  - Configuration of nodes supported in an IRIS FailSafe and CXFS coexecution environment.
  - Unwritten extent tracking (`unwritten=1|0`) with CXFS.
  - Including the CXFS mount point in the `/etc/exports` file.
  - Number of nodes supported: 16 CXFS nodes, and up to 8 IRIS FailSafe nodes with coexecution.
  - The flow of information in a coexecution cluster.

IRIX 6.5.13f

The 007–4016–009 update contains the following:

- The structure of the CXFS filesystem configuration has changed. CXFS filesystems can now be defined, modified, managed and deleted independently of each other and of the cluster definition. (Previously, the CXFS filesystems were defined as attributes to the cluster definition.)

  The new design improves the performance, flexibility and reliability of filesystem configuration. To accommodate clusters mixing nodes running 6.5.12 and 6.5.13, backwards compatibility is enforced by default in 6.5.13. The result is that the performance achievements are not visible; however, if you are 6.5.13 on all nodes in the cluster, you may wish to turn off backwards compatibility. Backwards compatibility will be turned off in the 6.5.14 release.

- Information about locating the `xfsdump` inventory in a shared directory.
- Information about the IRIS FailSafe `CXFS` resource type that can be used to failover applications that use CXFS filesystems.
- The `-p` option is no longer required when defining filesystems with the `cmgr` command; the scripting capability is therefore provided.
- For certain GUI tasks, the ability to select all nodes at once in addition to specifying nodes individually.
- New `/var/cluster/cmgr-scripts/rotatelogs` script to save log files with day and month name as suffixes.
- The setting to force an unmount of a filesystem using the `umount -k` option is turned off by default in the GUI. There is no default when using the `cmgr` command.
- Clarification of the term *CLI* to mean the underlying set of commands that are used by the `cmgr` cluster manager tool and by the GUI.
- Use of `sgi_apache.sw.server`.
- Correction: real-time filesystems are not currently supported. Changes to reflect this have been made in text.
- New and revised figures.

IRIX 6.5.14f

The 007–4016–011 update contains the following:

- The graphical user interface (GUI) has been improved. The separate cluster view (the `cxdetail` command) and task manager (the `cxtask` command) have been streamlined into one window, the **CXFS Manager**. Both the `cxtask` and `cxdetail` commands are kept for historical purposes; this document refers to just `cxtask` for simplicity.

  The new GUI provides the following features:

  – Access to tasks through the menu bar or by clicking the right mouse button within the tree view

  – Faster filesystem status and cluster status updates

  - Access to the `salog`(4) file, which shows every command run from the GUI
  - A **Find** textfield helps you find components within the displayed tree-view
- Information about the use of `xfs_repair` and CXFS filesystems.

Caution: Do not use `xfs_repair` on a CXFS filesystem unless you are certain there is a problem.

- Information about using `cmgr`(1M):
  - Invoking subcommands directly on the command line with the `-c` option
  - Using template scripts provided in the `/var/cluster/cmgr-templates` directory
- Information about MAC labels in a mixed Trusted IRIX and IRIX cluster.
- The structure of the CXFS filesystem configuration was changed with the release of IRIX 6.5.13. Backward compatibility with earlier versions is no longer maintained as of IRIX 6.5.14, because all nodes in the cluster must be running the same or adjacent releases.

  If you are upgrading from 6.5.13f entirely to 6.5.14f, there is no further impact.

  If you intend to run a mixture of 6.5.13f and 6.5.14f nodes, you must turn off backward compatibility.

  If you are upgrading from 6.5.12f or earlier without first installing and running 6.5.13f, then you must perform a one-time manual conversion of your CXFS filesystem definitions.

IRIX 6.5.15f

The 007–4016–012 update contains the following:

Note: Relocation and recovery are deferred in this release.

- Support for clients of other operating systems such as Solaris and Windows NT as defined in the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage*. These clients will be released asynchronously from the IRIX release. This support will require a minimum of IRIX 6.5.15f plus appropriate patches. For more information, see your SGI support contact.

- Default scripts are now provided in the `/var/cluster/clconfd-scripts` directory to permit NFS-exporting of CXFS filesystems listed in `/etc/exports`.
- Reset lines are mandatory for two-node and two-weighted node clusters. Larger clusters should have an odd number of weighted nodes, or must have serial reset lines if only two of the nodes are weighted.
- Simplification of Chapter 1. General information about the CXFS Manager GUI and `cmgr` have been moved to their respective reference chapters, coexecution details have been moved into a separate chapter, and the communication flow diagrams and daemon information have been moved into an appendix.
- Information about the error messages that may cause administrators to use `xfs_repair` inappropriately.
- Changes to the `rotatelogs` script syntax. The `root crontab` file now has an entry to run the `rotatelogs` script weekly. If you run the script twice in one day, it will append the current log file to the previous saved copy, rather than overwriting it.
- A new figure describing some of the various combinations of node and cluster types in a coexecution cluster.

IRIX 6.5.16f

The 007–4016–013 update contains the following:

Note: Relocation and recovery are fully implemented, but the number of associated problems prevents support of these features in CXFS. While data integrity is not compromised, cluster node panics or hangs are likely to occur. These features will be fully supported when these issues are resolved.

- Support for Solaris and Windows NT systems in a multiple operating system (multiOS) cluster, including the following:
  - Information about defining the operating system for a node. For existing clusters that are upgraded to IRIX 6.5.16f, existing nodes will be assigned an operating system type of `IRIX`.

  - Information about I/O fencing, which allows a problem node to be isolated from the storage area network (SAN) so that it cannot corrupt data in the shared CXFS filesystem. Solaris and Windows NT nodes require a Brocade switch in order to support I/O fencing for data integrity protection; therefore, the Brocade switch is a required piece of hardware in a cluster running multiple operating systems.
  - The new terms *multiOS* and *CXFS client-only node*.
- Support for the L1 controller on SGI Origin 300, SGI Origin 3200C, SGI Onyx 300, and SGI Onyx 3200C systems.
- Information about the CXFS GUI tasks to define and modify a filesystem, which have been split into two pages for ease of use.
- New GUI icons.

IRIX 6.5.17f

The 007–4016–014 update contains the following:

- A new appendix contains an example `/etc/ipfilterd.conf` file that can be used to provide IP filtering for the CXFS private network.
- The `build_cmgr_script` command, which generates a `cmgr` script from the cluster database. The script can be used later to recreate the cluster database after performing a `cdbreinit` command.
- A sample script to unexport and locally unmount an `lofs` filesystem.
- Use of the new command name `cxfsmgr`. The `cxfsmgr` command has the same function as the `cxtask` and `cxdetail` commands, which are kept for historical purposes.
- Clarifications to the following:
  - Starting the CXFS Manager graphical user interface
  - Masking and I/O fencing
  - Terminology such as *cluster*, *node*, and *pool*
  - Terminology used to describe the GUI

IRIX 6.5.18f

The 007–4016–015 update contains the following:

Note: In this release, relocation is disabled by default and recovery is supported only when using standby nodes.

A *standby node* is a metadata server-capable administration node that is configured as a potential metadata server for a given filesystem, but does not currently run any applications that will use that filesystem. To use recovery, you must not run any applications on any of the potential metadata servers for a given filesystem; after the active metadata server has been chosen by the system, you can then run applications that use the filesystem on the active metadata server and client-only nodes.

Relocation and recovery are fully implemented, but the number of associated problems prevents full support of these features in the current release. Although data integrity is not compromised, cluster node panics or hangs are likely to occur. Relocation and recovery will be fully supported in a future release when these issues are resolved.

- IRIX nodes may now be CXFS client-only nodes, meaning that they run a minimal implementation of the CXFS and cluster services, and do not contain a copy of the CXFS cluster database. Client-only nodes are installed with the `cxfs_client` software product.

  This change also introduces the term *CXFS administration node*, which is a node that is installed with the `cluster_admin` software product, allowing the node to perform cluster administration tasks and contain a copy of the cluster database. Nodes that you want to run as metadata servers must be installed as CXFS server-capable administration nodes; SGI recommends that all other nodes be installed as client-only nodes.

  When you define a node, you no longer need to specify the node weight. This has been replaced by the **Node Function** field, allowing you to choose **Server-capable Admin**, **Client Admin**, or **Client-Only**. (For Solaris and Windows nodes, **Client-Only** is automatically selected for you.) Similar fields are provided for the `cmgr` command.

  When upgrading to 6.5.18f, already existing IRIX nodes will by default be assigned as **Server-capable Admin** if they had a weight of 1.

  This version also clarifies the terms used for membership: *CXFS kernel membership* and *cluster database membership*.

- New system-tunable parameters:
  - `cxfs_relocation_ok` lets you enable or disable the relocation feature; relocation is disabled by default in this release, and SGI recommends that you **do not** enable it.
  - `cxfsd_min` and `cxfsd_max` let you specify the minimum and maximum number of `cxfsd` threads to run per CXFS filesystem.
- New commands:
  - `cxfs_info` provides status information about the cluster, nodes, and filesystems and is run from a client-only node.
  - `cxfsdump` gathers CXFS configuration information.
- A CXFS cluster is supported with as many as 32 nodes. As many as 16 of those nodes can be CXFS administration nodes and all other nodes can be client-only nodes. You can choose to define a node as a CXFS client administration node, however, SGI strongly recommends that only potential metadata servers be configured as CXFS server-capable administration nodes and that there be an odd number of server-capable nodes for quorum calculation purposes.
- The graphical user interfaces for XVM and CXFS have been combined into one. This guide provides an overview of the XVM-specific tasks provided by the GUI; for details about these tasks, see the *XVM Volume Manager Administrator's Guide*.

  The tasks to make, grow, mount/unmount a filesystem are now provided in the GUI.
- Tips about using CXFS and Trusted IRIX.
- Support for Microsoft Windows 2000 systems as client-only nodes. (This guide uses *Windows* to refer to both Microsoft Windows NT and Microsoft Windows 2000 nodes when the information applies equally to both. Information that applies to only one of these types of nodes is identified.)

IRIX 6.5.19f

The 007–4016–016 update contains the following:

- The new rolling annual upgrade policy that permits you to upgrade from 6.5.*n* to the *n*+1 or *n*+4 release, as of 6.5.18f.

- The time required to update and propagate the database across nodes in the cluster has been significantly decreased.
- If you use I/O fencing and `ipfilterd` on a node, the `ipfilterd` configuration must allow communication between the node and the `telnet`(1) port on the switch.
- The following nodes do not contain system controllers and therefore require I/O fencing for data integrity protection:
  - Silicon Graphics Fuel visual workstation
  - Silicon Graphics Octane system
  - Silicon Graphics Octane2 system
- The CXFS Manager graphical user interface (GUI) has added a new icon to represent client-only nodes.
- In preparation for future CXFS MultiOS client releases, the CXFS software now also allows you to specify the Linux, IBM AIX, and Hewlett-Packard HP-UX operating systems when defining a node. For support details, see the *CXFS MultiOS Client-Only Guide for SGI InfiniteStorage* and release notes.
- This version clarifies the various methods to perform cluster database backups and restorations.
- Application programmers should be aware that XFS recently relaxed the requirement that direct I/O be aligned along filesystem block boundaries. As of IRIX 6.5.19f, direct I/O will also be accepted using 512-byte alignment.

  This change makes the use of direct I/O on a CXFS partition more consistent with that of other vendor's requirements and thus makes the use of CXFS more transparent. See the description of direct I/O requirements in the `fcntl` man page.
- This version lists the system tunable parameters found in the `/var/sysgen/mtune/cell` file, some of which should not be modified.

IRIX 6.5.20f

The 007–4016–017 update contains the following:

- Changes to the CXFS graphical user interface (GUI):
  - New login connection choices, including support for a remote shell connection, which connects to the server via a user-specified command shell, such as `rsh` or `ssh`.
  - The ability for the `root` user to grant other users permission to execute specific GUI tasks.
  - Use of Java2 for the CXFS GUI, which simplifies the Java installation and co-operation with third-party GUIs. This also enhances the ability to run the GUI through a web browser (via http://*server*/CXFSManager/).
  - Information about using the right mouse button to access tasks appropriate to the selected GUI item.
- Changes to the `cxfsd_min` and `cxfsd_max` defaults, and the `cxfsd_max` legal values.
- More information about memberships, quorums, and tiebreakers.
- A new figure describing standby mode.
- More information about IRIX client-only issues:
  - Client-only node system files
  - Status in log files
  - `cxfs_client` error messages

CXFS Version 3: IRIX or Linux Servers

CXFS version 3 adds support for CXFS metadata servers on SGI Altix systems running SGI ProPack for Linux.

CXFS 3.0 for IRIX 6.5.22 and SGI ProPack 2.3 for Linux

The 007–4016–018 update contains the following:

- Support for SGI ProPack for Linux metadata servers on SGI Altix 3000 family of servers and superclusters. A CXFS cluster can contain either SGI ProPack 2.3 for Linux server-capable nodes on Altix systems or IRIX server-capable nodes; you cannot mix IRIX and Linux server-capable nodes within one cluster.

  CXFS does not support the relocation or recovery of DMAPI filesystems that are being served by Linux metadata servers.

  Coexecution with FailSafe is not supported on Linux nodes.

- Due to packaging enhancements, CXFS may now be installed on the M stream or the F stream.

  The IRIX CXFS software will no longer be bundled in the IRIX overlay CDs but instead is on a separate *CXFS IRIX Server and Client 3.0 for IRIX 6.5.22* CD. This changes the installation procedure.

  Note: If you are upgrading from a previous IRIX release and have CXFS installed, you must upgrade both IRIX and CXFS. If you try to upgrade one without the other, conflicts will occur.

- Information about defining networks for CXFS kernel messaging (in addition to the network used for heartbeat/control). However, use of these networks is **deferred**.
- Support for IRIX real-time filesystems.
- Suggestions for configuring large clusters.
- Information about using `ping` to verify general connectivity and CXFS heartbeat in a multicast environment.
- The GUI has been changed to show a single display for the nodes in the cluster and nodes that are in the pool but not in the cluster. This new selection is **View: Nodes and Cluster**.
- Information about information retaining system core files and the output from the `cxfsdump` utility when reporting problems.
- Information about monitoring heartbeat timeouts for IRIX using Performance Co-Pilot or the `icrash` command.

- The ability to define multiple CXFS filesystems at one time with the GUI.

CXFS 3.1 for IRIX 6.5.23 and SGI ProPack 2.4 for Linux

The 007–4016–019 update contains the following:

- Information about migrating from an IRIX cluster to a Linux cluster
- Support for a cluster of up to 64 nodes.
- Information about the TP9300 RAID.
- Information about the `cxfs-config` command.
- Clarification that serial hardware reset lines or I/O fencing is **required for all nodes** in order to protect data integrity.
- The ability to define a reset method for a given node to one of the following:
  - `powerCycle` to turn power off and on
  - `reset` to perform a serial reset
  - `nmi` to perform a nonmaskable interrupt

  You can define this method using either the `cmgr` command or the GUI. You can manually perform a powercycle or an NMI with the `cmgr` command.
- New appendixes summarizing operating system path differences and new features from previous releases

CXFS 3.2 for IRIX 6.5.24 and SGI ProPack 3.0 for Linux

The 007–4016–021 update contains the following:

- Information about private network failover as defined with the `cmgr` command. (Although the primary network must be private, the backup network may be public.)
- Support for Guaranteed-rate I/O version 2 (GRIOv2) in the IRIX installation procedure.
- Corrections to CXFS and cluster administration path differences between IRIX and Linux 64-bit on SGI Altix systems.

- Updated the example for `clconf_info` command. The `clconf_info` command now reports a node as `inactive` rather than `DOWN*` and the unused incarnation number has been removed.
- Support for token obtain optimization. To disable, use the `cxfs_prefetch` system tunable parameter.
- If you have a cluster with an even number of server-capable nodes and no tiebreaker: to avoid a split-brain scenario, you should not use the **Shutdown** setting on any server-capable node.
- Information about multiple Ethernet interfaces on SGI Altix systems and providing persistent device naming.
- Clarification about the `chkconfig` arguments used for IRIX administration nodes, Linux 64-bit administration nodes, and client-only nodes.
- Information about the correct options to use for quotas on Linux clusters (`uquota` and `gquota`).
- Information about serial reset configurations.
- Information about using the `hafence`(1M) command to define a QLogic switch. (You cannot use the GUI or the `cmgr` command to define or modify a switch other than a Brocade switch.)
- If you want to use quotas on a CXFS filesystem, you must install the quota package.
- Information about removing a metadata server from the cluster for maintenance.
- Mount options information.
- Addition of the XVM graphical user interface (GUI) to the CXFS Linux package.

CXFS 3.3

The 007–4016–022 update contains the following:

- Procedures to remove a single client from the cluster and restore it, and shut down the entire cluster.
- A new chapter about best practices.
- Information about XVM failover.

- Updates to the rolling upgrade policy.
- Information about performing a miniroot install.
- Information about installing the latest Java2 software for use with the CXFS GUI and SGI ProPack for Linux.
- Information about modifying the `httpd.conf` file in order to use the CXFS GUI on Linux.
- Clarifications to terminology.

Glossary

active metadata server

A server-capable administration node chosen from the list of potential metadata servers. There can be only one active metadata server for any one filesystem.

administration node

A node in the pool that is installed with the `cluster_admin.sw.base` software product, allowing the node to perform cluster administration tasks and contain a copy of the cluster database. There are two types of administration nodes: server-capable administration nodes and client administration nodes.

administrative stop

See *forced CXFS shutdown*

cell ID

A number associated with a node that is used by the CXFS software and appears in messages.

CLI

Underlying command line interface commands used by the CXFS Manager graphical user interface (GUI) and the `cmgr` command.

client

See *CXFS client node*, *CXFS client-only node* and *administration node*.

client administration node

A node that is installed with the `cluster_admin` software product, allowing the node to perform cluster administration tasks and contain a copy of the cluster database, but is not capable of coordinating CXFS metadata. Only to be used with FailSafe.

client-only node

A node that is installed with the `cxfs_client.sw.base` software product; it does not run cluster administration daemons and is not capable of coordinating CXFS metadata. Any node can be client-only node. See also *CXFS administration node* and *client administration node*.

cluster

A *cluster* is the set of systems (nodes) configured to work together as a single computing resource. A cluster is identified by a simple name and a cluster ID. A cluster running multiple operating systems is known as a *multiOS cluster*.

There is only one cluster that may be formed from a given pool of nodes.

Disks or logical units (LUNs) are assigned to clusters by recording the name of the cluster on the disk (or LUN). Thus, if any disk is accessible (via a Fibre Channel connection) from machines in multiple clusters, then those clusters must have unique names. When members of a cluster send messages to each other, they identify their cluster via the cluster ID. Cluster names must be unique.

Because of the above restrictions on cluster names and cluster IDs, and because cluster names and cluster IDs cannot be changed once the cluster is created (without deleting the cluster and recreating it), SGI advises that you choose unique names and cluster IDs for each of the clusters within your organization.

cluster administration daemons

The set of daemons on a CXFS administration node that provide the cluster infrastructure: `fs2d`, `cad`, `cmond`, `crsd`. See "CXFS Control Daemon" on page 26.

cluster administrator

The person responsible for managing and maintaining a cluster.

cluster database

Contains configuration information about all nodes and the cluster. The database is managed by the cluster administration daemons.

cluster domain

XVM concept in which a filesystem applies to the entire cluster, not just to the local node. See also *local domain*.

cluster database membership

The group of administration nodes in the **pool** that are accessible to cluster administration daemons and therefore are able to receive cluster database updates; this may be a subset of the nodes defined in the pool. The cluster administration daemons manage the distribution of the cluster database (CDB) across the administration nodes in the pool. (Also known as *user-space membership* and *fs2d database membership*.)

cluster ID

A unique number within your network in the range 1 through 128. The cluster ID is used by the operating system kernel to make sure that it does not accept cluster information from any other cluster that may be on the network. The kernel does not use the database for communication, so it requires the cluster ID in order to verify cluster communications. This information in the kernel cannot be changed after it has been initialized; therefore, you must not change a cluster ID after the cluster has been defined. Clusters IDs must be unique.

cluster mode

One of two methods of CXFS cluster operation, `Normal` or `Experimental`. In `Normal` mode, CXFS resets any node for which it detects heartbeat failure; in `Experimental` mode, CXFS ignores heartbeat failure. `Experimental` mode allows you to use the kernel debugger (which stops heartbeat) without causing node failures. You should only use `Experimental` mode during debugging.

control messages

Messages that cluster software sends between the cluster nodes to request operations on or distribute information about cluster nodes. Control messages and heartbeat messages are sent through a node's network interfaces that have been attached to a control network.

cluster node

A node that is defined as part of the cluster. See also *node*.

coexecution

The ability to run CXFS and IRIS FailSafe together. For more information, see "Overview of FailSafe Coexecution" on page 39.

control network

The network that connects nodes through their network interfaces (typically Ethernet) such that CXFS can send heartbeat messages and control messages through the network to the attached nodes. CXFS uses the highest priority network interface on the control network; it uses a network interface with lower priority when all higher-priority network interfaces on the control network fail.

CXFS client daemon

The daemon (`cxfs_client`) that controls CXFS services on a client-only node. See "CXFS Client Daemon" on page 26.

CXFS control daemon

The daemon (`clconfd`) that controls CXFS services on an administration node. See "Cluster Administration Daemons" on page 24.

CXFS database

See *cluster database*.

CXFS kernel membership

The group of CXFS nodes that can share filesystems in the cluster, which may be a subset of the nodes defined in a cluster. During the boot process, a node applies for CXFS kernel membership. Once accepted, the node can share the filesystems of the cluster. (Also known as *kernel-space membership*.) CXFS kernel membership differs from *cluster database membership* and FailSafe membership. For more information about FailSafe, see *FailSafe Administrator's Guide for SGI InfiniteStorage*.

CXFS services

The enabling/disabling of a node, which changes a flag in the cluster database. This disabling/enabling does not affect the daemons involved. The daemons that control CXFS services are `clconfd` on an administration node and `cxfs_client` on a client-only node. See "CXFS Services" on page 25.

CXFS services start

To enable a node, which changes a flag in the cluster database, by using an administrative task in the CXFS GUI or the `cmgr` command.

CXFS services stop

To disable a node, which changes a flag in the cluster database, by using a CXFS graphical user interface (GUI) or the `cmgr` command. See also *forced CXFS shutdown*.

CXFS shutdown

See *forced CXFS shutdown* and *shutdown*

CXFS tiebreaker node

A node identified as a tiebreaker for CXFS to use in the process of computing CXFS kernel membership for the cluster, when exactly half the nodes in the cluster are up and can communicate with each other. There is no default CXFS tiebreaker. SGI recommends that the tiebreaker node be a client-only node. The CXFS tiebreaker differs from the FailSafe tiebreaker; see *FailSafe Administrator's Guide for SGI InfiniteStorage*.

database

See *cluster database*.

database membership

See *cluster database membership*.

details area

The portion of the GUI window that displays details about a selected component in the view area. See also *view area*.

domain

See *cluster domain* and *local domain*.

FailSafe Membership

The group of nodes that are actively sharing resources in the cluster, which may be a subset of the nodes defined in a cluster. FailSafe membership differs from *CXFS kernel membership* and *cluster database membership*. For more information about FailSafe, see *FailSafe Administrator's Guide for SGI InfiniteStorage*.

failure action hierarchy

The set of instructions that determine what happens to a failed node; the second instruction will be followed only if the first instruction fails; the third instruction will be followed only if the first and second fail. The available actions are: *I/O fencing*, *reset*, and *shutdown*.

fencing

See *I/O fencing*.

fencing recovery

The process of recovery from fencing, in which the affected node automatically withdraws from the CXFS kernel membership, unmounts all file systems that are using an I/O path via fenced HBA(s), and then rejoins the cluster.

forced CXFS shutdown

The withdrawl of a node from the CXFS kernel membership, either due to the fact that the node has failed somehow or by issuing an `admin cxfs_stop` command. This disables filesystem and cluster volume access for the node. The node remains enabled in the cluster database. See also *CXFS services stop* and *shutdown*.

fs2d database membership

See *cluster database membership*.

heartbeat messages

Messages that cluster software sends between the nodes that indicate a node is up and running. Heartbeat messages and *control messages* are sent through the node's network interfaces that have been attached to a control network.

heartbeat interval

The time between heartbeat messages. The node timeout value must be at least 10 times the heartbeat interval for proper CXFS operation. The higher the number of heartbeats (smaller heartbeat interval), the greater the potential for slowing down the network.

I/O fencing

The failure action that isolates a problem node so that it cannot access I/O devices, and therefore cannot corrupt data in the shared CXFS filesystem. I/O fencing can be applied to any node in the cluster (CXFS clients and metadata servers). The rest of the cluster can begin immediate recovery.

kernel-space membership

See *CXFS kernel membership*.

local domain

XVM concept in which a filesystem applies only to the local node, not to the cluster. See also *cluster domain*.

log configuration

A log configuration has two parts: a *log level* and a *log file*, both associated with a *log group*. The cluster administrator can customize the location and amount of log output, and can specify a log configuration for all nodes or for only one node. For example, the `crsd` log group can be configured to log detailed level-10 messages to the `crsd-foo` log only on the node `foo` and to write only minimal level-1 messages to the `crsd` log on all other nodes.

log file

A file containing notifications for a particular *log group*. A log file is part of the *log configuration* for a log group.

log group

A set of one or more CXFS processes that use the same log configuration. A log group usually corresponds to one daemon, such as `gcd`.

log level

A number controlling the number of log messages that CXFS will write into an associated log group's log file. A log level is part of the log configuration for a log group.

membership

See *cluster database membership* and *CXFS kernel membership*.

membership version

A number associated with a node's cell ID that indicates the number of times the CXFS kernel membership has changed since a node joined the membership.

metadata

Information that describes a file, such as the file's name, size, location, and permissions.

metadata server

The administration node that coordinates updating of meta data on behalf of all nodes in a cluster. There can be multiple potential metadata servers, but only one is chosen to be the active metadata server for any one filesystem.

metadata server recovery

The process by which the metadata server moves from one node to another due to an interruption in CXFS services on the first node. See also *recovery*

multiOS

A cluster that is running multiple operating systems, such as IRIX and Solaris.

multiport serial adapter cable

A device that provides four DB9 serial ports from a 36-pin connector.

node

A *node* is an operating system (OS) image, usually an individual computer. (This use of the term *node* does not have the same meaning as a node in an SGI Origin 3000 or SGI 2000 system.)

A given node can be a member of only one pool (and therefore) only one cluster.

See also *CXFS administration node*, *client administration node*, *client-only node*, *server-capable administration node*, and *standby node*,

node ID

An integer in the range 1 through 32767 that is unique among the nodes in the pool. If you do not specify a number, CXFS will calculate an ID for you. You must not change the node ID number after the node has been defined.

node membership

The list of nodes that are active (have CXFS kernel membership) in a cluster.

node timeout

If no heartbeat is received from a node in this period of time, the node is considered to be dead. The node timeout value must be at least 10 times the heartbeat interval for proper CXFS operation.

notification command

The command used to notify the cluster administrator of changes or failures in the cluster and nodes. The command must exist on every node in the cluster.

owner host

A system that can control a node remotely, such as power-cycling the node. At run time, the owner host must be defined as a node in the pool.

owner TTY name

The device file name of the terminal port (TTY) on the *owner host* to which the system controller is connected. The other end of the cable connects to the node with the system controller port, so the node can be controlled remotely by the owner host.

pool

The *pool* is the set of nodes from which a particular cluster may be formed. Only one cluster may be configured from a given pool, and it need not contain all of the available nodes. (Other pools may exist, but each is disjoint from the other. They share no node or cluster definitions.)

A pool is formed when you connect to a given node and define that node in the cluster database using the CXFS GUI or `cmgr` command. You can then add other nodes to the pool by defining them while still connected to the first node, or to any other node that is already in the pool. (If you were to connect to another node and then define it, you would be creating a second pool).

port password

The password for the system controller port, usually set once in firmware or by setting jumper wires. (This is not the same as the node's root password.)

potential metadata server

A server-capable administration node that is listed in the metadata server list when defining a filesystem; only one node in the list will be chosen as the active metadata server.

quorum

The number of nodes required to form a cluster, which differs according to membership:

- For CXFS kernel membership:
  - A majority (>**50%**) of the server-capable nodes in the cluster are required to **form** an initial membership
  - Half (**50%**) of the server-capable nodes in the cluster are required to **maintain** an existing membership
- For cluster database membership, **50%** of the **nodes in the pool** are required to form and maintain a cluster.

recovery

The process by which a node is removed from the CXFS kernel membership due to an interruption in CXFS services. It is during this process that the remaining nodes in the CXFS kernel membership resolve their state for cluster resources owned or shared with the removed node. See also *metadata server recovery*

relocation

The process by which the metadata server moves from one node to another due to an administrative action; other services on the first node are not interrupted.

reset

The failure action that performs a system reset via a serial line connected to the system controller. The reset may be a powercycle, serial reset, or NMI (nonmaskable interrupt). This failure action hierarchy choice is recommended for all potential metadata servers; see "Requirements" on page 37.

server-capable administration node

A node that is installed with the `cluster_admin` product and is also capable of coordinating CXFS metadata.

shutdown

The fail action hierarchy selection that tells the other nodes in the cluster to wait before reforming the CXFS kernel membership. The surviving cluster delays the beginning of recovery to allow the node time to complete the shutdown. See also *forced CXFS shutdown*.

snooping

A security breach involving illicit viewing.

split-brain syndrome

A situation in which multiple clusters are formed due to a network partition and the lack of reset and/or CXFS tiebreaker capability.

spoofing

A security breach in which one machine on the network masquerades as another.

standby node

A server-capable administration node that is configured as a potential metadata server for a given filesystem, but does not currently run any applications that will use that filesystem.

storage area network (SAN)

A dedicated, high-speed, scalable network of servers and storage devices designed to enhance the storage, retrieval, and management of data

system controller port

A port sitting on a node that provides a way to power-cycle the node remotely. Enabling or disabling a system controller port in the cluster database tells CXFS whether it can perform operations on the system controller port.

system log file

Log files in which system messages are stored

tiebreaker node

See *CXFS tiebreaker node*.

user-space membership

See *cluster database membership*.

view area

The portion of the GUI window that displays components graphically. See also *details area*.

Index