SGI® InfiniteStorage CXFS™ Administration Guide

007–4016–018

CONTRIBUTORS

Written by Lori Johnson

Illustrated by Chrystie Danzer

Engineering contributions to the book by Rich Altmaier, Neil Bannister, François Barbou des Places, Ken Beck, Felix Blyakher, Laurie Costello, Mark Cruciani, Dave Ellis, Brian Gaffey, Philippe Gregoire, Dean Jansa, Erik Jacobson, 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, Wesley Smith, Kerm Steffenhagen, Paddy Sreenivasan, Andy Tran, Rebecca Underwood, Connie Waring, Geoffrey Wehrman

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

New Features in This Guide

Note: Relocation is not supported in this release.

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 relocation or 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.

This update contains the following:

- Support for SGI ProPack for Linux 64-bit metadata servers on SGI Altix 3000 family of servers and superclusters. A CXFS cluster can contain either Linux 64-bit for SGI ProPack 2.3 server-capable nodes on Altix systems or IRIX server-capable nodes; you cannot mix IRIX and Linux 64-bit server-capable nodes within one cluster.

  CXFS does not support the relocation or recovery of DMAPI filesystems that are being served by Linux 64-bit metadata servers.

  Coexecution with FailSafe is not supported on Linux 64-bit 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; see Chapter 6, "IRIX CXFS Installation", page 61.

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). If you supply multiple interfaces, the network will fail over from a higher-priority network to a lower-priority network. However, use of these networks is **deferred** in this release and may be available with a patch. See:
  - "Define a Node with the GUI", page 152
  - "Modify a Node Definition with the GUI", page 163
  - "Define a Node with `cmgr`", page 202
  - "Modify a Node with `cmgr`", page 212
- Support for IRIX real-time filesystems. See "Configuring Real-Time Filesystems For IRIX Nodes", page 262.
- Suggestions for configuring large clusters. See "Configuring a Large Cluster", page 125.
- Information about using `ping` to verify general connectivity and CXFS heartbeat in a multicast environment; see "Verifying Connectivity in a Multicast Environment", page 378.
- 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. See "Reporting Problems to SGI", page 379.
- Information about monitoring heartbeat timeouts for IRIX using Performance Co-Pilot or the `icrash` command. See "Heartbeat Timeout Status", page 320.
- The ability to define multiple CXFS filesystems at one time with the GUI; see "Define CXFS Filesystems with the GUI", page 183.

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.

Contents

Figures

Tables

About This Guide

This publication documents CXFS 3.0 running on a storage area network (SAN). It supports CXFS 3.0 for IRIX 6.5.22 and for SGI Altix 3000 systems running SGI ProPack 2.3 for Linux. 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 16, "Troubleshooting", page 323, before attempting to install and configure a CXFS cluster.

Related Publications

The following documents contain additional information:

- *SGI InfiniteStorage CXFS MultiOS Client-Only Guide*
- *SGI InfiniteStorage FailSafe 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 TP9400 and SGI TP9500 RAID Owner's Guide*
  - *SGI TP9400 and SGI TP9500 RAID Administration Guide*
  - *SGI Total Performance 9100 Storage System Owner's Guide*
- IRIX documentation:
  - *IRISconsole Administrator's Guide* (Note, CXFS does **not** support the Silicon Graphics O2 workstation as a CXFS node, therefore it cannot be a CXFS reset server.)
  - *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 64-bit and SGI Altix documentation:
  - *NIS Administrator's Guide*
  - *Personal System Administration Guide*
  - *SGI ProPack for Linux Start Here*
  - *SGI Altix 3000 User's Guide*
  - *Performance Co-Pilot for IA-64 Linux User's and Administrator's Guide*

The following man pages are provided with CXFS:

- `build_cmgr_script`
- `cbeutil`
- `cdbBackup`
- `cdbRestore`
- `cdbutil`
- `cluster_status`
- `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 Linux 64-bit systems, see `linux-64/README_CXFS_LINUX64_3.0.0.txt` on the *CXFS 3.0 Altix Server/Client and XVM Plexing for SGI ProPack 2.3* CD
- You can view man pages by typing `man` *title* at a command line.

Conventions

Note: 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.

Linux 64-bit refers to the SGI ProPack for Linux operating system running on the SGI Altix 3000 system.

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 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.

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
  1600 Amphitheatre Parkway, M/S 535
  Mountain View, California 94043–1351

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

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 16, "Troubleshooting", page 323, before attempting to install and configure a CXFS cluster.

This chapter discusses the following:

- "What is CXFS?"
- "Comparison of XFS and CXFS", page 2
- "Comparison of Network and CXFS Filesystems", page 6
- "Cluster Environment", page 8
- "Hardware and Software Support", page 32
- "Overview of FailSafe Coexecution", page 38
- "Cluster Manager Tools Overview", page 39

Note: In this book, *Linux 64-bit* refers to the SGI ProPack for Linux operating system running on the SGI Altix 3000 system.

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.

CXFS and IRIS FailSafe share the same infrastructure.

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 from all hosts (nodes) in the cluster. CXFS filesystems are mounted across the cluster by CXFS management software. All files in the filesystem are visible to all hosts that 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 ans 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:

- An icon indicating that the filesystem is mounted (the static and dynamic states match)
- An icon indicating that the filesystem is ready 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)
- 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` shows the static cluster state. This command is available on nodes used for cluster administration.
- `clconf_info` and `cluster_status` show both the static and dynamic cluster states. These commands are available on nodes used for cluster administration.
- `cxfs_info` command 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.

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 a CXFS metadata server than when it is performed from a CXFS client.
- 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.

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*.

  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. See "Initial Configuration Requirements and Recommendations", page 103.

The following XFS features are not supported in CXFS:

- Guaranteed-rate I/O.
- Swap to a file.

Cluster Environment

This section discusses the following:

- "Terminology"
- "Isolating Failed Nodes", page 22
- "The Cluster Database and CXFS Clients", page 30
- "Metadata Server Functions", page 31
- "System View", page 32

For details about CXFS daemons, communication paths, and the flow of metadata, see Appendix A, "CXFS Software Architecture", page 383.

Terminology

This section defines the terminology necessary to understand CXFS. Also see the Glossary, page 423.

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 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 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. Thus, if two clusters will be sharing the same network for communications, then they must have unique cluster IDs. In the case of multiOS clusters, both the names and IDs must be unique if the clusters share a network.

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.

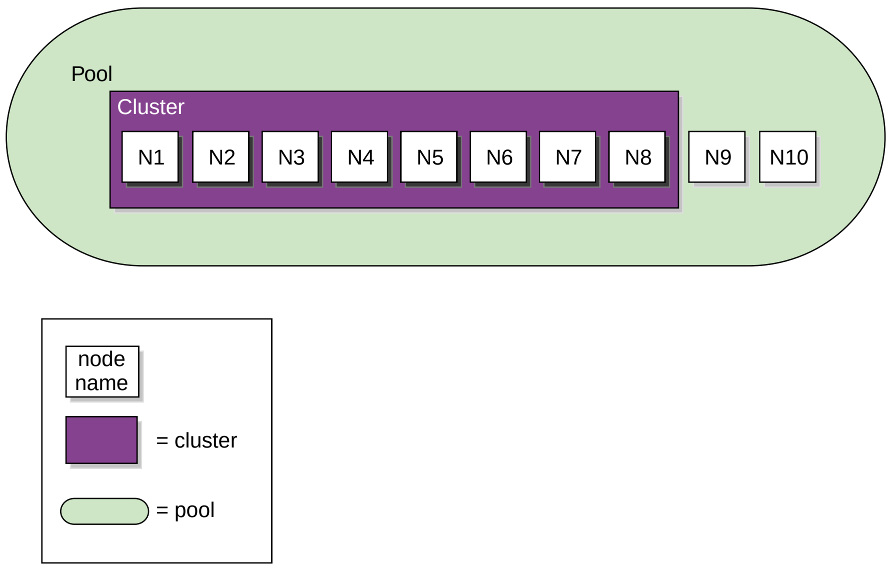


Figure 1-1 Pool and Cluster Concepts

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.

The database consists of a collection of files; you can view and modify the contents of the database by using the CXFS Manager GUI and the `cmgr`, `cluster_status`, `clconf_info` and `cxfs_info` commands. You must connect the GUI to a CXFS administration node, and the `cmgr`, `cluster_status`, and `clconf_info` commands must run on a CXFS administration node. You can use the `cxfs_info` command on client-only nodes.

Node Functions

A node can have one of the following functions:

- *CXFS metadata server-capable administration node* (IRIX or Linux 64-bit).

  This node is installed with the `cluster_admin` software product, which contains the full set of CXFS cluster administration daemons (`fs2d`, `clconfd`, `crsd`, `cad`, and `cmond`; for more details about daemons, see Appendix A, "CXFS Software Architecture", page 383.)

  This node type is capable of coordinating cluster activity and meta data. *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 64-bit. 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 10, "Reference to GUI Tasks for CXFS", page 129

  – Chapter 11, "Reference to `cmgr` Tasks for CXFS", page 195

  There should be an odd number of server-capable administration nodes for quorum calculation purposes.

- *CXFS client administration node* (IRIX or Linux 64-bit).

  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 runs a minimal implementation of the cluster services. 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 64-bit 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 64-bit are always configured as CXFS client-only nodes.

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

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.

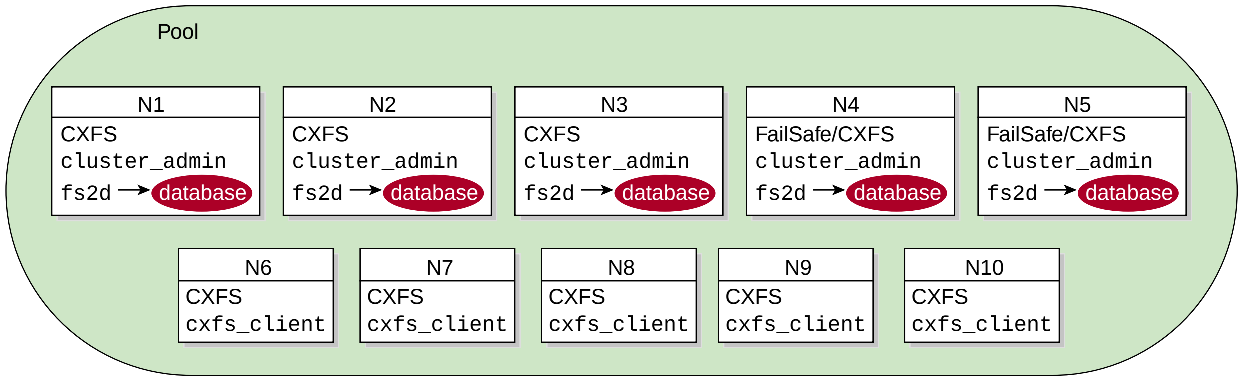


Figure 1-2 Installation Differences

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", page 96.

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 64-bit; the other nodes could be client-only nodes.

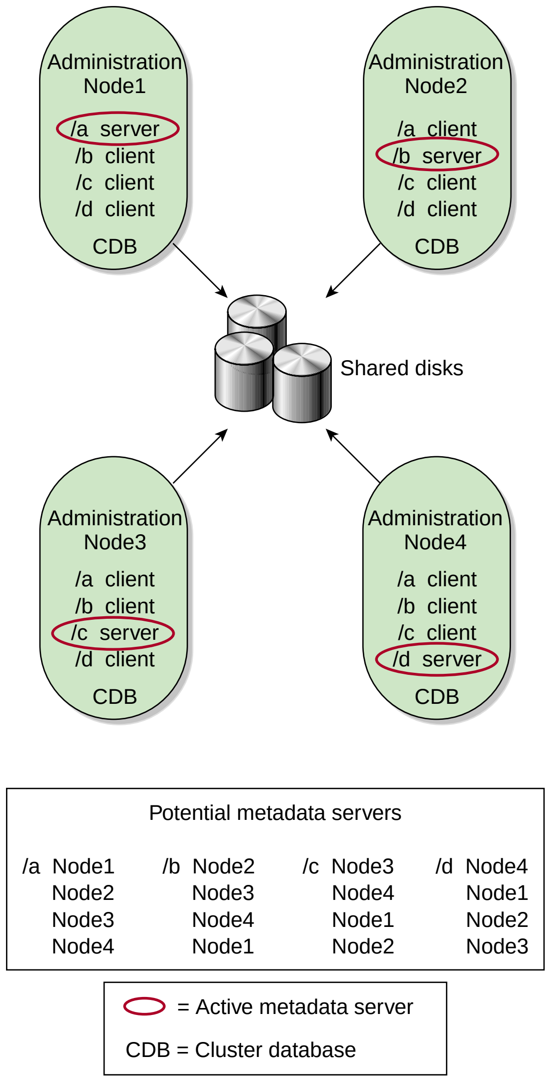


Figure 1-3 Evenly Distributed Metadata Servers

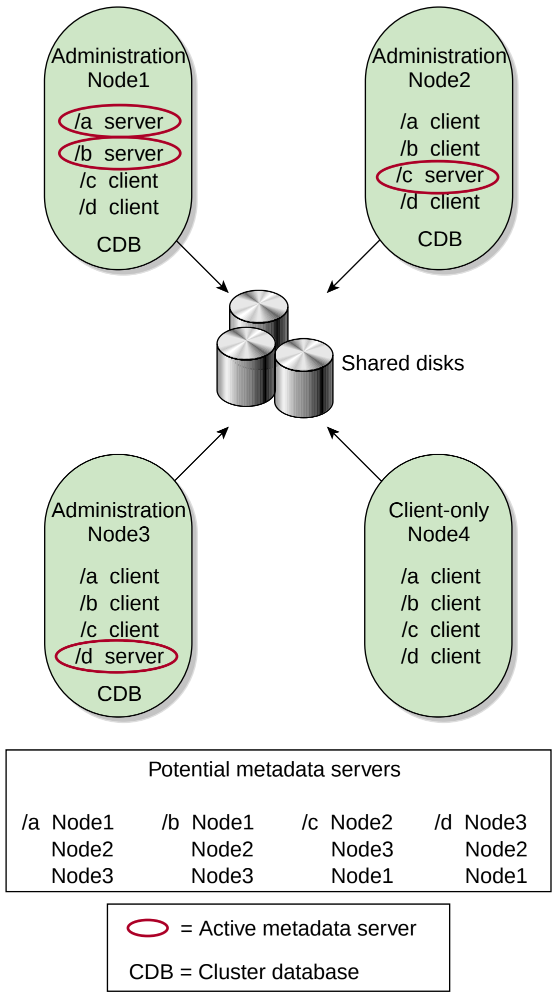


Figure 1-4 Multiple Metadata Servers

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.

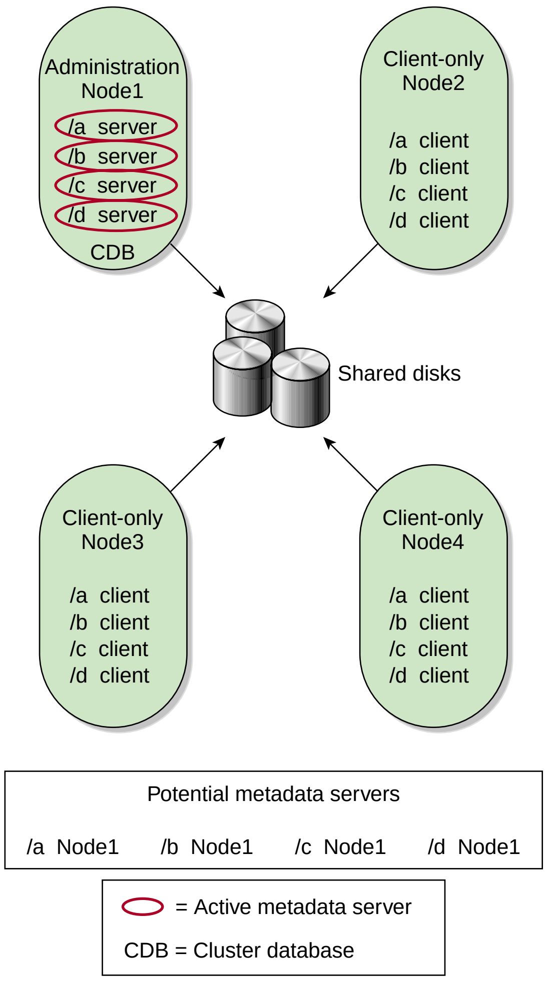


Figure 1-5 One Metadata Server

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.

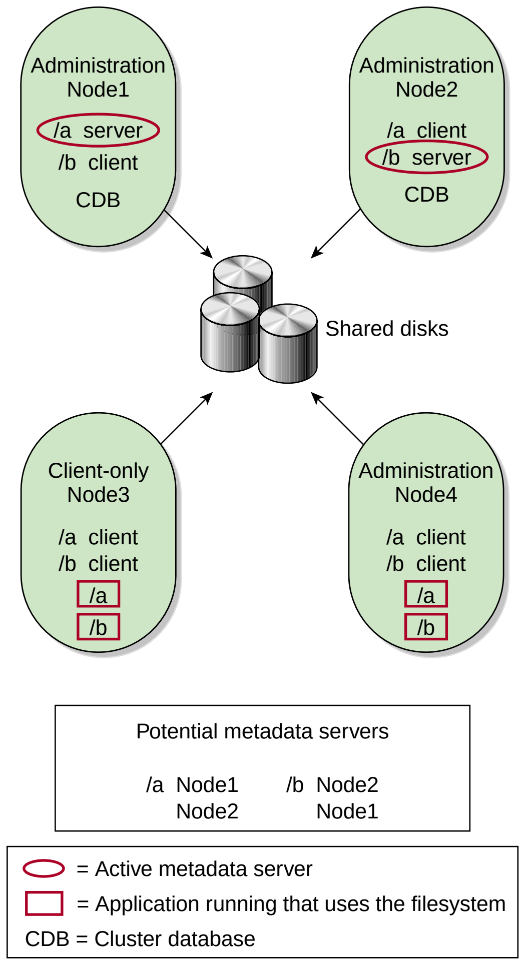


Figure 1-6 Standby Mode

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*. There are the following membership types in a cluster running both CXFS and FailSafe:

- *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.
- *FailSafe membership* is the group of nodes that provide highly available (HA) resources for the cluster.

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

For more information, see Appendix B, "Memberships and Quorums", page 397, and the *SGI InfiniteStorage FailSafe Administrator's Guide*.

Private Network

A *private network* is one that is **dedicated** to cluster communication and is accessible by administrators but not by users. A virtual local area network (VLAN) is not suitable 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 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.

For more information about network segments and partitioning, see Appendix B, "Memberships and Quorums", page 397. For information about using IP filtering for the private network, see Appendix C, "IP Filtering Example for the CXFS Private Network", page 417.

Relocation

Note: Not supported in this release.

To use relocation in standby mode, you must enable relocation on the metadata server (relocation is disabled by default.) To enable relocation, set the `cxfs_relocation_ok` parameter:

- IRIX:
  - Enable:

    ```
    irix# systune cxfs_relocation_ok 1
    ```

  - Disable:

    ```
    irix# systune cxfs_relocation_ok 0
    ```

- Linux 64-bit:
  - Enable:

    ```
    [root@linux64 root]# sysctl -w fs.cxfs.cxfs_relocation_ok=1
    ```

  - Disable:

    ```
    [root@linux64 root]# sysctl -w fs.cxfs.cxfs_relocation_ok=0
    ```

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.

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 metadata server. (FailSafe coexecution only applies to IRIX administration nodes.)
- The system administrator unmounts the CXFS filesystem on an IRIX metadata server. (Unmounting on a Linux 64-bit metadata server does not trigger relocation; the Linux 64-bit server will just return an `EBUSY` flag.)

Recovery

Note: Recovery is supported only on standby nodes.

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.

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.)

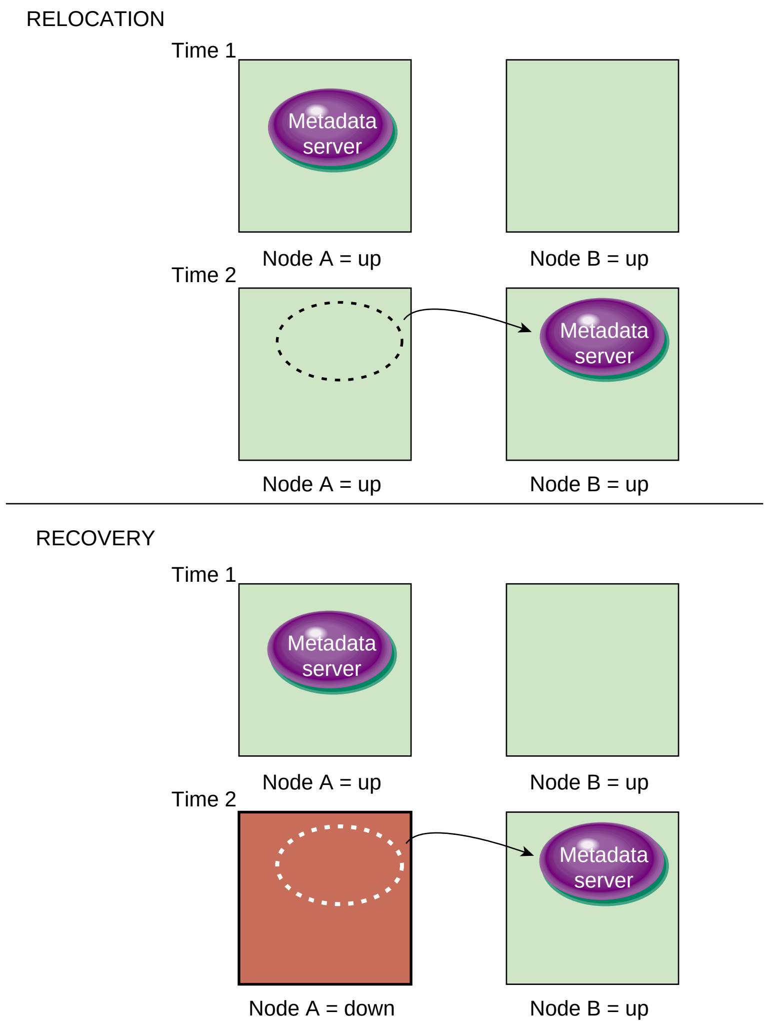


Figure 1-7 Relocation versus Recovery

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 the following nodes because they do not provide system controllers:

  – The following nodes running IRIX:

    - Silicon Graphics Fuel nodes
    - Silicon Graphics Octane nodes
    - Silicon Graphics Octane2 nodes

  – Nodes running other operating systems other than IRIX or Linux 64-bit

  To support I/O fencing, these platforms require a Brocade Fibre Channel switch sold and supported by SGI. The fencing network connected to the Brocade switch must be physically separate from the private heartbeat network.

  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.

- *Serial hardware reset*, which performs a system reset via a serial line connected to the system controller. This method applies only to nodes with system controllers.

- *I/O fencing and serial hardware reset*, which disables access to the SAN from the problem node and then, if the node is successfully fenced, performs an asynchronous reset if the node has a system controller; recovery begins without waiting for reset acknowledgment. This method applies only to nodes with system controllers.

- *CXFS shutdown*, which stops CXFS kernel-based services on the node in response to a loss of CXFS kernel membership. The surviving cluster delays the beginning of recovery to allow the node time to complete the shutdown.

On nodes without system controllers, your only choice for data integrity protection is I/O fencing.

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 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. You would want to use serial hardware reset for I/O protection on a node when CXFS is a primary activity and you want to get it back online fast; for example, a CXFS fileserver. However, I/O fencing cannot return a nonresponsive node to the cluster; this problem will require intervention from the system administrator.

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", page 152, and "Define a Node with `cmgr`", page 202. If you do not define a failure action hierarchy, the default is to perform a serial hardware reset and a CXFS shutdown.

The rest of this section provides more details about I/O fencing and serial hardware resets. For more information, see "Normal CXFS Shutdown", page 273.

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 Brocade Fibre Channel switch and disables the port.

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

The Brocade Fibre Channel switch then blocks the problem node from communicating with the storage area network (SAN) resources via the corresponding HBA. Figure 1-8, page 26, 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", page 177
- "Define a Switch with `cmgr`", page 248

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

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`", page 202, and "Define a Node with the GUI", page 152.

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 Brocade Fibre Channel 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", page 179, and "Lower the I/O Fence for a Node with `cmgr`", page 250.

Caution: If you choose to use I/O fencing, you must understand that 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.

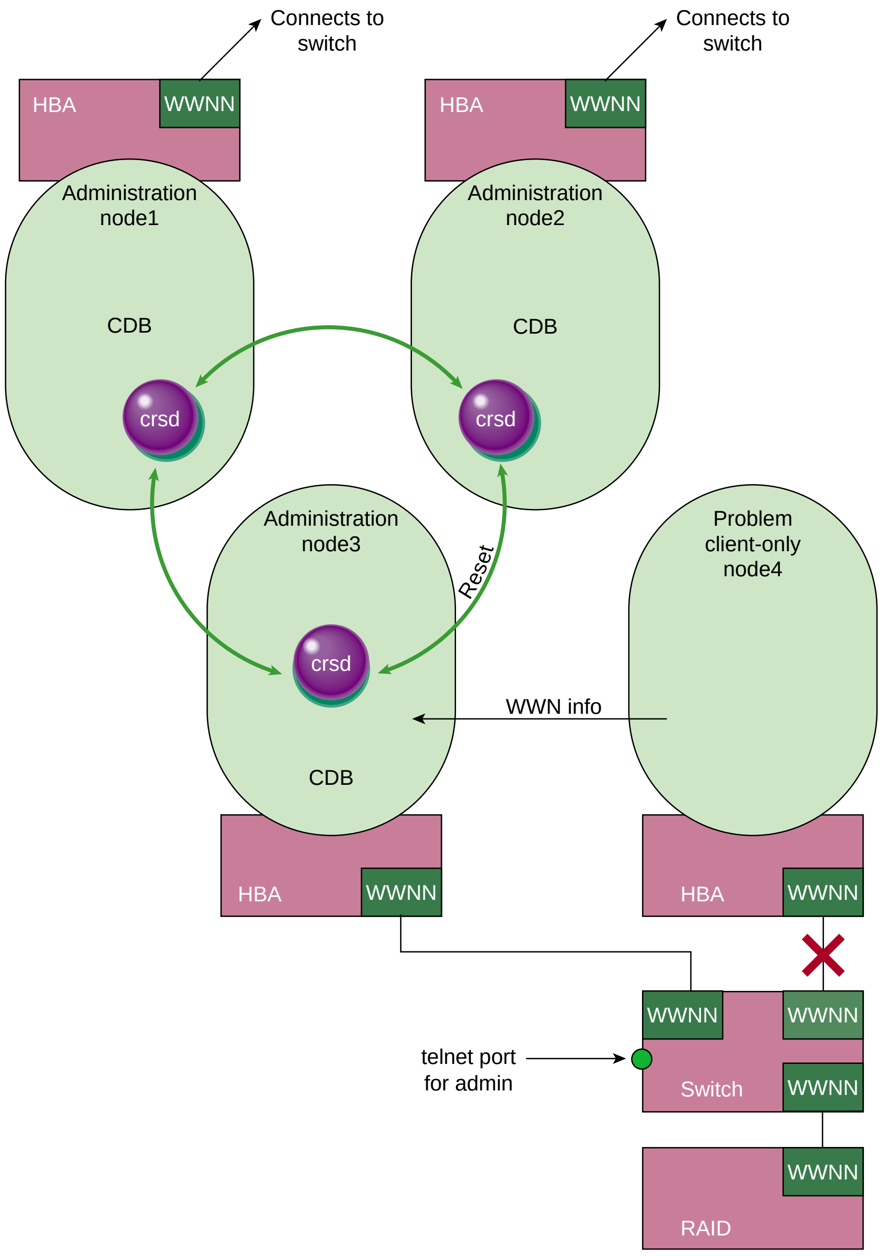


Figure 1-8 I/O Fencing

For more information, see "Switches and I/O Fencing Tasks with the GUI", page 176, and "Switches and I/O Fencing Tasks with `cmgr`", page 248.

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

Serial Hardware Reset

IRIX and Linux 64-bit nodes with system controllers can be reset via a serial line connected to the system controller.

Figure 1-9 shows an example of the CXFS hardware components for a cluster using the serial hardware reset capability and an Ethernet serial port multiplexer.

Note: The serial hardware reset capability or the use of I/O fencing and switches is **mandatory** to ensure data integrity for clusters with only two server-capable nodes and it is highly recommended for all server-capable nodes. Larger clusters should have an odd number of server-capable nodes, or must have serial hardware reset lines or use I/O fencing and switches if only two of the nodes are server-capable. (See "CXFS Recovery Issues in a Cluster with Only Two Server-Capable Nodes ", page 414.)

The reset connection has the same connection configuration as FailSafe; for more information, contact SGI professional or managed services.

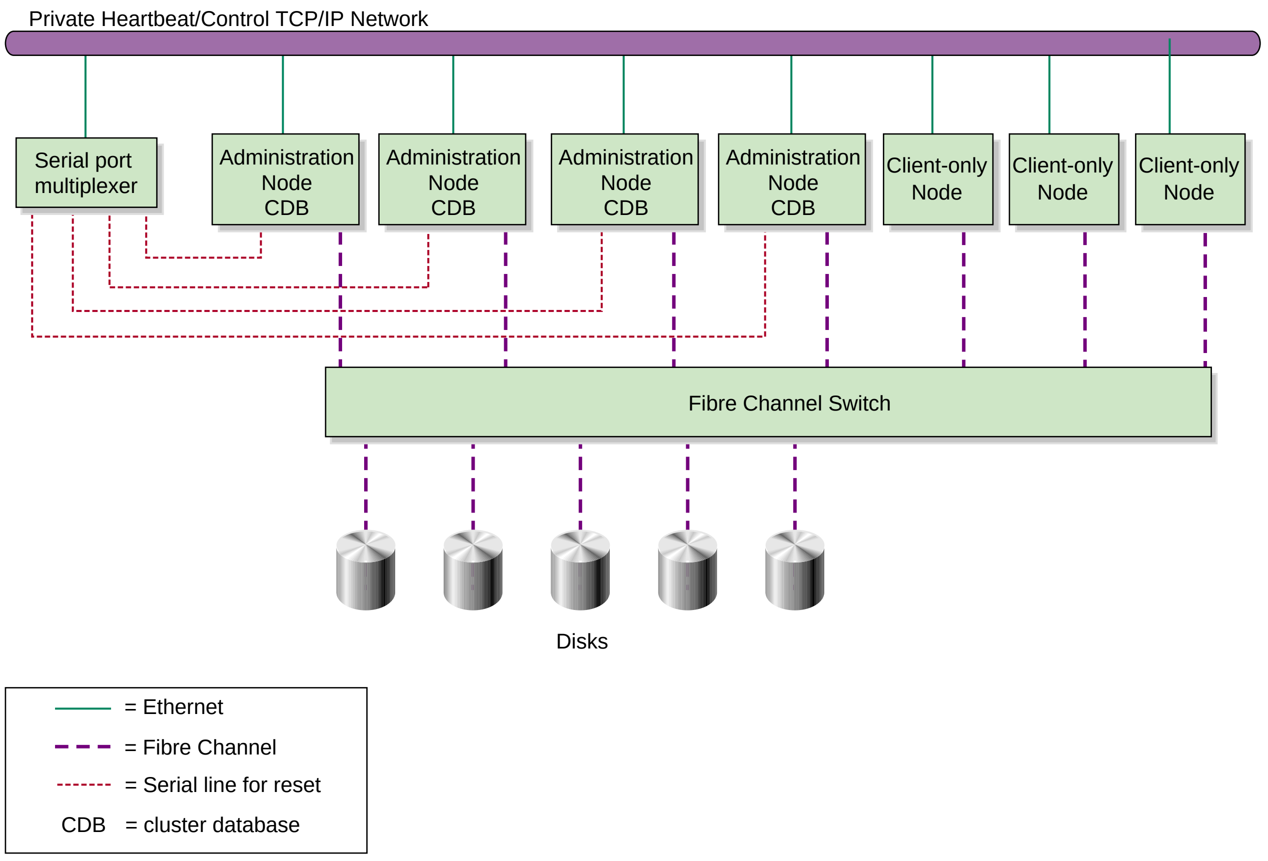


Figure 1-9 Example of a Cluster using Serial Hardware Reset

Nodes that have lost contact with the cluster will forcibly terminate access to shared disks. In the absence of serial hardware reset hardware, this may be sufficient to ensure data integrity on the shared disks, assuming the following:

- The node is able to detect it has lost communication; that is, an error on the node does not prevent it from detecting the loss.
- The node detects the loss in a timely fashion (which it is designed to do); that is, an error on the node does not delay the detection.

However, to ensure data integrity in certain rarely seen error situations, SGI recommends that you use the hardware required for serial hardware reset, especially for clusters with an even number of server-capable nodes. A cluster containing nodes without system controllers also requires the use of I/O fencing to protect data integrity.

The serial hardware reset capability or the use of I/O fencing with switches is **mandatory** to ensure data integrity for clusters with only two server-capable nodes, and it is highly recommended for all server-capable nodes. Larger clusters should have an odd number of server-capable nodes, or must have serial hardware reset lines or use I/O fencing with switches if only two of the nodes are server-capable. Reset is required for FailSafe.

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 serial hardware 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 serial hardware reset or 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 reset or 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 reset or 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 always have at least the tiebreaker node or reset or fencing capability configured. 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.

The serial hardware reset connections take the following forms:

- Clusters of two nodes can be directly connected with serial hardware reset lines.
- Clusters of three or more nodes should be connected with a serial port multiplexer. Each node is defined to have an *owner host*, which is the node that has the ability to reset it.

For more information, contact SGI professional or managed services.

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.
- 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.
- 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.

- 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", page 270.)
- 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 "NFS Export Scripts", page 264.

For more information, see "Flow of Metadata for Reads and Writes", page 392.

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.

Hardware and Software Support

This section discusses the following:

- "Requirements"
- "Maximum Filesystem Size and Offset within a File", page 36
- "Compatibility", page 36
- "Recommendations", page 37

Requirements

CXFS requires the following:

- All server-capable administration nodes must run the same type of operating system, either IRIX or SGI ProPack for Linux 64-bit. This guide supports the IRIX 6.5.22 and SGI ProPack 2.3 for Linux. (Additional release levels may be used in the cluster while performing an upgrade; see "Rolling Upgrades", page 96.)
- A supported SAN hardware configuration using the following platforms for metadata servers:
  - IRIX systems with system controllers (which means that the nodes can use **either** serial hardware reset or I/O fencing for data integrity protection):
    - SGI Origin 300 server
    - SGI Origin 3000 series
    - SGI 2000 series
    - SGI Origin 200 server
    - Silicon Graphics Onyx2 system
    - Silicon Graphics Onyx 3000 series
  - IRIX systems **without** system controllers (which means that the nodes **require** I/O fencing for data integrity protection):
    - Silicon Graphics Fuel visual workstation
    - Silicon Graphics Octane system
    - Silicon Graphics Octane2 system
  - An SGI Altix 3000 server

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. CXFS does **not** support the Silicon Graphics O2 workstation and therefore it cannot be a CXFS serial hardware reset server. CXFS does not support JBOD.

- In a CXFS multiOS cluster, CXFS client-only platforms as defined in the *SGI InfiniteStorage CXFS MultiOS Client-Only Guide*. These platforms do not contain system controllers.
- 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 hardware reset lines or I/O fencing with Brocade Fibre Channel switches sold and supported by SGI. One of these solutions is mandatory for clusters with only two server-capable nodes. Larger clusters should have an odd number of server-capable nodes.
- A Brocade Fibre Channel switch sold and supported by SGI is mandatory to support I/O fencing. Nodes without system controllers require I/O fencing for data integrity protection. See Chapter 3, "Brocade Fibre Channel Switch Verification", page 47.
- At least one QLogic 2310 or QLogic 2342 Fibre Channel host bus adapter (HBA)
- RAID hardware as specified in Chapter 2, "SGI RAID", page 45.
- 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. 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.
- A FLEXlm license key for CXFS. Linux 64-bit 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. Only those nodes that will access the mirrored volume must be licensed. For information about purchasing this license, see your sales representative.

Note: 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 license for all the partitions in the system. For more information, see the *Start Here/Welcome* and the following web page: `http://www.sgi.com/support/licensing/partitionlic.html`.

- 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 48 nodes, of which as many as 16 can be server-capable administration nodes. (See "Initial Configuration Requirements and Recommendations", page 103.)

A cluster in which both CXFS and FailSafe are run (known as *coexecution*) is supported with a maximum of 48 nodes, as many as 8 of which can run FailSafe. The administration nodes must run IRIX; FailSafe is not supported on Linux 64-bit nodes. Even when running with FailSafe, there is only one pool and one cluster. See "Overview of FailSafe Coexecution", page 38, for further configuration details.

Requirements Specific to IRIX

- The IRIX nodes in a cluster containing nodes running other operating systems must be running 6.5.16f or later.
- 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.

Requirements Specific to Linux 64-bit

Using a Linux 64-bit node to support CXFS requires the following:

- CXFS 3.0 for SGI Altix 3000 running SGI ProPack 2.3 for Linux

- SGI ProPack 2.3 for Linux
- An SGI Altix server
- At least one QLogic 2310 or QLogic 2312 Fibre Channel host bus adapter (HBA)

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 64-bit CXFS filesystem, this is not recommended.

Maximum Filesystem Size and Offset within a File

The maximum size of an XFS filesystem is 2^{64} bytes (about 18 million terabytes). The maximum offset within an XFS file is 2^{63}-1 bytes (about 9 million terabytes).

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", page 305.
- FailSafe (coexecution). See the "Overview of FailSafe Coexecution", page 38, and the *SGI InfiniteStorage FailSafe Administrator's Guide*.
- IRISconsole; see the *IRISconsole Administrator's Guide*. (CXFS does **not** support the Silicon Graphics O2 workstation as a CXFS node.)
- A serial port multiplexer used for the serial hardware reset capability.

Recommendations

SGI recommends the following when running CXFS:

- You should isolate the power supply for the Brocade Fibre Channel switch from the power supply for a node and its system controller (MSC, MMSC, L2, or L1). 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 of multiple clusters being formed due to a lack of serial hardware reset (also known as *split-brain syndrome*).
- If you use I/O fencing, SGI recommends that you use a switched network of at least 100baseT.
- Only those nodes that you want to be potential metadata servers should be CXFS administration nodes (installed with the `cluster_admin` software product). CXFS client administration nodes should only be used when necessary for coexecution with IRIS FailSafe. All other nodes should be client-only nodes (installed with `cxfs_client`).

  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 network switch rather than a hub for performance and control.
- All nodes should be on the same physical network segment. Two clusters should not share the same private network.
- A production cluster should be configured with a minimum of three server-capable nodes.
- If you want to run CXFS and Trusted IRIX, have all nodes in the cluster run 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.
- As for any case with long running jobs, you should use the IRIX checkpoint and restart feature. For more information, see the `cpr` man page.

For more configuration and administration suggestions, see "Initial Configuration Requirements and Recommendations", page 103.

Overview of FailSafe Coexecution

CXFS allows groups of computers to coherently share large amounts of data while maintaining high performance. The FailSafe product provides a general facility for providing highly available services.

You can therefore use FailSafe in a CXFS cluster (known as *coexecution*) to provide highly available 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.

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 1-10 depicts the contents of a coexecution cluster database.

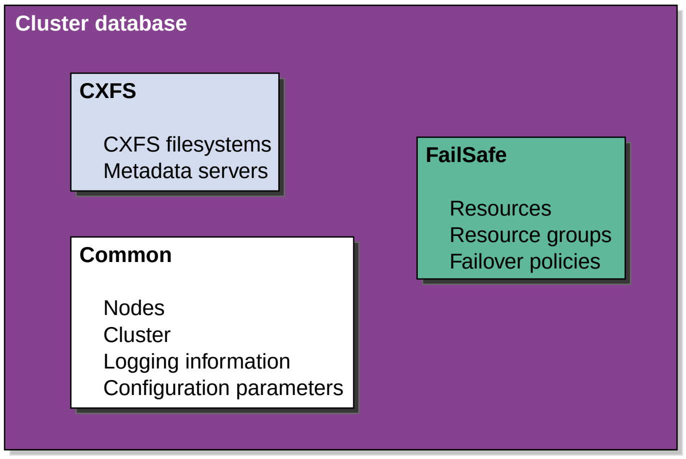


Figure 1-10 Contents of a Coexecution Cluster Database

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 `cluster_status` and `clconf_info` commands.

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", page 297.

Cluster Manager 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`)
  - `cluster_status`
  - `clconf_info`
- 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", page 130.

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`, `cluster_status`, 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", page 129
- "`cmgr` Overview", page 196
- "Creating a `cmgr` Script Automatically", page 255

- Chapter 15, "Monitoring Status", page 307

Overview of the Installation and Configuration Steps

This section provides an overview of the installation, verification, and configuration steps for IRIX and for Linux 64-bit for SGI ProPack on SGI Altix 3000 systems.

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/cxfslicense`

These commands provide all of the services needed to include an IRIX or a Linux 64-bit client-only node.

For more information, see the `cxfs_client` man page.

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_shutdown
/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 6.5.22 according to the *IRIX 6.5 Installation Instructions* (if not already done).

2. Install and verify the SGI RAID. See Chapter 2, "SGI RAID", page 45
3. Install and verify the Brocade Fibre Channel switch. See Chapter 3, "Brocade Fibre Channel Switch Verification", page 47.
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 4, "Obtaining CXFS and XVM FLEXlm Licenses", page 53.
5. Prepare the node, including adding a private network. See "Adding a Private Network", page 57.
6. Install the CXFS software. See Chapter 6, "IRIX CXFS Installation", page 61.
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", page 149.

Linux 64-bit on SGI Altix Overview

Following is the order of installation and configuration steps for a Linux 64-bit node:

1. Read the release notes `README` file for the Linux 64-bit platform to learn about any late-breaking changes in the installation procedure.
2. Install the SGI ProPack 2.3 for Linux release, according to the directions in the SGI ProPack documentation. Ensure that you select the `SGI Licensed` package group for installation. If the system will be a CXFS administration node, also ensure that you select the `SGI Cluster Infrastructure` package group.
3. Install and verify the SGI RAID. See Chapter 2, "SGI RAID", page 45
4. Install and verify the Brocade Fibre Channel switch. See Chapter 3, "Brocade Fibre Channel Switch Verification", page 47.
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 4, "Obtaining CXFS and XVM FLEXlm Licenses", page 53.

6. Prepare the node, including adding a private network. See "Adding a Private Network", page 57.
7. Install the CXFS software. See Chapter 7, "Linux 64-bit CXFS Installation", page 73.
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", page 149.

SGI RAID

The SGI RAID will be initially installed and configured by SGI personnel.

Supported SGI RAID Hardware

SGI supports the following RAID hardware:

- SGI TP9500
- SGI TP9400
- SGI TP9100

Required SGI RAID Firmware

This section describes the required RAID firmware.

Required SGI TP9500 RAID Firmware

The TP9400/TP9500 6.0 CD contains the required controller firmware and NVSRAM files for the 5884 unit. The 05.30.*xx.xx* code or later must be installed.

Note: By default, the TP9500 supports 32 logical units (LUNs). If additional LUNs are required, you must obtain a separate software-enabling key; this key will support a maximum of 2048 LUNs in separate partitions, which requires that the Fibre Channel ports be mapped to a partition. Contact your SGI sales representative for the SGI software partitioning key.

Required SGI TP9400 RAID Firmware

The TP9400 4.0 CD contains the required controller firmware and NVSRAM files for the 4774 or 4884 units:

- If you have a 4774 unit, the 04.01.*xx.xx* code or later must be installed according to FCO 1056.

- If you have a 4884 unit, the 04.02.*xx.xx* code or later is installed by default.

Note: By default, the TP9400 supports 32 LUNs. If additional LUNs are required, you must obtain a separate software-enabling key; this key will support a maximum number of LUNs in separate partitions, which requires that the Fibre Channel ports be mapped to a partition. The maximum depends upon the code installed:

- 04.01.*xx.xx* code supports a maximum of 128 LUNs
- 04.02.*xx.xx* code supports a maximum of 128 LUNs
- 05.30.*xx.xx* code supports a maximum of 1024 LUNs

Contact your SGI sales representative for the SGI software partitioning key.

Required SGI TP9100 RAID Firmware

The TP9100 4.0 CD contains the required version 7.75 controller firmware and NVSRAM files for the 1-Gbit TP9100. The TP9100 5.0 CD contains the required version 8.29 firmware and NVSRAM files for the 2-Gbit TP9100. The TP9100 is limited to 64 host connections.

RAID Firmware Verification

To verify that the SGI RAID is properly installed and ready for use with CXFS, you can dump the RAID's profile and verify the controller software revisions.

For More Information

The following documentation is used to install and verify the RAID:

- *TPM Installation Instructions and User's Guide for TP9100*
- *SGI TP9400 and SGI TP9500 Software Concepts Guide*
- *SGI TP9400 and SGI TP9500 RAID Owner's Guide*
- *SGI TP9400 and SGI TP9500 RAID Administration Guide*

Chapter 3

Brocade Fibre Channel Switch Verification

In order to protect data integrity, nodes without system controllers must use the *I/O fencing* feature, which isolates a problem node so that it cannot access I/O devices and therefore cannot corrupt data in the shared CXFS filesystem.

This feature can only be used with a Brocade Fibre Channel switch supported by SGI; therefore, the Brocade switch is a required piece of hardware in a multiOS cluster.

The Brocade Fibre Channel switches will be initially installed and configured by SGI personnel. You can use the information in this chapter to verify the installation.

Required Brocade Fibre Channel Switch Firmware and License

This release supports Brocade Silkworm Fibre Channel switches that are supported by SGI:

- 2400 (8-port)
- 2800 (16-port)
- 3200 (8-port, 2-Gbit)
- 3800 (16-port, 2-Gbit)
- 3900 (32-port, 2-Gbit)
- 12000 (32-, 64-, or dual 64-port, 2-Gbit)

All Brocade switches contained within the SAN fabric must have the appropriate Brocade license key installed. The following firmware is required:

- 2400 and 2800 switches: 2.6.0d or later
- 3200 and 3800 switches: 3.0.2c or later
- 3900 and 12000 switches: v4.0.2c or later

If the current firmware level of the switches must be upgraded, please contact your local SGI service representative or customer support center.

The Brocade switch must be configured so that its Ethernet interface is accessible from all administration nodes using `telnet`. The fencing network connected to the Brocade switch must be physically separate from the private heartbeat network.

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

The 3900 and 12000 series switches permit multiple `telnet` sessions. However, CXFS I/O fencing requires a `telnet` lockout for global mutual exclusion when a fencing race occurs. Therefore, you must configure the 3900 and 12000 series switches to set the maximum allowed simultaneous `telnet` sessions for the `admin` user to one. Only the 3900 and 12000 switches require this configuration (other Brocade switch models are shipped with the required restrictions configured by default). See "Configuring the Brocade Silkworm 3900", page 49, and "Configuring the Brocade Silkworm 12000", page 50.

Verifying the Brocade License

To verify the Brocade license, log into the switch as user `admin` and use the `licenseshow` command, as shown in the following example:

```
brocade:admin> licenseshow
dcRyzyScSedSz0p:
    Web license
    Zoning license
    SES license
    Fabric license
SQQQSyddQ9TRRdUP:
    Release v2.2 license
```

Verifying the Brocade Switch Firmware Version

To verify the firmware version, log into the switch as user `admin` and use the `version` command, as shown in the following example:

```
workstation% telnet brocade1
Trying 169.238.221.224...
Connected to brocade1.acme.com
Escape character is '^]'.
```

```
Fabric OS (tm)  Release v2.6.0d

login: admin
Password:
brocade1:admin> version
Kernel:     5.4
Fabric OS:  v2.6.0d                  <== Firmware Revision
Made on:    Fri May 17 16:33:09 PDT 2002
Flash:      Fri May 17 16:34:55 PDT 2002
BootProm:   Thu Jun 17 15:20:39 PDT 1999
brocade1:admin>
```

Configuring the Brocade Silkworm 3900

To limit the maximum allowed simultaneous `telnet` sessions for the `admin` user to one on the Brocade Silkworm 3900, do the following:

1. Connect to the switch via the `telnet` command and login as `root`.

2. Issue the `sync` command to avoid filesystem corruption:

   ```
   # sync
   ```

3. Edit the `/etc/profile` file to change the `max_telnet_sessions` from `2` to `1` and place the information in a new file. For example:

```
# cd /etc
# sed -e 's/max_telnet_sessions=2/max_telnet_sessions=1/' profile >profile.new
```

4. Distribute the edited `profile` file to both partitions on both central processors. For example:

   ```
   # cp profile.new profile
   # cp profile.new /mnt/etc/profile
   ```

5. Issue the `sync` command again to avoid filesystem corruption:

   ```
   # sync
   ```

Configuring the Brocade Silkworm 12000

To limit the maximum allowed simultaneous `telnet` sessions for the `admin` user to one on the Brocade Silkworm 12000, do the following:

1. Connect to the switch via the `telnet` command and login as `root`.
2. Use the `haShow` command to make sure that both central processors are up. This is indicated by the message `Heartbeat Up` within the output of the `haShow` command. If it is not up, wait a couple of minutes and run `haShow` again to check for the status.
3. Issue the `sync` command on the filesystems to avoid filesystem corruption:

   ```
   # rsh 10.0.0.5 sync
   # rsh 10.0.0.6 sync
   ```

4. Edit the `/etc/profile` file to change the `max_telnet_sessions` from `2` to `1` and place the information in a new file. For example:

```
# cd /etc
# sed -e 's/max_telnet_sessions=2/max_telnet_sessions=1/' profile >profile.new
```

5. Distribute the new profile to both partitions and central processors. For example:

   ```
   # rcp /etc/profile.new 10.0.0.5:/etc/profile
   # rcp /etc/profile.new 10.0.0.5:/mnt/etc/profile
   # rcp /etc/profile.new 10.0.0.6:/etc/profile
   # rcp /etc/profile.new 10.0.0.6:/mnt/etc/profile
   ```

6. Issue the `sync` command again to avoid filesystem corruption:

   ```
   # rsh 10.0.0.5 sync
   # rsh 10.0.0.6 sync
   ```

Changing the Brocade FC Cable Connections

To change Brocade Fibre Channel cable connections used by nodes in the CXFS cluster, do the following:

1. Cleanly shut down CXFS services on the nodes affected by the cable change using either the CXFS GUI or the `cmgr` command.
2. Rearrange the cables as required.
3. Restart CXFS services.

4. Reconfigure I/O fencing if required. You must perform this step if I/O fencing is enabled on the cluster and if you added/removed any Brocade switches. You must use the CXFS GUI or the `cmgr` command to add/remove switches from the CXFS configuration as required.

5. If any CXFS client nodes are connected to a new (or different) Brocade switch, restart CXFS services on those nodes. This will ensure that the administration nodes can correctly identify the Brocade ports used by all clients.

Chapter 4

Obtaining 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 SGI sales representative.

This section discusses the following:

- "Obtaining the Host Information Required for the License"
- "Obtaining and Installing the Licenses", page 55
- "License Verification", page 55
- "For More Information", page 56

Obtaining 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, execute the following IRIX commands:

```
/usr/bsd/hostid
/usr/bsd/hostname
```

For example:

```
irix# /usr/bsd/hostid
0x80a2e84f
irix# /usr/bsd/hostname
cxfsirix1
```

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.

Linux 64-bit Host Information

For SGI Altix systems, you must obtain the host identifier, hostname, serial number, and CPU count of the system on which you will run CXFS. Execute the following commands:

```
cat /proc/sgi_sn/system_serial_number
/sbin/ifconfig eth0
/bin/hostname
hinv -c processor
```

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

The host identifier is the value for the `HWaddr` field, minus the colons.

For example:

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

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

[root@linux64 root]# /sbin/ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:E0:81:24:77:D1
          inet addr:128.162.240.135  Bcast:128.162.240.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:17941247 errors:2 dropped:0 overruns:0 frame:2
          TX packets:16373834 errors:10 dropped:0 overruns:0 carrier:10
          collisions:0 txqueuelen:100
          RX bytes:1539518033 (1468.1 Mb)  TX bytes:1573522873 (1500.6 Mb)
          Interrupt:19 Base address:0xb880 Memory:fe3fe000-fe3fe038[
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 `00E0812477D1` 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.

Obtaining and Installing the Licenses

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.

License Verification

Use the `cxfslicense` command with the `-d` option to verify that the FLEXlm licenses have been installed properly.

For example, if the CXFS license is properly installed on an Altix 3000 system, you will see the following:

```
[root@linux64 root]# /usr/cluster/bin/cxfslicense -d
XVM_STD_IPF license granted.
XVM_PLEX_IPF license granted.
CXFS_IPF license
```

If you do not have the CXFS license properly installed, you will see an error message. For example, if the license is not installed on a client-only node, you will see the following on the console and in the `cxfs client` logs when trying to run CXFS:

```
May 12 14:40:17 cxfs_client: cis_main FATAL: cxfs_client failed the CXFS
license check. Use the cxfslicense command to diagnose the license
problem.
May 12 14:40:17 cxfs_client: FATAL: aborting on fatal error
```

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

For More Information

For more information about licensing, see the following webpage:

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

If you cannot use the web key generation page, you can contact the SGI order desk at 800 800 4SGI (800 800 4744).

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

Chapter 5

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:

- "Adding a Private Network", page 57
- "Verifying the Private and Public Networks", page 59

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 (such as NFS) are going to resolve names for their client server programs, 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 64-bit 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 Required for CXFS Connectivity Diagnostics", page 72, and "Linux 64-bit Modifications Required for CXFS Connectivity Diagnostics", page 80.

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.

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 complete set of CXFS cluster services and the cluster database) or a *client-only node*. 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 16, "Troubleshooting", page 323, before attempting to install and configure a CXFS cluster. If you are using coexecution with FailSafe, see the *SGI InfiniteStorage FailSafe Administrator's Guide*. If you are using a multiOS cluster, see the *SGI InfiniteStorage CXFS MultiOS Client-Only Guide*.

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", page 68.

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

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 6.5.22 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. Install the CXFS license key. When you order a product that requires a license key, the key will be sent to you automatically through e-mail by the order desk along with instructions for installing it. If you do not have this information, contact SGI or your local support provider.

      If the license is properly installed, you will see the following output from the `cxfslicense` command after the CXFS software installation is complete:

      ```
      # /usr/cluster/bin/cxfslicense -d
      CXFS license granted.
      ```

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

      ```
      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/cluster restart'.
      ```

      An error such as the following example will appear in the `SYSLOG` 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 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/partitionlic.html`.

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

b. Insert CD-ROM #1 into the CD drive.

c. 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.

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

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

e. 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
```

f. 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).

g. 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 the step documented by "Rebooting the System", page 92.

h. Insert CD-ROM #3 into the CD drive.

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

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

j. 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_cxfs
   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_cxfs.man.pages
sysadm_cxfs.man.relnotes
sysadm_cxfs.sw.client
sysadm_cxfs.sw.desktop
sysadm_cxfs.sw.server
sysadm_cxfs.sw.web
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
```

k. Insert the *CXFS 3.0 IRIX Server and Client for IRIX 6.5.22* 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.

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

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

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

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

n. Install the CXFS software:

```
Inst> keep *
Inst> install 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
```

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 6.5.22*:

```
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 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 IRIX 6.5.22 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 6.5.22 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.

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", page 61.

Note: For information about installing software for a node running an operating system other than IRIX, see the *SGI InfiniteStorage CXFS MultiOS Client-Only Guide*.

To install the required IRIX software, do the following:

1. On each IRIX client-only node in the pool, upgrade to IRIX 6.5.22 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. Install the CXFS license key. When you order a product that requires a license key, the key will be sent to you automatically through e-mail by the order desk along with instructions for installing it. If you do not have this information, contact SGI or your local support provider.

      If the license is properly installed, you will see the following output from the `cxfslicense` command:

      ```
      # /usr/cluster/bin/cxfslicense -d
      CXFS license granted.
      ```

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

      ```
      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/cluster restart'.
      ```

      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/partitionlic.html`.

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

   b. Insert CD-ROM #1 into the CD drive.

   c. 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.

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

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

e. 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
```

f. 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).

g. Insert the *CXFS 3.0 IRIX Server and Client for IRIX 6.5.22* IRIX 6.5.22 CXFS CD into the CD drive.

h. 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.

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

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

j. 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 the step documented by "Rebooting the System", page 92.

IRIX Modifications Required 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.

Chapter 7

Linux 64-bit 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 Altix systems running SGI ProPack for Linux 64-bit, CXFS supports either an *administration node* (containing the complete set of CXFS cluster services and the cluster database) or a *client-only node*. The software you install on a node determines the node type.

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 64-bit Limitations and Considerations"
- "Linux 64-bit Administration Software Installation", page 75
- "Linux 64-bit Client-only Software Installation", page 78

After completing these steps, see Chapter 9, "Initial Configuration of the Cluster", page 103. For details about specific configuration tasks, see Chapter 10, "Reference to GUI Tasks for CXFS", page 129, and Chapter 11, "Reference to `cmgr` Tasks for CXFS", page 195.

You should read through this entire book, especially Chapter 16, "Troubleshooting", page 323, before attempting to install and configure a CXFS cluster. If you are using coexecution with IRIS FailSafe, see the *SGI InfiniteStorage FailSafe Administrator's Guide*. If you are using a multiOS cluster, see the *SGI InfiniteStorage CXFS MultiOS Client-Only Guide*.

Linux 64-bit Limitations and Considerations

The following limitations and considerations apply to any Linux 64-bit node (client-only or administration):

- The maximum block size supported is 16 KB, determined by the kernel page size. (XFS uses a default block size of 4 KB unless overridden by an administrator to a different block-size value, for example 8 KB or 16 KB.)
- Due to Linux 64-bit kernel limitations, if a filesystem was **ever** mounted with the `inode64` mount option, it cannot be used with Linux CXFS because it may have 64-bit inodes already on the filesystem.
- CXFS filesystems with XFS version 1 directory format cannot be mounted on Linux 64-bit nodes.
- All CXFS files have UNIX mode bits (read, write, and execute) and optionally an access control list (ACL). For details, see the `chmod` and `setfacl` man pages.

  SGI ProPack file utilities do not provide ACL support, so commands such as `ls` and `cp` will not show or preserve ACLs. However, the commands in the `acl` package will allow manipulation of the ACLs of files on CXFS filesystems.

Client-only Linux 64-bit 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 64-bit 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
```

Linux 64-bit 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.

Linux 64-bit 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 Linux 64-bit 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 64-bit Client-only Software Installation", page 78.

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 Linux 64-bit administration node:

1. Read the `README` file for the Linux 64-bit platform to learn about any late-breaking changes in the installation procedure.
2. Install SGI ProPack 2.3 for Linux release, according to the directions in the SGI ProPack documentation. Ensure that you select the `SGI Licensed` and `SGI Cluster Infrastructure` package groups.
3. Ensure that the necessary XVM and cluster infrastructure packages from the *SGI ProPack v2.3 for Linux - Proprietary Software* CD have been installed:
   - XVM:

```
[root@linux-64 root]# rpm -q xvm-cmds
xvm-cmds-2.3-sgi230c1
```

If this package is not installed, install the following RPMs from the *SGI ProPack v2.3 for Linux - Proprietary Software* CD:

```
[root@linux-64 PP_CDROM]# rpm -Uvh xvm-cmds-2.3-sgi230c1.ia64.rpm  \
xvm-standalone-module-2.3-sgi230c1.ia64.rpm
Preparing...                ########################################### [100%]
   1:xvm-standalone-module  ########################################### [ 50%]
   2:xvm-cmds               ########################################### [100%]
```

Note: The `xvm-standalone-module` RPM is installed at this stage to satisfy package dependencies. It will be replaced in a subsequent step.

- Cluster infrastructure:

```
[root@linux-64 PP_CDROM]# rpm -qa | grep "cluster\|sysadm"
cluster_control-2.3-sgi230c1
cluster_services-2.3-sgi230c1
cluster_admin-2.3-sgi230c1
sysadm_xvm-client-2.1-sgi230c1
sysadm_xvm-server-2.1-sgi230c1
sysadm_cluster_base-client-3.0-sgi230c1
sysadm_base-lib-3.0-sgi230c1
sysadm_base-tcpmux-3.0-sgi230c1
sysadm_cluster_base-server-3.0-sgi230c1
sysadm_xvm-web-2.1-sgi230c1
sysadm_base-server-3.0-sgi230c1
sysadm_base-client-3.0-sgi230c1
```

If these packages are not installed, install them from the *SGI ProPack v2.3 for Linux - Proprietary Software* CD:

```
[root@linux-64 CXFS_CDROM]# rpm -Uvh cluster_* sysadm_*
Preparing...                ########################################### [100%]
   1:cluster_admin          ########################################### [  8%]
   2:sysadm_base-client     ########################################### [ 16%]
   3:sysadm_base-lib        ########################################### [ 25%]
   4:cluster_control        ########################################### [ 33%]
   5:cluster_services       ########################################### [ 41%]
   6:sysadm_base-tcpmux     ########################################### [ 50%]
   7:sysadm_base-server     ########################################### [ 58%]
   8:sysadm_cluster_base-cli########################################### [ 66%]
   9:sysadm_cluster_base-ser########################################### [ 75%]
  10:sysadm_xvm-client      ########################################### [ 83%]
  11:sysadm_xvm-server      ########################################### [ 91%]
  12:sysadm_xvm-web         ########################################### [100%]
```

4. Install the CXFS kernel libraries from the *CXFS 3.0 Altix Server/Client and XVM Plexing for SGI ProPack 2.3* CD:

```
[root@linux-64 CXFS_CDROM]# rpm -Uvh cxfs-modules-3.0-sgi230a12.ia64.rpm
Preparing...                ########################################### [100%]
   1:cxfs-modules           ########################################### [100%]
```

5. Install the CXFS application binaries, documentation, and support tools from the *CXFS 3.0 Altix Server/Client and XVM Plexing for SGI ProPack 2.3* CD:

```
[root@linux-64 CXFS_CDROM]# rpm -Uvh cxfs_util-3.0-sgi230a4.ia64.rpm \
cxfs_cluster-3.0-sgi230c1.ia64.rpm \
cxfs_docs-3.0-sgi230c1.ia64.rpm
Preparing...                ########################################### [100%]
   1:cxfs_util              ########################################### [ 33%]
   2:cxfs_cluster           ########################################### [ 66%]
   3:cxfs_docs              ########################################### [100%]
```

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

6. Install the CXFS graphical user interface (GUI) packages from the *CXFS 3.0 Altix Server/Client and XVM Plexing for SGI ProPack 2.3* CD:

```
[root@linux-64 CXFS_CDROM]# rpm -Uvh sysadm_cxfs-client-2.5-sgi230a3.ia64.rpm \
sysadm_cxfs-server-2.5-sgi230a3.ia64.rpm \
sysadm_cxfs-web-2.5-sgi230a3.ia64.rpm

Preparing...                ########################################### [100%]
   1:sysadm_cxfs-client     ########################################### [ 33%]
   2:sysadm_cxfs-server     ########################################### [ 66%]
   3:sysadm_cxfs-web        ########################################### [100%]
```

7. Reboot the system.

Caution: If XVM standalone was in use prior to CXFS installation, you **must** reboot the system before starting CXFS services, or else the system will panic.

Verifying the Linux 64-bit Administration Installation

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

For example, the following output indicates that the CXFS packages are installed properly:

```
[root@linux-64 root]# rpm -qa | grep cxfs
cxfs-modules-3.0-sgi230a12
cxfs_util-3.0-sgi230a4
cxfs_cluster-3.0-sgi230c1
cxfs_docs-3.0-sgi230c1
sysadm_cxfs-client-2.5-sgi230a3
sysadm_cxfs-server-2.5-sgi230a3
sysadm_cxfs-web-2.5-sgi230a3
```

To verify the license, use the following command:

```
[root@linux root]# /usr/cluster/bin/cxfslicense -d
CXFS license granted.
```

Linux 64-bit 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: Version numbers shown here are examples; your installed system may differ.

Linux 64-bit Client-Only Installation Overview

Installing the Linux 64-bit client requires approximately 60 MB of space, depending upon the packages installed at your site.

To install the required software on a Linux 64-bit node, SGI personnel will do the following:

1. Read the `README` file for the Linux 64-bit platform to learn about any late-breaking changes in the installation procedure.
2. Install SGI ProPack 2.3 for Linux release, according to the directions in the SGI ProPack documentation. Ensure that you select the `SGI Licensed` package group.
3. Ensure that the necessary XVM package from the *SGI ProPack v2.3 for Linux - Proprietary Software* CD has been installed:

```
[root@linux-64 root]# rpm -q xvm-cmds
xvm-cmds-2.3-sgi230c1
```

   If this package is not installed, install the following RPMs from the *SGI ProPack v2.3 for Linux - Proprietary Software* CD:

```
[root@linux-64 PP_CDROM]# rpm -Uvh xvm-cmds-2.3-sgi230c1.ia64.rpm  \
xvm-standalone-module-2.3-sgi230c1.ia64.rpm
Preparing...                ########################################### [100%]
   1:xvm-standalone-module  ########################################### [ 50%]
   2:xvm-cmds               ########################################### [100%]
```

   Note: The `xvm-standalone-module` RPM is installed at this stage to satisfy package dependencies. It will be replaced in a subsequent step.

4. Install the CXFS kernel libraries from the *CXFS 3.0 Altix Server/Client and XVM Plexing for SGI ProPack 2.3* CD:

```
[root@linux-64 CXFS_CDROM]# rpm -Uvh cxfs-modules-3.0-sgi230a12.ia64.rpm
Preparing...                ########################################### [100%]
   1:cxfs-modules           ########################################### [100%]
```

5. Install the CXFS application binaries, documentation, and support tools from the *CXFS 3.0 Altix Server/Client and XVM Plexing for SGI ProPack 2.3* CD:

```
[root@linux-64 CXFS_CDROM]# rpm -Uvh cxfs_client-3.0-sgi230a4.ia64.rpm \
cxfs_util-3.0-sgi230a4.ia64.rpm \
cxfs_docs-3.0-sgi230a4.ia64.rpm
Preparing...                ########################################### [100%]
   1:cxfs_util              ########################################### [ 33%]
   2:cxfs_client            ########################################### [ 66%]
   3:cxfs_docs              ########################################### [100%]
```

6. Reboot the system.

Caution: If XVM standalone was in use prior to CXFS installation, you **must** reboot the system before starting CXFS services, or else the system will panic.

Verifying the Linux 64-bit Client-Only Installation

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

For example, the following output indicates that the CXFS packages are installed properly:

```
[root@linux-ia64 root]# rpm -q cxfs-modules cxfs_client cxfs_util
cxfs-modules-3.0-sgi230a12
cxfs_client-3.0-sgi230a4
cxfs_docs-3.0-sgi230a4
cxfs_util-3.0-sgi230a4
```

To verify the license, use the following command:

```
[root@linux root]# /usr/cluster/bin/cxfslicense -d
CXFS license granted.
```

Linux 64-bit Modifications Required 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 `xinted`.
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.

Chapter 8

Postinstallation Steps

This chapter discusses the following:

After completing these step discussed in this chapter, see Chapter 9, "Initial Configuration of the Cluster", page 103. For details about specific configuration tasks, see Chapter 10, "Reference to GUI Tasks for CXFS", page 129, and Chapter 11, "Reference to `cmgr` Tasks for CXFS", page 195. For information about installing CXFS and Trusted IRIX, see Chapter 14, "Trusted IRIX and CXFS", page 305. For information about upgrades, see "Rolling Upgrades", page 96.

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 a CXFS mount point is included in the `exports` file, the empty mount point is exported via the CXFS post/pre-mount scripts unless the filesystem is re-exported after the CXFS mount. For more information, see "NFS Export Scripts", page 264.

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`. Cluster services for CXFS do 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 64-bit: `/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:

```
# /etc/init.d/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 64-bit: `/etc/cluster/config/fs2d.options`

Table 8-1 shows the options can that can be set in the `fs2d.options` file.

Table 8-1 `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:

```
# /etc/init.d/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 64-bit: `/var/log/messages`
- Tracing information for RPCs, updates, and transactions are sent to `/var/cluster/ha/log/fs2d_ops1`.

  When the size 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 64-bit client-only nodes. For client-only nodes running other operating systems, see the *SGI InfiniteStorage CXFS MultiOS Client-Only Guide*.

`/etc/config/cxfs_client.options` on Client-only Nodes

You can modify the CXFS client service by placing options in the `/etc/config/cxfs_client.options` file. 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.

To see if `cxfs_client` is using the options in `cxfs_client.options`, enter the following:

```
# ps -ef | grep cxfs_client
```

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.

Configuring IRIX Interfaces

To configure IRIX network interfaces, do the following:

1. Ensure that name services are available. Using local name resolution is required. Even if you are using DNS or NIS, you must add every IP address and hostname

for the nodes to `/etc/hosts` on IRIX nodes and as defined in the *CXFS MultiOS for CXFS Client-Only Nodes: Installation and Configuration Guide*. 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`(4), `named`(1M), `dns`(7P), and `nis`(7P) man pages; *IRIX Admin: Networking and Mail*; and *NIS Administrator's Guide*.

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. 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)

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 serial hardware 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 *SGI InfiniteStorage CXFS MultiOS Client-Only Guide*.

1. Determine which port is used for the serial hardware 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
   ```

   Note: If you configure a cluster with the reset daemon running on an IRISconsole system, do not configure the reset port into the IRISconsole; it may conflict with the reset daemon that the CXFS system is running. (CXFS does **not** support the Silicon Graphics O2 workstation as a CXFS node and therefore it cannot be a CXFS serial hardware reset server.)

Rebooting the System

Execute the following command on each node to reboot it:

```
# reboot
```

The shutdown process then runs `autoconfig` to generate the kernel with your changes.

Testing the System

This section discusses the following:

- "Private Network Interface"
- "Serial Hardware Reset Connection for CXFS Administration Nodes", page 93

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.

Serial Hardware Reset Connection for CXFS Administration Nodes

To test the serial hardware 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 Startup/Shutdown

On administration nodes, the `/etc/init.d/cluster` script will be invoked automatically during normal system startup and shutdown procedures; on client-only nodes, the script is `/etc/init.d/cxfs_cluster`. 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:

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

- On a client-only node:

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

To stop CXFS processes manually, enter the following command:

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

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

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

The output in the following example shows that the service is stopped:

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

Modifying the CXFS Client Service

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

- IRIX: `/usr/cluster/bin/cxfs_client.options`
- Linux 64-bit: `/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
```

Rolling Upgrades

Beginning with IRIX 6.5.18f, SGI supports a policy for CXFS that permits a rolling annual upgrade. This policy allows you to upgrade a subset of the nodes in your cluster from IRIX 6.5.*n* to *n*+1 or *n*+4.

This policy lets you to keep your cluster running and filesystems available during the upgrade process.

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. (In a later release, the node will be able to run applications that use other filesystems; however, this feature does not apply to this release.) After the upgrade process is complete, all nodes should be running the same release.

Each CXFS MultiOS Client release is paired with a given IRIX or SGI ProPack release; the MultiOS Client release will also support the same *n*+1, *n*+4 release set during an upgrade. For example, for IRIX, the MultiOS 2.3 release supports IRIX 6.5.18, 6.5.19, and 6.5.22. It is recommended that you upgrade all MultiOS Client nodes at least annually. For more information, see the product release notes and the *SGI InfiniteStorage CXFS MultiOS Client-Only Guide*.

Note: In production mode, CXFS supports a cluster running a single IRIX release and a single CXFS MultiOS Client, or a single SGI ProPack release and a single CXFS for SGI ProPack release. If you are running multiple IRIX releases and run into problems, you may have to bring all administration nodes to a single operating system release before the problem can be addressed.

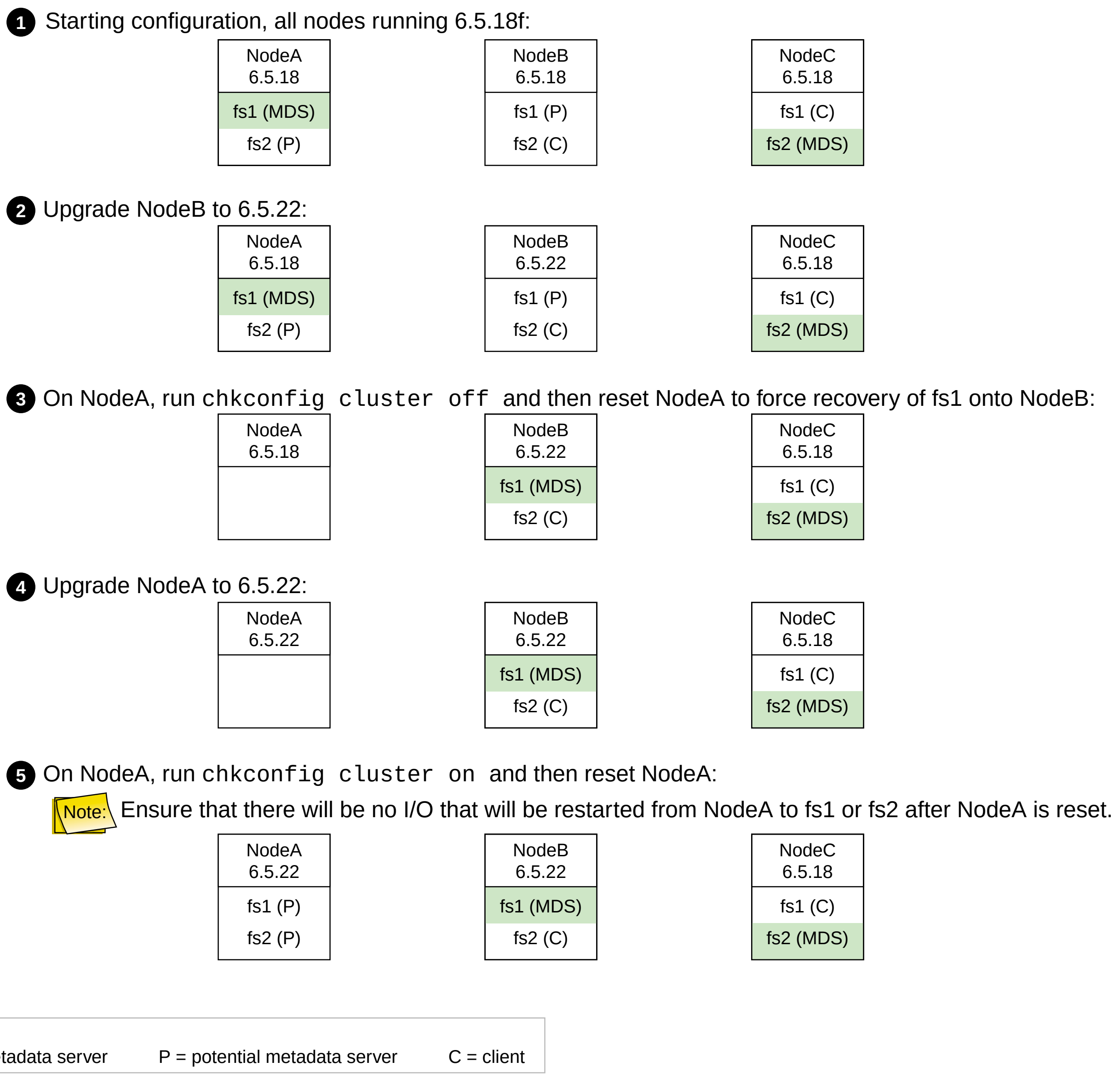
For Sites Running IRIX 6.5.16 and 6.5.17

Sites running IRIX 6.5.16 in a supported manner (with a single metadata server for a given filesystem and no potential metadata servers) may override the current restriction to adjacent releases and upgrade directly to 6.5.18.

For sites running 6.5.16 or 6.5.17 in an unsupported manner (that is, attempting to make use of the unsupported potential metadata server feature), the upgrade procedure documented in "Example Procedure: Upgrading from IRIX 6.5.18f to IRIX 6.5.22" **might** work; however, this cannot be guaranteed because recovery is not supported in the 6.5.16 and 6.5.17 releases due to its unreliability.

Example Procedure: Upgrading from IRIX 6.5.18f to IRIX 6.5.22

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 6.5.18f.



1 Starting configuration, all nodes running 6.5.18f:

2 Upgrade NodeB to 6.5.22:

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

4 Upgrade NodeA to 6.5.22:

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 8-1 Example Rolling Upgrade Procedure (steps 1-5)

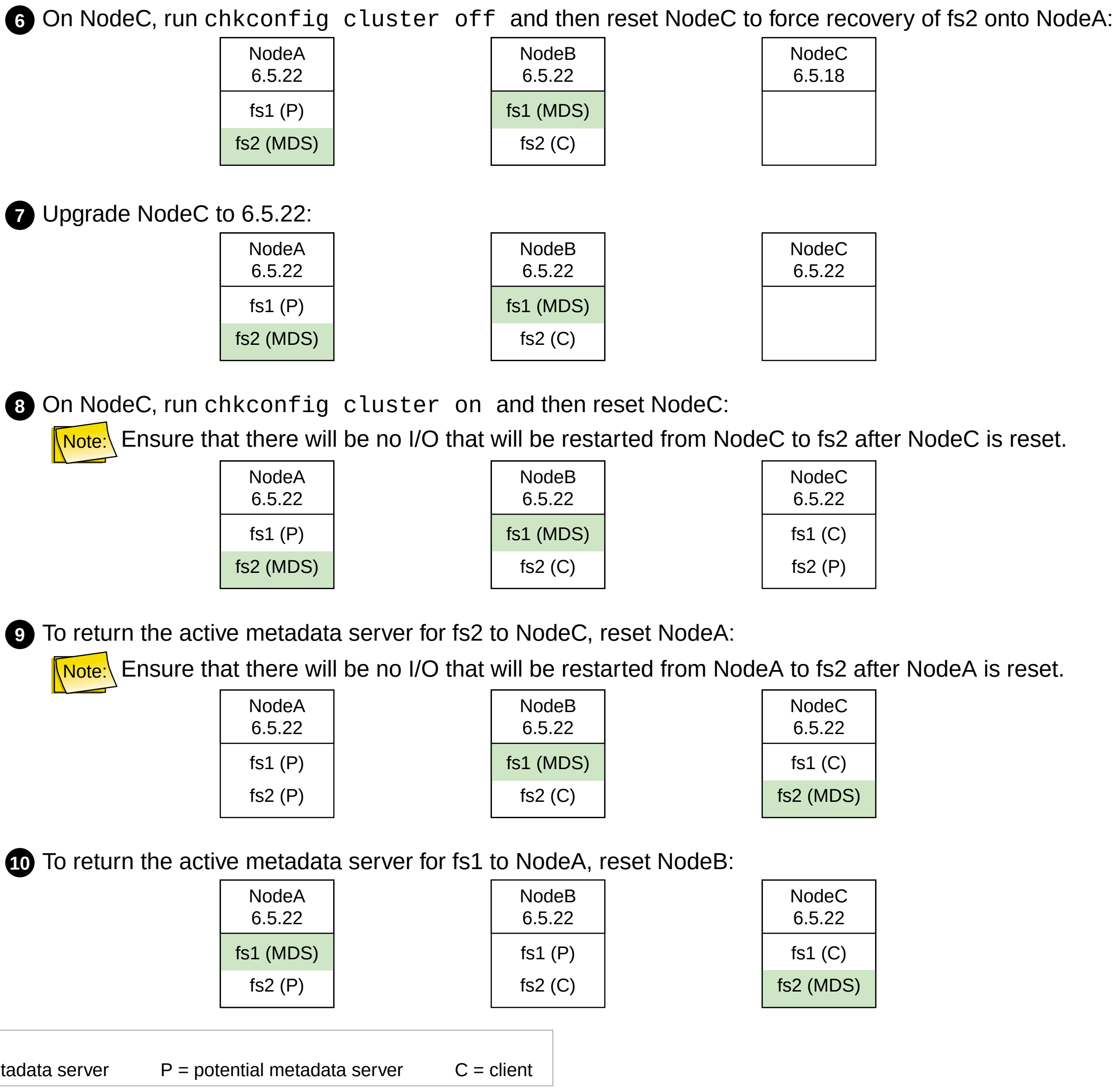

6 On NodeC, run chkconfig cluster off and then reset NodeC to force recovery of fs2 onto NodeA:
NodeA 6.5.22
fs1 (P)
fs2 (MDS)
NodeB 6.5.22
fs1 (MDS)
fs2 (C)
NodeC 6.5.18
7 Upgrade NodeC to 6.5.22:
NodeA 6.5.22
fs1 (P)
fs2 (MDS)
NodeB 6.5.22
fs1 (MDS)
fs2 (C)
NodeC 6.5.22
8 On NodeC, run chkconfig cluster 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.
NodeA 6.5.22
fs1 (P)
fs2 (MDS)
NodeB 6.5.22
fs1 (MDS)
fs2 (C)
NodeC 6.5.22
fs1 (C)
fs2 (P)
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.
NodeA 6.5.22
fs1 (P)
fs2 (P)
NodeB 6.5.22
fs1 (MDS)
fs2 (C)
NodeC 6.5.22
fs1 (C)
fs2 (MDS)
10 To return the active metadata server for fs1 to NodeA, reset NodeB:
NodeA 6.5.22
fs1 (MDS)
fs2 (P)
NodeB 6.5.22
fs1 (P)
fs2 (C)
NodeC 6.5.22
fs1 (C)
fs2 (MDS)
Key:
MDS = metadata server
P = potential metadata server
C = client

Figure 8-2 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"
```

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 may also wish to use the worksheet provided in Appendix D, "Initial Configuration Checklist", page 421. If you are converting from an existing FailSafe cluster, see "Set Up an Existing FailSafe Cluster for CXFS with the GUI", page 150.

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", page 110. You should also read through the entire book, including Chapter 16, "Troubleshooting", page 323, 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 6, "IRIX CXFS Installation", page 61. For information about other operating systems, see the *SGI InfiniteStorage CXFS MultiOS Client-Only Guide*.

Initial Configuration Requirements and Recommendations

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.

SGI recommends the following:

- If there are any network issues on the private network, fix them before trying to use CXFS.
- Use a network switch rather than a hub for performance and control.
- A production cluster should be configured with an odd number of server-capable nodes and a maximum of 48 nodes total, 16 of which can be administration nodes.
- 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", page 125.

- Create a new cluster using server-capable nodes that have the same version of the OS release installed. When the cluster is functioning, you can later upgrade selected nodes to an adjacent release.
- If you want to run CXFS and Trusted IRIX, all server-capable nodes must run Trusted IRIX. Client-only nodes can run IRIX (no nodes can run Linux 64-bit or other multiOS platforms). You should configure your system such that all nodes in the cluster have the same user IDs, access control lists (ACLs), and capabilities.
- Avoid unnecessary metadata traffic by avoiding the use of the `find` command. 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 doing a `find`. Edit the nodes' `crontab` file to only execute `find` on one metadata server of the cluster.
- 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 an odd number of server-capable nodes.
- Use an odd number of CXFS administration nodes.
- Shut down cluster services before maintenance. Disabled nodes are not used in CXFS kernel membership calculations, so this action may prevent a loss of quorum.
- Avoid recovery in the current release.

Note: In this release, relocation is disabled by default and recovery is supported only when using standby 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.

Do the following before shutting down a node:

1. Use the CXFS GUI or the `cmgr` command to unmount the filesystem from all hosts.
2. Shut down cluster services.

- 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 64-bit 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", page 187, and "Define a CXFS Filesystem with `cmgr`", page 235.

- Do not use any filesystem defragmenter software. You can use the IRIX `fsr` command or the Linux 64-bit `xfs_fsr` command **only** on a metadata server for the filesystem it acts upon.
- Do not include the `find` command in a `crontab` file on any node in the cluster. CXFS filesystems act like local filesystems and therefore the search will be done on each node. Using `find` will slow the system and temporarily consume large quantities of memory on the metadata server.
- 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.

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 cluster 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 subnetwork.

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.

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"
- "Start the Cluster Daemons", page 108
- "Verify that the Cluster Daemons are Running", page 108
- "Determine the Hostname of the CXFS Administration Node", page 109
- "Verify that the `chkconfig` Flags are On", page 109

During the course of configuration, you will see various information-only messages in the log files. See "Normal Messages", page 353.

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 64-bit: `/etc/flexlm/license.dat`

Note: The `license.dat` file cannot simply be copied between nodes because it is unique to each node.

For Linux 64-bit, you also need a license for XVM.

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

Start the Cluster Daemons

Enter the following on CXFS administration node to start the cluster daemons:

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

Verify that the Cluster Daemons are Running

When you **first install** the software, the following daemons should be running:

- `fs2d`
- `cmond`
- `cad`
- `crsd`

After you start CXFS services, the `clconfd` daemon is also started.

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:

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

For more information, see "Stopping and Restarting Cluster Infrastructure Daemons", page 377, and "Daemons", page 383.

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 6, "IRIX CXFS Installation", page 61.

Verify that the `chkconfig` Flags are On

Ensure that the appropriate `chkconfig` flags are `on`.

IRIX `chkconfig` Verification

For an IRIX node, ensure that `chkconfig` displays the following

```
# chkconfig
        Flag                 State
        ====                 =====
        cluster              on
        cxfs_cluster         on
```

If they are not, set them 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
irix# init 2
```

Linux 64-bit `chkconfig` Verification

For a Linux 64-bit node, use the following commands to verify the `chkconfig` flags:

```
[root@linux64 root]# chkconfig  --list cluster
cluster         0:off 1:off 2:on 3:on 4:on 5:on 6:off

[root@linux64 root]# chkconfig  --list cxfs
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@linux64 root]# chkconfig cxfs on
[root@linux64 root]# chkconfig cluster on
```

Configuring with the GUI

To initially configure the cluster with GUI, do the following:

- "Preliminary Cluster Configuration Steps", page 107
- "Start the GUI", page 111
- "Set Up a New Cluster with the GUI", page 112
- "Set Up a New CXFS Filesystem with the GUI", page 114

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:

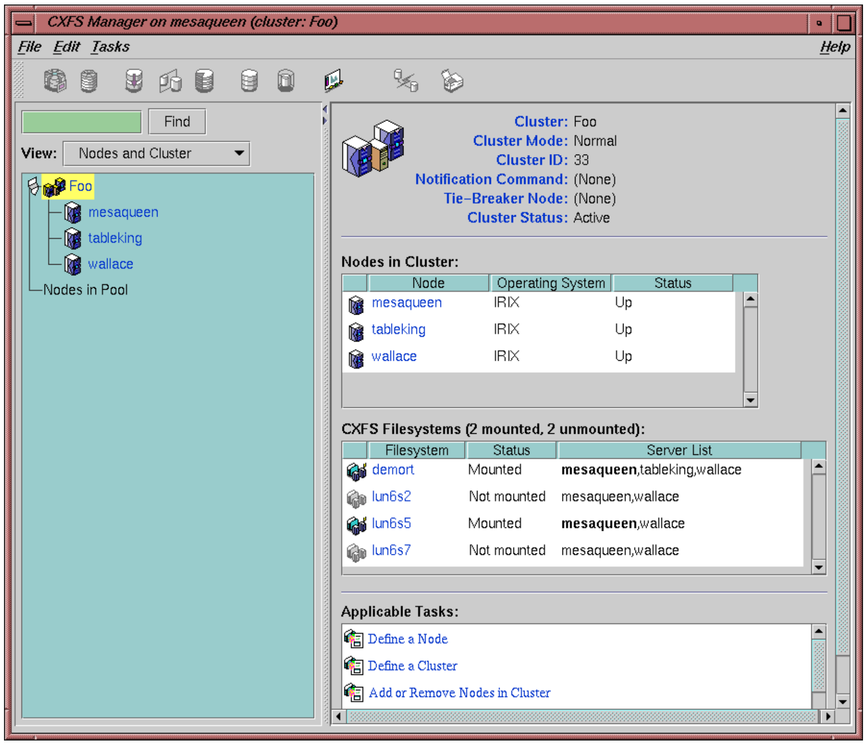
- IRIX: `sysadm_cxfs.sw.web`
- Linux 64-bit: `sysadm_cxfs-web`

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", page 130.

Supply the name of the CXFS administration node you wish to connect to and the `root` password.

Figure 9-1 shows an example of the CXFS Manager window.


CXFS Manager on mesaqueen (cluster: Foo)

File Edit Tasks Help

Find

View: Nodes and Cluster

Foo
- mesaqueen
- tableking
- wallace

Nodes in Pool

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 9-1 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", page 152.

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. 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 the GUI", page 152.

   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

3. Click **Define a Cluster** to create the cluster definition. See "Define a Cluster with the GUI", page 168. 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", page 162.

   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", page 167. (This test requires the proper configuration; see "IRIX Modifications Required for CXFS Connectivity Diagnostics", page 72, "Linux 64-bit Modifications Required for CXFS Connectivity Diagnostics", page 80.)

6. If you are using I/O fencing, define the Brocade Fibre Channel switch in the cluster. I/O fencing is required for nodes without system controllers; see "Requirements", page 33.

7. Click **Start CXFS Services**. See "Start CXFS Services with the GUI", page 171.

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", page 171.
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 64-bit `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", page 183.

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", page 186.

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", page 110.

For details about `cmgr` commands, see the man page and Chapter 11, "Reference to `cmgr` Tasks for CXFS", page 195.

To initially configure the cluster with the `cmgr` command, do the following:

1. Follow the directions in "Preliminary Cluster Configuration Steps", page 107.
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`", page 202.

   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
Node Function <server_admin|client_admin|client_only> ? server_admin
```

```
Operating System  <IRIX|Linux32|Linux64|AIX|HPUX|Solaris|Windows> ? irix
Node ID[optional]?
Partition ID[optional] ? (0)
Do you wish to define failure hierarchy[y/n]:n
Reset type <powerCycle>  ? (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 ? cxfs8
Sysctrl Device ? /dev/ttyd2
Sysctrl Owner Type <tty> ? (tty)
NIC 1 - IP Address ? cxfs6
Number of CXFS Interfaces ? (0)

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|Windows> ? irix
Node ID[optional] ?
Partition ID[optional] ? (0)
Do you wish to define failure hierarchy[y/n]:n
Reset type <powerCycle>  ? (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> ? (tty)
Number of Network Interfaces ? (1)
NIC 1 - IP Address ? cxfs7
Number of CXFS Interfaces ? (0)
```

```
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|Windows> ? irix
Node ID[optional] ?
Partition ID[optional] ? (0)
Do you wish to define failure hierarchy[y/n]:n
Reset type <powerCycle>  ? (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> ? (tty)
Number of Network Interfaces ? (1)
NIC 1 - IP Address ? cxfs8
Number of CXFS Interfaces ? (0)

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`", page 222.

   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
```

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`", page 248.

For more information, see "Define a Cluster with `cmgr`", page 222.

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
```

For an example of this step using a script, see "Script Example", page 252.

4. Start CXFS services for 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 the system 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 the cluster is up, 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
Membership since Wed May 16 14:42:48 2001
Node         NodeId     Status    Age   Incarnation     CellId
cxfs7           12812       UP      0             0          1
cxfs6           13203       UP      0             0          0
cxfs8           14033       UP      0             0          2
0 CXFS FileSystems
```

For more information, see "Display a Cluster with `cmgr`", page 228.

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*.

8. Mount the filesystems by using the `define cxfs_filesystem` subcommand to `cmgr`. See "CXFS Filesystem Tasks with `cmgr`", page 235.

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. 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", page 114.

   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 Shutdown|Reset-Shutdown|etc
        set reset_type to powerCycle
        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
```

For more information about using scripts and the `cmgr` command, see Chapter 11, "Reference to `cmgr` Tasks for CXFS", page 195

Chapter 10

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:

- "GUI Overview"
- "Guided Configuration Tasks", page 149
- "Node Tasks with the GUI", page 151
- "Cluster Tasks with the GUI", page 168
- "Cluster Services Tasks with the GUI", page 171
- "Switches and I/O Fencing Tasks with the GUI", page 176
- "Filesystem Tasks with the GUI", page 179
- "Privileges Tasks with the GUI", page 190

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 6, "IRIX CXFS Installation", page 61. For information about licensing on nodes running operating systems other than IRIX or Linux 64-bit, see the *SGI InfiniteStorage CXFS MultiOS Client-Only Guide*.

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:

- "Starting the GUI"
- "GUI Windows", page 133

- "GUI Features", page 136
- "Key to Icons and States", page 145

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", page 131.

Starting the GUI on Linux 64-bit

To start the GUI on a Linux 64-bit system where the CXFS GUI-client software (`sysadm_cxfs-client`) is installed, do the following:

1. Obtain and install the Java J2SE 1.4.2 SDK 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 64-bit: `sysadm_cxfs-web` and `sysadm_xvm-web`
- Install the Java2 v1.4.1 or v1.3.2 plug-in on your PC.
- 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 10-1 describes the platforms where the GUI may be started, connected to, and displayed.

Table 10-1 GUI Platforms

GUI Mode	Where You Start the GUI	Where You Connect the GUI	Where the GUI Displays
`cxfsmgr` or 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 (note, CXFS does not support the Silicon Graphics O2 workstation as a CXFS node and therefore it cannot be a CXFS serial hardware reset server) A Linux 64-bit SGI Altix 3000 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
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", page 190. (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
- **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, CXFS Manager 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

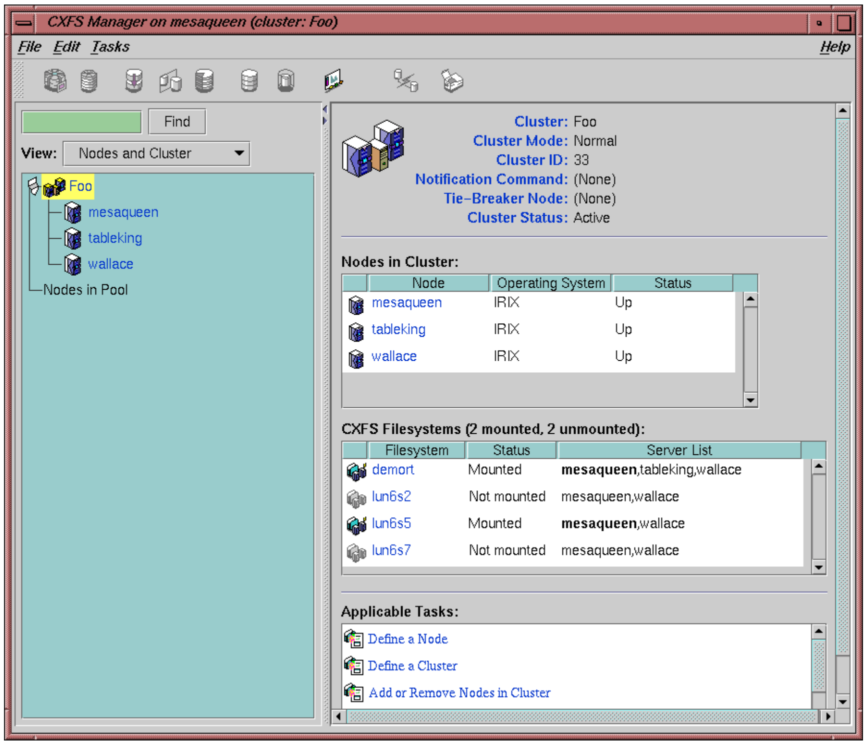
Making Changes Safely

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 10-1, page 134.) 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 10-1 shows the **CXFS Manager** window. Figure 10-2 shows 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", page 310.



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

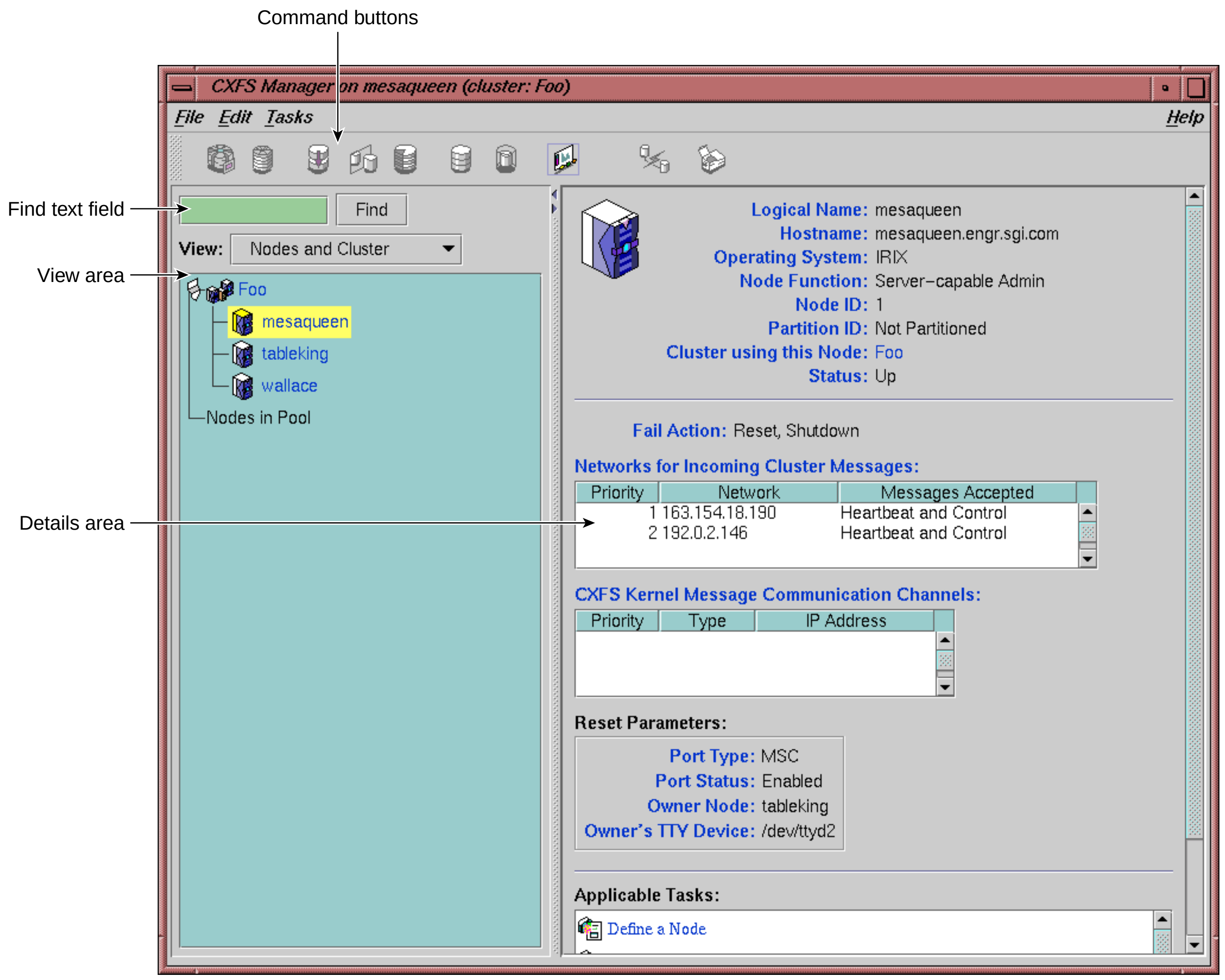
CXFS Filesystems (2 mounted, 2 unmounted):

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 10-1 CXFS Manager GUI Window



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

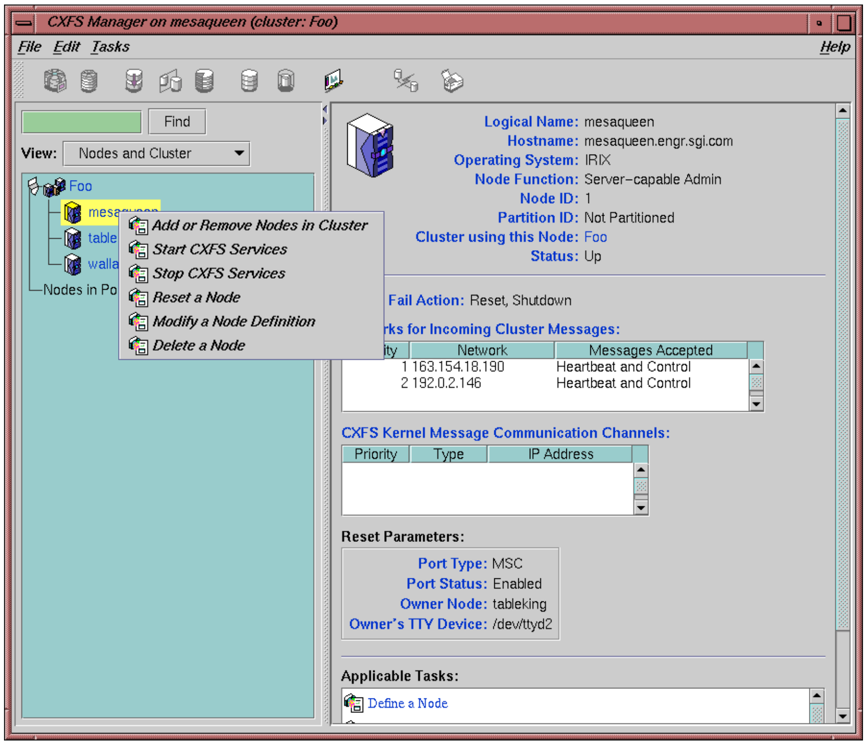
Reset Parameters:

Port Type: MSC
Port Status: Enabled
Owner Node: tableking
Owner's TTY Device: /dev/ttyd2

Applicable Tasks:

Figure 10-2 CXFS Manager GUI Showing Details for a Node

Figure 10-3 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.



View: Nodes and Cluster

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

Applicable Tasks:

Figure 10-3 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"
- "File Menu"
- "Edit Menu", page 138
- "Tasks Menu", page 138
- "Help Menu", page 139
- "Shortcuts Using Command Buttons", page 139
- "View Menu", page 141
- "Performing Tasks", page 142
- "Using Drag-and-Drop for XVM Configuration", page 142
- "Analyzing I/O Performance with Performance Co-Pilot on an IRIX Node", page 143
- "Structuring Volume Topologies", page 143
- "Configuring Disks", page 144
- "Getting More Information", page 145
- "Important GUI and `xvm` Command Differences", page 145

GUI Window Layout

By default, the window is divided into two sections: the *view area* and the *details area*. 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 64-bit: `/var/log/messages`

- System administration log file:
  - IRIX: `/var/sysadm/salog`
  - Linux 64-bit: `/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 Brocade Fibre Channel 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

  Note: Relocation is not supported in this release.

- **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 10-2 summarizes the shortcuts available; for details about these tasks, see the *XVM Volume Manager Administrator's Guide*.

Table 10-2 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

Button	Task
	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
	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", page 141.

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", page 139. 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 see term definitions, instructions on what to input, or item configuration details, or to launch tasks.

In general, clicking on blue text will 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, it is recommended 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 10-3 Key to Icons

Icon	Entity
	IRIX node (server-capable administration, client administration, or client-only)k
	Linux 64-bit (SGI Altix) node (server-capable administration, client administration, or client-only)
	AIX, Linux 32-bit, Solaris, or Windows node (client-only)
	Cluster
	Expanded tree in view area
	Collapsed tree in view area
	Switch
	XVM disk

Icon	Entity
	Unlabeled disk
	Foreign disk
	Slice
	Volume
	Subvolume
	Concat
	Mirror
	Stripe
	Slot

Icon	Entity
	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 10-4 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", page 150
- "Make Changes to Existing Cluster", page 151
- "Fix or Upgrade Cluster Nodes", page 151

Also see "Set Up a New Cluster with the GUI", page 112, "Set Up a New CXFS Filesystem with the GUI", page 114, and "Check Cluster Status with the GUI", page 310. 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", page 170.

2. Stop high availability (HA) services on the nodes to be converted using the FailSafe GUI. See the *SGI InfiniteStorage FailSafe Administrator's Guide*.

3. Add the second heartbeat and control network to the node definitions using the CXFS GUI. See "Modify a Node Definition with the GUI", page 163.

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", page 166.

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 cluster 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", page 169
- "Set Up a New CXFS Filesystem with the GUI", page 114
- "Modify a CXFS Filesystem with the GUI", page 185
- "Define a Node with the GUI", page 152
- "Test Node Connectivity with the GUI", page 167
- "Add or Remove Nodes in the Cluster with the GUI", page 162

Fix or Upgrade Cluster Nodes

This task leads you through the steps required to remove a node from a cluster. It covers the following steps:

- "Stop CXFS Services (Normal CXFS Shutdown) with the GUI", page 172.
- Perform the necessary maintenance on the node. Only if required, see "Reset a Node with the GUI ", page 162.
- "Start CXFS Services with the GUI", page 171.
- Monitor the state of the cluster components in the view area. See "Check Cluster Status with the GUI", page 310.

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", page 112.

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 NT, Windows 2000, or Windows XP.

An IRIX node or a Linux 64-bit 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, HP-UX, Linux 32-bit, Solaris, and Windows nodes are always CXFS client-only nodes.

IRIX nodes and Linux 64-bit nodes with system controllers may be reset; if you select a fail action that includes reset, you will be given an opportunity to provide serial hardware reset information on a second page.

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 64-bit 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 64-bit 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 IRIX, Linux 32-bit, Linux 64-bit, AIX, Solaris, or Windows. (Nodes other than IRIX and Linux 64-bit are **required** to be client-only nodes.)

5. **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", page 400.)

6. **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 64-bit `proc` command to determine the partition ID value (see below).

   Click **Next** to move to the next screen.

7. **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. Recovery begins without waiting for reset acknowledgment. This action is available only for administration nodes with system controllers; see "Requirements", page 33.
   - **Reset:** Performs a system reset via a serial line connected to the system controller. This action is available only for administration nodes with system controllers.
   - **Shutdown:** Stops CXFS kernel-based services on the node in response to a loss of CXFS kernel membership. The surviving cluster delays the beginning of recovery to allow the node time to complete the shutdown. This action is required for all nodes.

   On nodes without system controllers, your only choice for data integrity protection is I/O fencing.

   Note: A Brocade Fibre Channel switch sold and supported by SGI is mandatory to support I/O fencing; **therefore, multiOS CXFS clusters and clusters including Silicon Graphics Fuel, Octane, and Octane2 nodes require a Brocade switch**.

On other IRIX or Linux 64-bit nodes, 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 serial hardware reset for I/O protection when CXFS is a primary activity and you want to get it back online fast; for example, a CXFS file server.

The default fail action hierarchy for IRIX and Linux 64-bit 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", page 33.

Click **Next** to move to the next screen.

8. **Networks for Incoming Cluster Messages**: Do the following:

   - **Network**: Enter the IP address or hostname of the private network. (The hostname must be resolved in the `/etc/hosts` file.) SGI strongly recommends that the priorities of the networks be the same for each node in the cluster. For information about why a private network is required, see "Private Network", page 18.

     FailSafe requires at least two networks.

   - **Messages to Accept**: Select **Heartbeat and Control**.

     You can use the **None** setting if you want to temporarily define a network but do not want it to accept messages. For more information, see "Cluster Environment", page 8.

   - Click **Add** to add the network to the list.

     If you later want to modify the network, click the network in the list to select it, then click **Modify**.

     To delete a network from the list, click the network in the list to select it, then click **Delete**.

9. **CXFS Kernel Message Communication Channels**:This section defines the network interfaces that will be used for CXFS kernel-to-kernel messaging.

   Note: Deferred implementation.

If you supply multiple interfaces, the network will fail over from a higher-priority network to a lower-priority network; priority 1 is the highest priority.

If you do not enter an IP address (or if all of the IP addresses you do enter are unavailable), CXFS will use the priority 1 cluster administration network (defined in the previous step) for both cluster control and CXFS kernel messages.

Do the following:

- **IP Address**: Enter the IP address or hostname of the private network. (The hostname must be resolved in the `/etc/hosts` file.) SGI recommends that the priorities of the networks be the same for each node in the cluster. For information about why a private network is required, see "Private Network", page 18.
- Click **Add** to add the network to the list.

  If you later want to modify the network, click the network in the list to select it, then click **Modify**.

  To delete a network from the list, click the network in the list to select it, then click **Delete**. Use the arrow keys to reorder the selected network.

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 64-bit, 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):

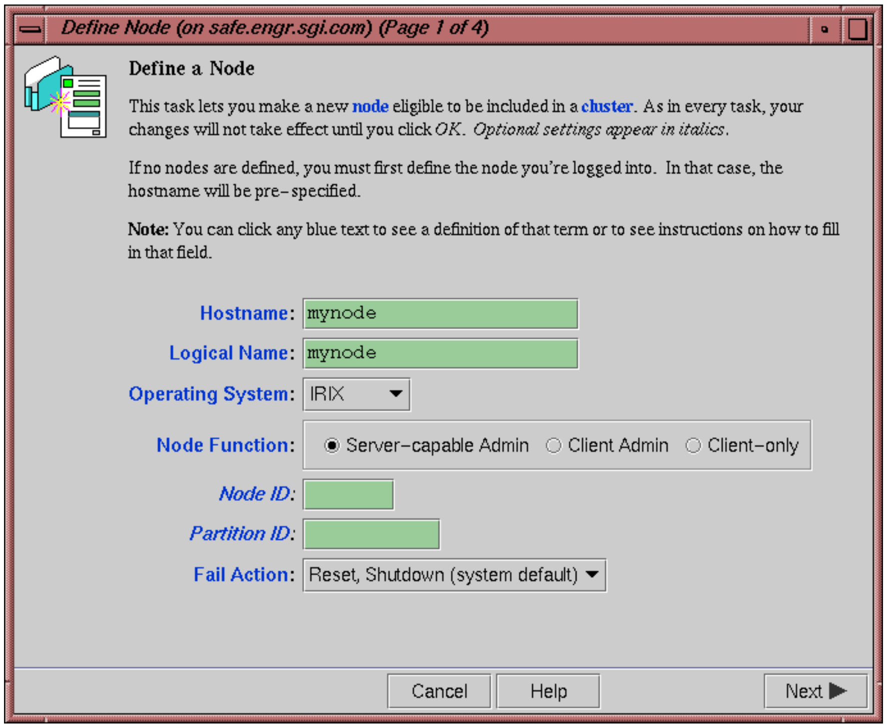
```
# mkpart -n
Partition id = 1
# mkpart -l
partition: 3 = brick: 003c10 003c13 003c16 003c21 ...
partition: 1 = brick: 001c10 001c13 001c16 001c21 ...
```

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 a Node

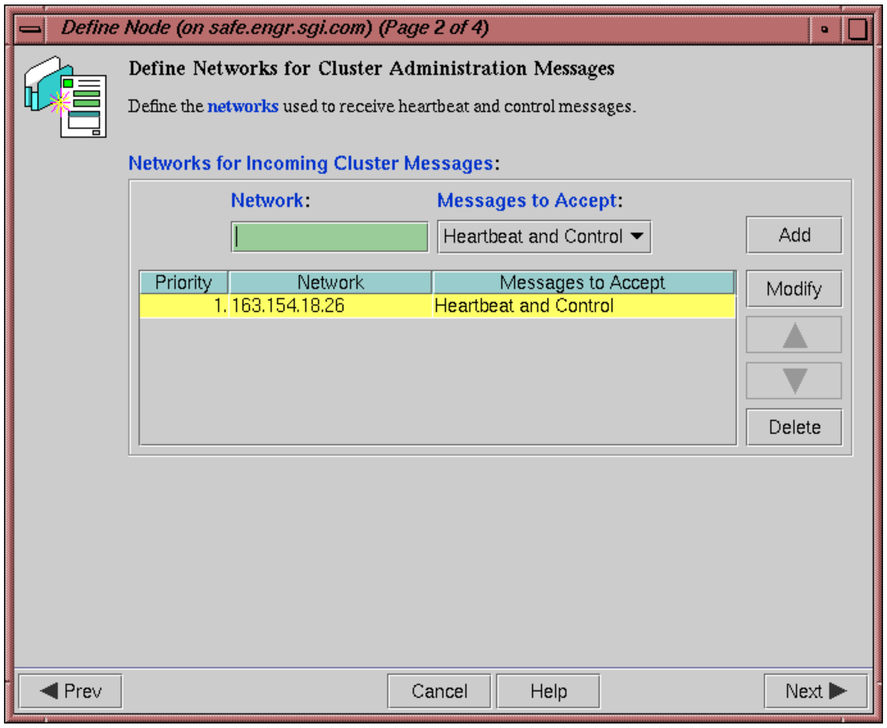
This task lets you make a new **node** eligible to be included in a **cluster**. As in every task, your changes will not take effect until you click *OK*. *Optional settings appear in italics.*

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
Node ID:	
Partition ID:	
Fail Action:	Reset, Shutdown (system default)

Figure 10-4 Example Node Definition


Define Node (on safe.engr.sgi.com) (Page 2 of 4)

Define Networks for Cluster Administration Messages

Define the networks used to receive heartbeat and control messages.

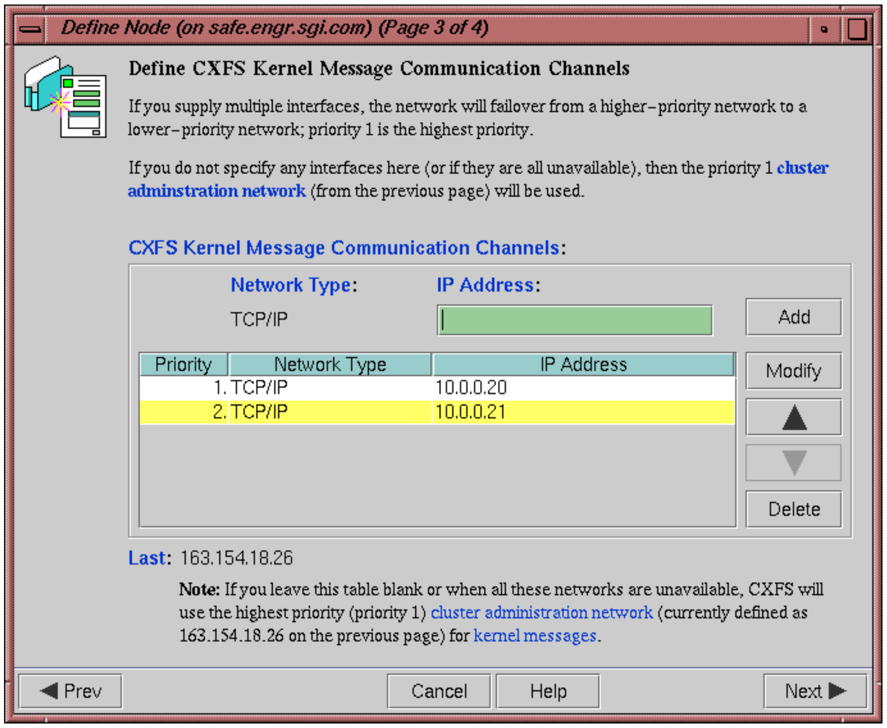
Networks for Incoming Cluster Messages:

Network:

Messages to Accept: Heartbeat and Control

Priority	Network	Messages to Accept
1.	163.154.18.26	Heartbeat and Control

Figure 10-5 Example Cluster Control Network Setting



Define CXFS Kernel Message Communication Channels

If you supply multiple interfaces, the network will failover from a higher–priority network to a lower–priority network; priority 1 is the highest priority.

If you do not specify any interfaces here (or if they are all unavailable), then the priority 1 cluster adminstration network (from the previous page) will be used.

CXFS Kernel Message Communication Channels:

Network Type: TCP/IP

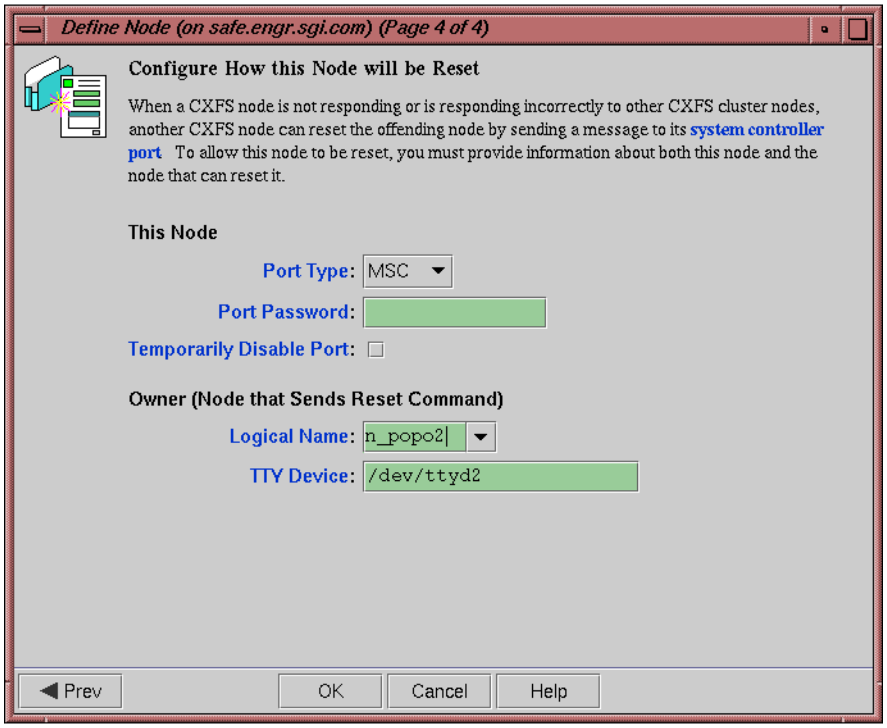
IP Address:

Priority	Network Type	IP Address
1.	TCP/IP	10.0.0.20
2.	TCP/IP	10.0.0.21

Last: 163.154.18.26

Note: If you leave this table blank or when all these networks are unavailable, CXFS will use the highest priority (priority 1) cluster administration network (currently defined as 163.154.18.26 on the previous page) for kernel messages.

Figure 10-6 Example CXFS Kernel Message Interface Settings (Deferred Implementation)



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.

This Node

Port Type: MSC

Port Password:

Temporarily Disable Port: ☐

Owner (Node that Sends Reset Command)

Logical Name: n_popo2

TTY Device: /dev/ttyd2

Figure 10-7 Example System Controller 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 64-bit 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. **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 64-bit `proc` command to determine the partition ID value; see below.
3. **Keep FailSafe Settings:** If a CXFS administration node has been defined for use with FailSafe, you can reconfigure the node so that it can be used with CXFS. Optionally, you can reconfigure the node so that it can no longer be used with FailSafe. Clear this option if you no longer wish to use this node with FailSafe.
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 serial hardware 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", page 33.
   - **Reset:** Performs a system reset via a serial line connected to the system controller. This action is available for nodes with system controllers.
   - **Shutdown:** Stops CXFS kernel-based services on the node in response to a loss of CXFS kernel membership. The surviving cluster delays the beginning of recovery to allow the node time to complete the shutdown. This action is required for all nodes.

   The default fail action hierarchy for IRIX or Linux 64-bit nodes is **Reset, Shutdown**. The default for other nodes is **Shutdown**.

Click **Next** to move to the next page.

5. **Networks for Incoming Cluster Messages**: SGI strongly recommends that the priorities of the networks must be the same for each node in the cluster.

   - **Network**: To add a network for incoming cluster messages, enter the IP address or hostname into the **Network** text field and click **Add**.
   - To modify a network that is already in the list, click the network in the list in order to select it. Then click **Modify**. This moves the network 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 network, click the network in the priority list in order to select it. Then click **Delete**.
   - To change the priority of a network, click the network 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 network but do not want it to accept messages. For more information, see "Cluster Environment", page 8.

   Click **Next** to move to the next page.

6. **CXFS Kernel Message Communication Channels**:This section defines the network interfaces that will be used for CXFS kernel-to-kernel messaging.

   Note: Deferred implementation.

   If you supply multiple interfaces, the network will fail over from a higher-priority network to a lower-priority network; priority 1 is the highest priority.

   If you do not enter an IP address (or if all of the IP addresses you do enter are unavailable), CXFS will use the priority 1 cluster administration network (defined in the previous step) for both cluster control and CXFS kernel messages.

   Do the following:

   - **IP Address**: Enter the IP address or hostname of the private network. (The hostname must be resolved in the `/etc/hosts` file.) SGI recommends that the priorities of the networks be the same for each node in the cluster. For information about why a private network is required, see "Private Network", page 18.

- Click **Add** to add the network to the list.

  If you later want to modify the network, click the network in the list to select it, then click **Modify**.

  To delete a network from the list, click the network in the list to select it, then click **Delete**. Use the arrow keys to reorder the selected network.

Click **Next** to move to the next page.

7. Change the values as needed.
8. 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 64-bit, 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 IRIS 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 *SGI InfiniteStorage FailSafe Administrator's Guide*.
2. Add the second **Heartbeat and Control** network to the node definition using the CXFS GUI. See "Modify a Node Definition with the GUI", page 163.
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", page 163. 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 *SGI InfiniteStorage FailSafe Administrator's Guide*.

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", page 169.

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", page 112.

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 that share a network 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 that share a network 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", page 168, for information about `Experimental` mode.
   - **Notify Administrator**: Select the desired notification. For more information, see "Define a Cluster with the GUI", page 168.
2. Click **OK**.

To modify the nodes that make up a cluster, see "Add or Remove Nodes in the Cluster with the GUI", page 162.

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 after Renaming Cluster", page 348.

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", page 162.

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", page 162.

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.

You can also use the `cluster_status` command; see Chapter 15, "Monitoring Status", page 307.

Cluster Services Tasks with the GUI

The following tasks let you start and stop CXFS cluster 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 (Normal CXFS Shutdown) 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:

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

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", page 400.

Caution: If the CXFS tiebreaker node in a cluster with 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.

To ensure data integrity, SGI highly recommends that you use serial hardware reset and/or I/O fencing for all CXFS systems, especially clusters with two server-capable 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 64-bit: `/var/log/messages`

ou 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", page 308.

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", page 277, "`cad.options` on CXFS Administration Nodes", page 84, and "`fs2d.options` on CXFS Administration Nodes", page 86.

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 64-bit `/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`", page 231.

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", page 271).

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", page 271.

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", page 175.
- When instructed to by an error message on the console or in system log file:
  - IRIX: `/var/adm/SYSLOG`
  - Linux 64-bit: `/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 64-bit `/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 Brocade Fibre Channel switches for the CXFS cluster and perform I/O fencing.

Note: Nodes without system controllers require I/O fencing to protect data integrity. A Brocade Fibre Channel switch sold and supported by SGI is mandatory to support I/O fencing; therefore, multiOS CXFS clusters and clusters containing Silicon Graphics Fuel, Octane, and Octane2 nodes require a Brocade switch.

Define a Switch with the GUI

This task lets you define a new Brocade Fibre Channel switch sold and supported by SGI to support I/O fencing in a cluster. Do the following:

1. Enter the following information:

   - **Switch Name:** Enter the hostname of the Brocade Fibre Channel 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 Brocade Fibre Channel switch. By default, this value is `admin`.
   - **Password:** Enter the password for the specified **Username** field.
   - **Mask:** Enter a hexadecimal string (representing a 64-bit port bitmap) that indicates the list of ports in the Brocade Fibre Channel switch that will not be fenced. Ports are numbered from zero. If bit `i` is nonzero, then the port that corresponds to `i` will always be excluded from any fencing operations. For example, a mask of `A4` indicates that the 3rd, 6th, and 8th ports (port numbers 2, 5, and 7) will not be affected by fencing.

     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 Fibre Channel switch definition. Do the following:

1. Enter the following information:

   - **Switch Name:** Select the hostname of the Brocade Fibre Channel switch to be modified.

- **Username:** Enter the user name to use when sending a `telnet` message to the Brocade Fibre Channel switch. By default, this value is `admin`.
- **Password:** Enter the password for the specified **Username** field.
- **Mask:** Enter a 64-bit bitmap that indicates the list of ports in the switch that will not be fenced. For more information, see "Define a Switch with the GUI", page 177.

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 Brocade Fibre Channel 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 Brocade Fibre Channel 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 Brocade 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", page 114.

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 64-bit `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: Before you can grow a filesystem, you must first increase the size of the logical volume on which the filesystem is mounted. You can launch the **Insert Mirrors or Concats above Volume Elements** task to add a concat, or you can use the drag-and-drop mechanism to attach a slice to an existing concat. You cannot add a slice to an existing volume element if this changes the way that the data is laid out in that volume element or in any ancestor of that volume element. For more information, 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 64-bit `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", page 110.

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", page 187.

- **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 64-bit).

  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: In this release, recovery is 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", page 19.

  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", page 179 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", page 188.

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", page 171.

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", page 187.

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

Note: Relocation is not supported in this release.

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", page 19). 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", page 171.
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.

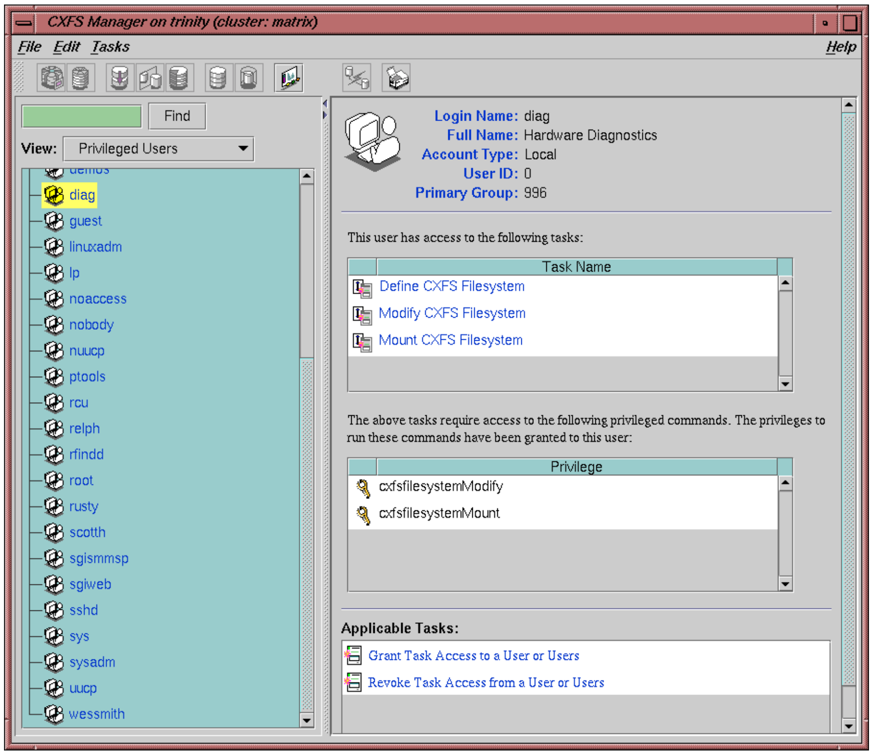
Granting Access to a Few Tasks

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 10-8 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 10-8 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 11

Reference to cmgr Tasks for CXFS

This chapter discusses the following:

- "cmgr Overview", page 196
- "Set Configuration Defaults with cmgr", page 201
- "Node Tasks with cmgr", page 202
- "Cluster Tasks with cmgr", page 222
- "Cluster Services Tasks with cmgr", page 228
- "CXFS Filesystem Tasks with cmgr", page 235
- "Switches and I/O Fencing Tasks with cmgr", page 248
- "Script Example", page 252
- "Creating a cmgr Script Automatically", page 255

For an overview of the tasks that must be performed to configure a cluster, see "Configuring with the cmgr Command", page 115.

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 cluster_status tool to view status in curses mode. See Chapter 15, "Monitoring Status", page 307.

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 4, "Obtaining CXFS and XVM FLEXlm Licenses", page 53, Chapter 6, "IRIX CXFS Installation", page 61, and Chapter 7, "Linux 64-bit CXFS Installation", page 73. For information about licensing and nodes running an operating system other than IRIX or Linux 64-bit, see the *SGI InfiniteStorage CXFS MultiOS Client-Only Guide*

---

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|Windows> ?
Node ID ?[optional]
Partition ID ?[optional] (0)
Do you wish to define failure hierarchy[y/n]:
Reset type <powerCycle> ? (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> ? (tty)
NIC 1 - IP Address ?
Number of CXFS Interfaces ? (0)
```

For details about this task, see "Define a Node with `cmgr`", page 202.

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", page 252.

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
    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
    set sysctrl_owner_type to tty_device
    set is_failsafe to true|false
    set is_cxfs to true|false
    set operating_system to irix|linux32|linux64|aix|hpux|solaris|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]
    add cxfs_interface interface
            set ic_type to IP|NUMAlink
            set ic_priority to 1|2|3|...
    remove cxfs_interface interface
```

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 64-bit 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 64-bit 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, Linux 32-bit, Linux 64-bit, AIX, Solaris, or Windows. (Nodes other than IRIX and Linux 64-bit are **required** to be client-only nodes.)

AIX , Solaris, 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@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.

- `reset_type` must be `powerCycle`
- `sysctrl_type` is the system controller type based on the node hardware, as show in Table 11-1.

Table 11-1 System Controller Types

`l1`	`l2`	`mmsc`	`msc`
Origin 300	Origin 3400	SGI 2400 rackmount	Origin 200
Origin 3200C	Origin 3800	SGI 2800 rackmount	Onyx2 deskside
Onyx 300	Origin 300 with NUMAlink module	Onyx2 rackmount	SGI 2100 deskside
Onyx 3200C	Onyx 3000 series		SGI 2200 deskside
	Altix 3000		

- `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 serial hardware 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", page 33.

- sysctrl_device is the port used. /dev/ttyd2 is the most commonly used port, except on Origin 300 and Origin 350 systems, where /dev/ttyd4 is commonly used.
- sysctrl_owner is the name of the node that sends the serial hardware 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 the 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 end.
- 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, hpux, solaris, or windows. (Use windows for Windows NT, Windows 2000, or Windows XP.)

  Note: For support details, see the *SGI InfiniteStorage CXFS MultiOS Client-Only Guide*.

  If you specify aix, hpux, solaris, 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. If the first priority network fails, the second will be used, and so on. SGI requires that this network be private; see "Private Network", page 18.

The priorities of the networks must be the same for each node in the cluster. For more information about using the hostname, see "Hostname Resolution and Network Configuration Rules", page 106.

- `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 stopped. Therefore, you should choose a setting other than the default for nodes without system controllers; see "Requirements", page 33. 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 serial hardware 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 Brocade Fibre Channel switch sold and supported by SGI is mandatory to support I/O fencing.

  - `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", page 33.
  - `reset` performs a reset via a serial line connected to the system controller. This action is available only for nodes with system controllers.
  - `shutdown` stops CXFS kernel-based services on the node in response to a loss of CXFS kernel membership . The surviving cluster delays the beginning of recovery to allow the node time to complete the shutdown. This action is available for all nodes.

  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 shutdown 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
  ```

- `cxfs_interface` is one or more interfaces to use for CXFS kernel-to-kernel messaging. NUMAlink addresses are not currently supported.

  If you supply multiple interfaces, the network will fail over from a higher-priority network to a lower-priority network; priority 1 is the highest priority.

  If you do not enter an interface (or if all of the interfaces you do enter are unavailable), CXFS will use the priority 1 NIC network for both cluster control and CXFS kernel messages.

  Note: Deferred implementation.

For more information, see "CXFS Kernel Membership, Quorum, and Tiebreaker", page 400, and "Define a Node with the GUI", page 152.

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", page 152. 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|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> ? (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
Sysctrl Owner Type <tty> ? (tty) tty_device
NIC 1 - IP Address ? IP_address_or_hostname (if DNS)
Number of CXFS Interfaces ? (0) number
```

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|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>  ? (powerCycle)
Do you wish to define system controller info[y/n]:n
NIC 1 - IP Address ? 111.11.11.111
Number of CXFS Interfaces ? (0)
```

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|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] ?
NIC 1 - IP Address ? 163.154.18.172
Number of CXFS Interfaces ? (0)
```

Following is an example of defining a node using multiple CXFS interfaces for CXFS kernel-to-kernel messaging:

Note: Deferred implementation.

```
cmgr> define node n_safe
Enter commands, you may enter "done" or "cancel" at any time to exit

Hostname[optional] ? safe.engr.sgi.com
Is this a FailSafe node <true|false> ? false
Is this a CXFS node <true|false> ? true
Operating System <IRIX|Linux32|Linux64|AIX|HPUX|Solaris|Windows> ? IRIX
Node Function <server_admin|client_admin|client_only> ? server_admin
Node ID[optional] ? 103
Partition ID[optional] ? (0)
Do you wish to define failure hierarchy[y/n]:y
Hierarchy option 0 <System|FenceReset|Fence|Reset|Shutdown>[optional] ? Reset
Hierarchy option 1 <System|FenceReset|Fence|Reset|Shutdown>[optional] ? Shutdown
Hierarchy option 2 <System|FenceReset|Fence|Reset|Shutdown>[optional] ?
Reset type <powerCycle> ? (powerCycle)
Do you wish to define system controller info[y/n]:y
Sysctrl Type <msc|mmsc|l2|l1> ? (msc)
Sysctrl Password[optional] ? ( )
Sysctrl Status <enabled|disabled> ? enabled
Sysctrl Owner ? n_safer
Sysctrl Device ? /dev/ttyd2
Sysctrl Owner Type <tty> ? (tty)
NIC 1 - IP Address ? safe
Number of CXFS Interfaces ? (0) 2
CXFS INTF 0 - Interface Address ? 10.0.0.10
CXFS INTF 0 - Interface Type ? IP
CXFS INTF 0 - Interface Priority ? 1
CXFS INTF 1 - Interface Address ? safe
CXFS INTF 1 - Interface Type ? IP
CXFS INTF 1 - Interface Priority ? 2

Successfully defined node n_safe
```

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
    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
    set sysctrl_owner_type to tty_device
    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]
    add cxfs_interface interface
            set ic_type to IP|NUMAlink
            set ic_priority to 1|2|3|...
    remove cxfs_interface interface
```

Note: 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`.

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`", page 202.

Caution: Do not change the node ID number after the node has been defined.

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> ? (powerCycle)
Do you wish to modify system controller info[y/n]:n
NIC 1 - IP Address ? (preston)
Number of CXFS Interfaces? (0)

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.

```
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> ? (powerCycle)
Do you wish to modify system controller info[y/n]:n
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> ? (powerCycle)
Do you wish to modify system controller info[y/n]:n
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
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 the reset device name and type and the system controller's type are not already defined in the database (for example, if you want to test reset before defining the node that will perform the reset), you could use the following commands to connect to the system controller or reset the node:

`admin ping dev_name` *tty* `of dev_type` *port* `with sysctrl_type` *msc|mmsc|l2|l1*

`admin reset dev_name` *tty* `of dev_type` *port* `with sysctrl_type` *msc|mmsc|l2|l1*

For more information about the command elements, see "Define a Node with `cmgr`", page 202.

This command does not go through the CXFS daemons.

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> ? (powerCycle)
Do you wish to modify system controller info[y/n]:n
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 from cluster cxfs6-8
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 Required for CXFS Connectivity Diagnostics", page 72, "Linux 64-bit Modifications Required for CXFS Connectivity Diagnostics", page 80.

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 "Serial Hardware Reset Connection for CXFS Administration Nodes", page 93.

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", page 168.

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
    ...
```

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 that share a network 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 that share a network 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.

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

Add nodes to or remove nodes 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
```

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

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
```

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|false
    set is_cxfs to true|false
    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
...
```

These commands are the same as the `define cluster` commands. For more information, see "Define a Cluster with `cmgr`", page 222, and "Define a Cluster with the GUI", page 168.

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 after Renaming Cluster", page 348.

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:

```
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
```

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
```

```
start cx_services for cluster clustername
```

For example:

```
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 temporarily (that is, allowing them to restart with a reboot), use the following command line in a shell window outside of `cmgr`:

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

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]
```

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 the CXFS tiebreaker node in a cluster with 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.

The serial hardware reset capability or I/O fencing with switches is **mandatory** to ensure data integrity for clusters with only two server-capable nodes and it is highly recommended for all server-capable nodes. Larger clusters should have an odd number of server-capable nodes, or must have serial hardware reset lines or use I/O fencing with switches if only two of the nodes are server-capable. (See "CXFS Recovery Issues in a Cluster with Only Two Server-Capable Nodes ", page 414.)

Solaris and Windows nodes require I/O fencing to protect data integrity. A Brocade switch is mandatory to support I/O fencing; therefore, multiOS CXFS clusters require a Brocade switch.

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", page 173.

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`", page 234.
- 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", page 271.

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`", page 235.
- The global status of the filesystem is enabled. See "Mount a CXFS Filesystem with `cmgr`", page 241.

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:

- In this release, relocation is not supported and is disabled by default. Recovery is 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", page 19.
- 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`", page 241.

- `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`", page 247.

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`", page 235.

Nodes must first be enabled by using the `define cxfs_filesystem` and `modify cxfs_filesystem` commands; see "Define a CXFS Filesystem with `cmgr`", page 235, and "Modify a CXFS Filesystem with `cmgr`", page 243.

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`", page 229.

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`", page 235.

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`

Note: Relocation is fully implemented, but the number of associated problems prevents support of this feature in CXFS. While data integrity is not compromised, cluster node panics or hangs are likely to occur. This feature will be fully supported when these issues are resolved. Relocation is disabled by default in this release.

If relocation is explicitly enabled in the kernel with the `cxfs_relocation_ok` systune (see "Relocation", page 19), 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`", page 243.

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", page 23.

Note: Solaris and Windows nodes require I/O fencing to protect data integrity. A Brocade switch is mandatory to support I/O fencing; therefore, multiOS CXFS clusters require a Brocade switch.

Define a Switch with cmgr

To define a new switch to support I/O fencing in a cluster, 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 (representing a 64-bit port bitmap) that indicates the list of ports in the switch that will not be fenced. Ports are numbered from zero. If bit `i` is nonzero, then the port that corresponds to `i` will always be excluded from any fencing operations. For example, a mask of `A4` indicates that the 3rd, 6th, and 8th ports (port numbers 2, 5, and 7) will not be affected by fencing.

  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 given Fibre Channel 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`", page 248.

For example, to change the mask for switch `ptg-brocade` from `A4` to `0` (which means that none of the ports on the switch will be excluded from 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 Brocade 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:

`/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
```

Chapter 12

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.)

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:

- "Granting Task Execution Privileges to Users", page 263
- "Transforming an Existing Node into a Client-Only Node", page 264
- "NFS Export Scripts", page 264
- "Using `telnet` and I/O Fencing", page 267
- "Using `fsr` and `xfs_fsr`", page 267
- "Using `cron` in a CXFS Cluster", page 267
- "Using Hierarchical Storage Management (HSM) Products", page 268
- "Discovering the Active Metadata Server for a Filesystem", page 268
- "Metadata Server Recovery", page 270
- "Shutdown of the Database and CXFS", page 271
- "Avoiding a CXFS Restart at Reboot", page 277
- "Log File Management", page 277

- "Volume Management", page 279
- "Disk Management", page 279
- "Filesystem Maintenance", page 280
- "Dump and Restore", page 283
- "Cluster Database Backup and Restore", page 284
- "`chkconfig` Settings", page 288
- "System Tunable Parameters", page 290

If you have upgraded directly from 6.5.12f or earlier, you must manually convert you filesystem definitions to the new format. See "IRIX: Converting Filesystem Definitions for Upgrades", page 100.

Configuring Real-Time Filesystems For IRIX Nodes

IRIX nodes support real-time CXFS filesystems on real-time XVM volumes. 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 64-bit.)

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", page 190.

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", page 185, or "Modify a CXFS Filesystem with `cmgr`", page 243.
2. Stop the CXFS services on the node. See "Stop CXFS Services (Normal CXFS Shutdown) with the GUI", page 172, or "Stop CXFS Services with `cmgr`", page 229.
3. Modify the cluster so that it no longer contains the node. See "Modify a Cluster Definition with the GUI", page 169, or "Modify a Cluster with `cmgr`", page 225.
4. Delete the node definition. See "Delete a Node with the GUI", page 167, or "Delete a Node with `cmgr`", page 218.
5. Reboot the node to ensure that all previous node configuration information is removed.
6. Install the node with the `cxfs_client` package. See"IRIX Client-only Software Installation", page 68.
7. Redefine the node and use a node function of client-only. See "Define a Node with the GUI", page 152, or "Define a Node with `cmgr`", page 202.
8. Modify the cluster so that it contains the node. See "Modify a Cluster Definition with the GUI", page 169, or "Modify a Cluster with `cmgr`", page 225.
9. Start the CXFS services on the node. See "Start CXFS Services with the GUI", page 171, or "Start CXFS Services with `cmgr`", page 229.

NFS Export Scripts

When you install CXFS, the following default scripts are placed in the appropriate directory:

- 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.

These scripts allow you to use NFS to export the CXFS filesystems listed in `/etc/exports` if they are successfully mounted. The scripts also ensure that LUN path failover works properly after fencing by executing the following:

```
/etc/init.d/failover stop
/etc/init.d/failover start
```

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 64-bit `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 64-bit 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", page 183, or "Modify a Cluster with `cmgr`", page 225.
- 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 `cluster_status` or `clconf_info` commands.

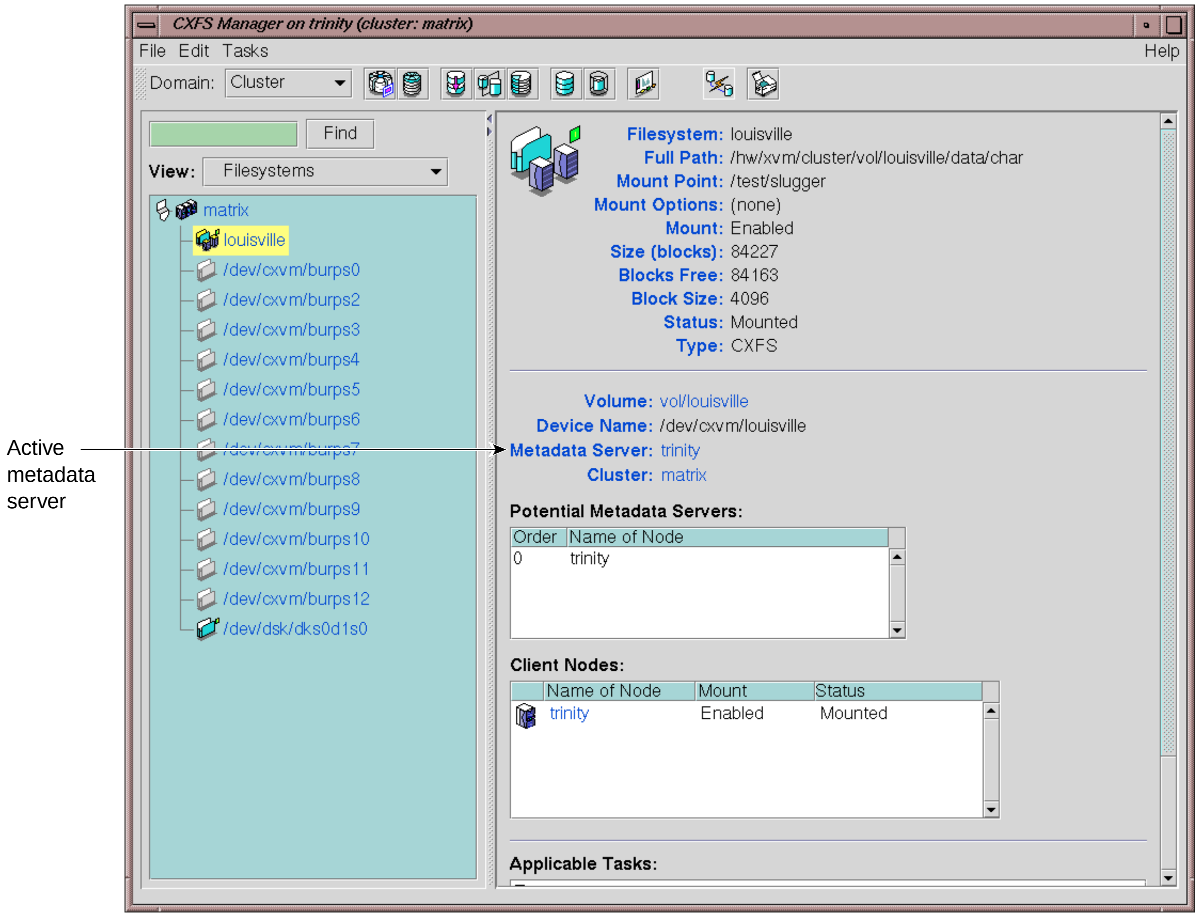
Metadata Server Discovery with the GUI

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 12-1 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

Figure 12-1 Window Showing the Metadata Server

Metadata Server Discovery with `cluster_status`

You can use the `cluster_status` command to discover the active metadata server. For example:

```
# /var/cluster/cmgr-scripts/cluster_status

+ Cluster=cxfs6-8  FailSafe=Not Configured CXFS=ACTIVE               15:15:33
   Nodes =   cxfs6    cxfs7    cxfs8
FailSafe =
    CXFS =      UP       UP       UP

CXFS              DevName           MountPoint           MetaServer     Status
        /dev/cxvm/concat0             /concat0                cxfs7         UP
```

For more information, see "Check Cluster Status with `cluster_status`", page 311.

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
Membership since Thu Mar  1 08:15:39 2001
Node         NodeId     Status    Age   Incarnation     CellId
cxfs6             6         UP      0             0          2
cxfs7             7         UP      0             0          1
cxfs8             8         UP      0             0          0
1 CXFS FileSystems
/dev/cxvm/concat0 on /concat0  enabled  server=(cxfs7)  2 client(s)=(cxfs8,cxfs6)
```

Metadata Server Recovery

Note: In this release, relocation is disabled by default and 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", page 183, or "Modify a Cluster with `cmgr`", page 225.

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 filesystem is 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:

- 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", page 272
- "Normal CXFS Shutdown", page 273
- "Forced CXFS Shutdown: Revoke Membership of Local Node", page 275

If there are problems, see Chapter 16, "Troubleshooting", page 323. For more information about states, Chapter 15, "Monitoring Status", page 307.

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.

To perform a cluster database shutdown, enter the following:

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

If you also want to disable the daemons from restarting at boot time, enter the following:

```
# chkconfig cluster off
```

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:

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

Normal CXFS Shutdown

You should perform a *normal CXFS shutdown* when you want to stop all 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", page 183, or "Modify a Cluster with `cmgr`", page 225. 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 (Normal CXFS Shutdown) with the GUI", page 172.

Note: This action deactivates CXFS services on **one** node, forming a new CXFS kernel membership after deactivating the node. If you want to stop services on multiple nodes, you must enter this command multiple times or perform the task using the GUI.

After you shut down cluster services on a node, the node is marked as inactive and is no longer used when calculating the CXFS kernel membership. See "Node Status", page 313.

Node Status and Normal CXFS Shutdown

After performing normal CXFS shutdown on a node, its state will be `INACTIVE`; therefore, it will not impact CXFS kernel membership quorum calculation. See "Normal CXFS Shutdown", page 273.

When You Should Not Perform a Normal CXFS Shutdown

You should not perform a normal CXFS shutdown 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", page 275.

Rejoining the Cluster after a Normal CXFS Shutdown

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", page 171) 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* 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 cluster filesystems on the local node
- Attempts to access the cluster 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`

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 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 a forced shutdown) will **not** reset the departing node.

To perform a forced CXFS shutdown, 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", page 175. 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 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", page 162, or "Modify a Cluster with `cmgr`", page 225
- "Delete a Node with the GUI", page 167, or "Delete a Node with `cmgr`", page 218

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", page 175.

If you perform a forced 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.

For example:

```
cmgr> admin cxfs_start
```

Serial Hardware Reset Capability and Forced CXFS Shutdown

Caution: If you perform forced shutdown on an administration node with serial hardware reset capability and the shutdown will not cause loss of cluster quorum, the node will be reset (rebooted) by the appropriate node.

For more information about resets, see "Serial Hardware Reset", page 27.

Avoiding a CXFS Restart at Reboot

The IRIX `cxfs_cluster` flag and the Linux 64–bit `cxfs` flag to `chkconfig` controls the `clconfd` daemon on a CXFS administration node. The `cxfs_client` flag controls the `cxfs_client` daemon on a client-only node. On Linux 64-bit nodes, `chkconfig` settings are saved by updating various symbolic links in the `/etc/rc.`*n* directories.

If these settings are turned off, the daemons will not be started at the next reboot and the kernel will not be configured to join the cluster. It is useful to turn them off before rebooting if you want to temporarily remove the nodes from the cluster for system or hardware upgrades or for other maintenance work.

To avoid restarting the cluster database on a CXFS administration node, set the `cluster` option to `off`.

For example, do the following:

- IRIX administration node:

  ```
  irix# /etc/chkconfig cxfs_cluster off
  irix# /etc/chkconfig cluster off
  irix# reboot
  ```

- Linux 64-bit administration node:

  ```
  [root@linux64 root]# /sbin/chkconfig cxfs off
  [root@linux64 root]# /sbin/chkconfig cluster off
  [root@linux64 root]# reboot
  ```

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", page 174.

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", page 84, and "`fs2d.options` on CXFS Administration Nodes", page 86

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.

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. This can be performed on any node.

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 64-bit 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.

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", page 179; for information about `cmgr`, see "Cluster Tasks with `cmgr`", page 222.

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", page 114, and "Define CXFS Filesystems with the GUI", page 183.
- For information about defining and mounting a new filesystem with `cmgr`, see "Modify a Cluster with `cmgr`", page 225.
- For information about mounting a filesystem that has already been defined but is currently unmounted, see "Define a CXFS Filesystem with `cmgr`", page 235.

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", page 187, or "Unmount a CXFS Filesystem with `cmgr`", page 242.

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", page 187, or "Unmount a CXFS Filesystem with `cmgr`", page 242.
2. Mount the filesystem as an XFS filesystem. See *IRIX Admin: Disks and Filesystems*.
3. Use the `xfs_growfs` command or the GUI task; see "Grow a Filesystem with the GUI", page 181.
4. Unmount the XFS filesystem with the `umount` command.
5. Mount the filesystem as a CXFS filesystem. See "Mount CXFS Filesystems with the GUI", page 186, or "Define a CXFS Filesystem with `cmgr`", page 235.

Dump and Restore

You must perform dump and restore procedures from the active metadata server. The `xfsdump` and `xfsrestore` commands make use of special system calls that will only function on the 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 64-bit, 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 64–bit `/var/lib/xfsdump`) that should be shared, rather than the IRIX `/var/xfsdump/inventory` directory (Linux 64-bit `/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.

Cluster Database Backup and Restore

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.

Restoring the Database from Another Node

If the database has been accidentally deleted from an individual node, you can replace it with a copy from another 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 node:

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

2. Run `cdbreinit` on nodes that are missing the cluster database.
3. Restart the daemons by running the following commands on each node:

   ```
   # /etc/init.d/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", page 255.

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:

`/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 for 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 nodes, do the following:

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

On each node:
```
# /usr/cluster/bin/cdbreinit
```

On each node:
```
# /etc/init.d/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 corrupted cluster database. Do the following:

1. Stop CXFS services.
2. Stop cluster services on each node.
3. Remove the old database by using the `cdbreinit` command on each node.
4. Stop cluster services again (these were restarted automatically by `cdbreinit` in the previous step) on each node.
5. Use the `cdbRestore` command on each node.
6. Start cluster services on each node.

For example, to backup the current database, clear the database, and then restore the database to all nodes, do the following:

On each node:
```
# /usr/cluster/bin/cdbBackup
```

On one node in the cluster:
```
# stop cx_services for cluster clusterA
```

On each node:
```
# /etc/init.d/cluster stop
```

On each node:
```
# /usr/cluster/bin/cdbreinit
```

On each node (again):
```
# /etc/init.d/cluster stop
```

On each node:
```
# /usr/cluster/bin/cdbRestore
```

On each node:
```
# /etc/init.d/cluster start
```

For more information, see the `cdbBackup` and `cdbRestore` man page.

`chkconfig` Settings

Note: 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, and starting the processes will start the cluster **only** if the CXFS services are marked as activated in the database.

CXFS has the following flags to the `chkconfig` command:

- On administration nodes, `cluster` controls the other cluster administration daemons, such as the replicated cluster database. If it 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.

- On administration nodes, a flag 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.

  The flag name differs between the operating systems:

  – IRIX: `cxfs_cluster`

  – Linux 64-bit: `cxfs`

- On client-only nodes, the `cxfs_client` flag controls whether or not the `cxfs_client` daemon should be started.

System Tunable Parameters

Table 12-1 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.

Table 12-1 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: `/proc/sys/kernel/cell`
`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: `/proc/sys/kernel/cell`

Parameter	Description	Location
	• 0 - Causes the node waiting for the fence acknowledgement to forcibly withdraw from the cluster, equivalent to a forced 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: `/proc/sys/kernel/cell`
`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: `/proc/sys/kernel/cell`

Parameter	Description	Location
	• 0 - Causes the node waiting for the reset acknowledgement to forcibly withdraw from the cluster, equivalent to a forced 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. 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 16.	IRIX: `/var/sysgen/mtune/cxfs` Linux: `/proc/sys/fs/cxfs`

Parameter	Description	Location
`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: `/proc/sys/fs/cxfs`
`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 not supported in this release.	IRIX: `/var/sysgen/mtune/cxfs` Linux: `/proc/sys/fs/cxfs`

Parameter	Description	Location
`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: `/proc/sys/kernel/cell`
`mtcp_nodelay`	Enables `TCP_NODELAY` on CXFS message channels. SGI recommends that you do not change this value.	IRIX: `/var/sysgen/mtune/cell` Linux: `/proc/sys/kernel/cell`
`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: `/proc/sys/kernel/cell`

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: `/proc/sys/kernel/cell`
`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: `/proc/sys/kernel/cell`

Chapter 13

Coexecution with FailSafe

This chapter discusses the following:

Also see "Communication Paths in a Coexecution Cluster", page 391.

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 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 highly available 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 highly available service.

You can therefore use FailSafe in a CXFS cluster (known as *coexecution*) to provide highly available 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 64-bit). 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 a minimum of 3 server-capable nodes. (A cluster with serial hardware 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 ", page 414.)

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.

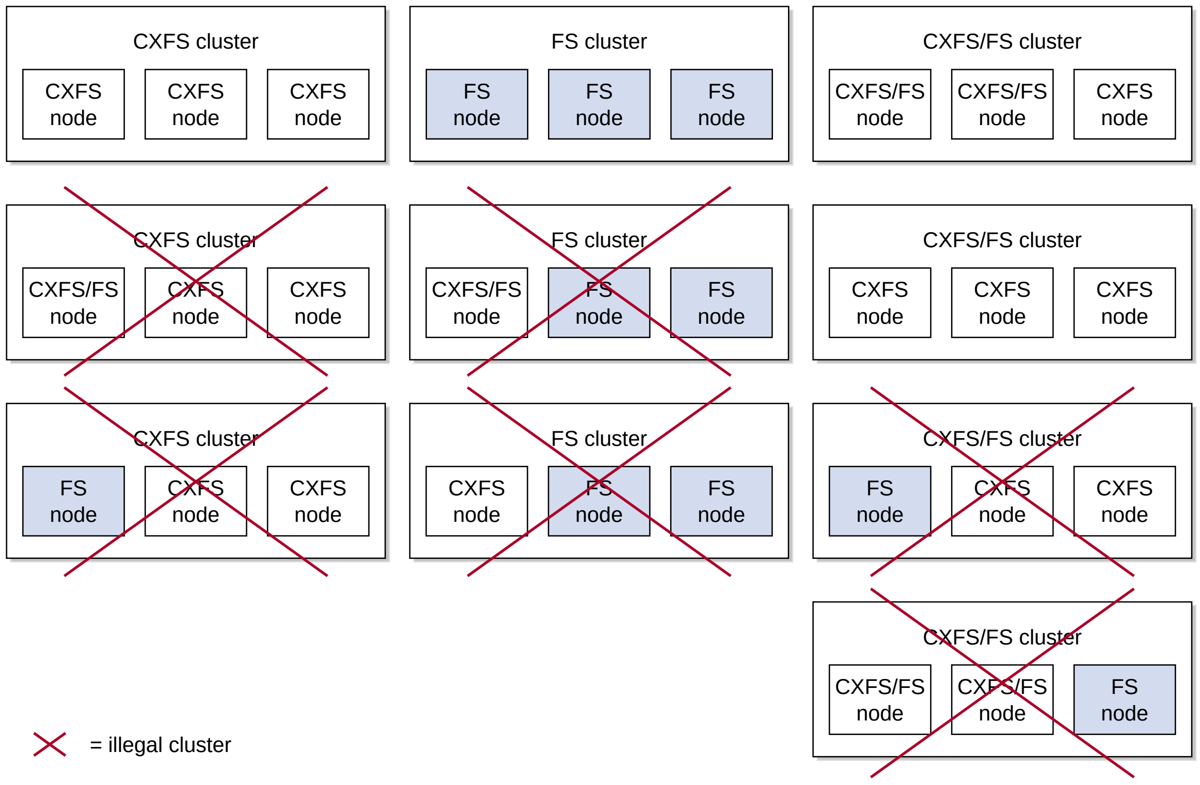


Figure 13-1 Cluster and Node Type Combinations

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", page 150.

However:

- You cannot change the type of a node if the respective high availability (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`", page 217, and "Convert a Cluster to CXFS or FailSafe with `cmgr`", page 226.

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.)

When using FailSafe and CXFS on the same node, only the priority 1 network will be used for CXFS and it must be set to allow both heartbeat and control messages.

Note: CXFS will not fail over to the second network. If the priority 1 network fails, CXFS will fail but FailSafe services may move to the second network if the node is `CXFS and FailSafe`.

If CXFS resets the node due to the loss of the priority 1 network, it will cause FailSafe to remove the node from the FailSafe membership; this in turn will cause resource groups to fail over to other FailSafe nodes in the cluster.

CXFS Tie-Breaker Node and Coexecution

Do not use a CXFS tiebreaker node if you have only two FailSafe nodes.

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 *SGI InfiniteStorage FailSafe Administrator's Guide*.

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. Linux 64-bit and the multiOS 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", page 308
- 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 `cluster_status` 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 6, "IRIX CXFS Installation", page 61.

The following sections describe the procedures for performing some of these tasks:

- "Status in Log Files"
- "Cluster Status", page 310
- "Node Status", page 313
- "XVM Statistics", page 317
- "I/O Fencing Status", page 318
- "Heartbeat Timeout Status", page 320

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 64-bit: `/var/log/cxfs_client`
  - – System log:
    - IRIX: `/var/adm/SYSLOG`
    - Linux 64-bit: `/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 64-bit 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 *SGI InfiniteStorage CXFS MultiOS Client-Only Guide*.

If the disk is filling with log messages, see "Log File Management", page 277.

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 GUI or the `cluster_status`, `clconf_info`, `cmgr`, or `cxfs_info` commands. Also see "Key to Icons and States", page 145

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", page 313.

Check Cluster Status with `cluster_status`

You can use the `cluster_status` command to monitor the cluster using a `curses` interface. For example, the following shows a three-node cluster with a single filesystem mounted and the help text displayed:

```
# /var/cluster/cmgr-scripts/cluster_status
+ Cluster=cxfs6-8  FailSafe=Not Configured CXFS=ACTIVE               15:15:33
   Nodes =   cxfs6    cxfs7    cxfs8
FailSafe =
    CXFS =      UP       UP       UP

CXFS              DevName           MountPoint           MetaServer     Status
        /dev/cxvm/concat0             /concat0                cxfs7         UP

                       +-------+ cluster_status Help +--------+
                       |  on s  - Toggle Sound on event       |
                       |  on r  - Toggle Resource Group View  |
                       |  on c  - Toggle CXFS View            |
                       |     h  - Toggle help screen          |
                       |     i  - View Resource Group detail  |
                       |     q  - Quit cluster_status         |
                       +--- Press 'h' to remove help window --+

----------------------------------------------------------------
cmd('h' for help) >
```

The above shows that a sound will be activated when a node or the cluster changes status. (The `r` and `i` commands are not relevant for CXFS; they are of use only with FailSafe.) You can override the `s` setting by invoking `cluster_status` with the `-m` (mute) option.

The following output shows that the CXFS cluster is up and that `cxfs7` is the metadata server for the `/dev/cxvm/concat0` XVM volume:

```
cxfs6# /var/cluster/cmgr-scripts/cluster_status

+ Cluster=cxfs6-8  FailSafe=Not Configured CXFS=ACTIVE               15:18:28
   Nodes =   cxfs6    cxfs7    cxfs8
FailSafe =
    CXFS =      UP       UP       UP

CXFS              DevName            MountPoint           MetaServer     Status
        /dev/cxvm/concat0             /concat0                cxfs7         UP
```

Check Cluster Status with `clconf_info`

If the cluster is up, you can see detailed information by using `/usr/cluster/bin/clconf_info`.

For example:

```
cxfs6 # clconf_info
Membership since Thu Mar  1 08:15:39 2001
Node         NodeId     Status    Age   Incarnation     CellId
cxfs6             6         UP      0             0          2
cxfs7             7         UP      0             0          1
cxfs8             8         UP      0             0          0
1 CXFS FileSystems
/dev/cxvm/concat0 on /concat0  enabled  server=(cxfs7)  2 client(s)=(cxfs8,cxfs6)
```

Inactive nodes are shown as `DOWN`.

Check Cluster Status with `cmgr`

To query node and cluster status, use the following `cmgr` command:

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", page 400.
- **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:

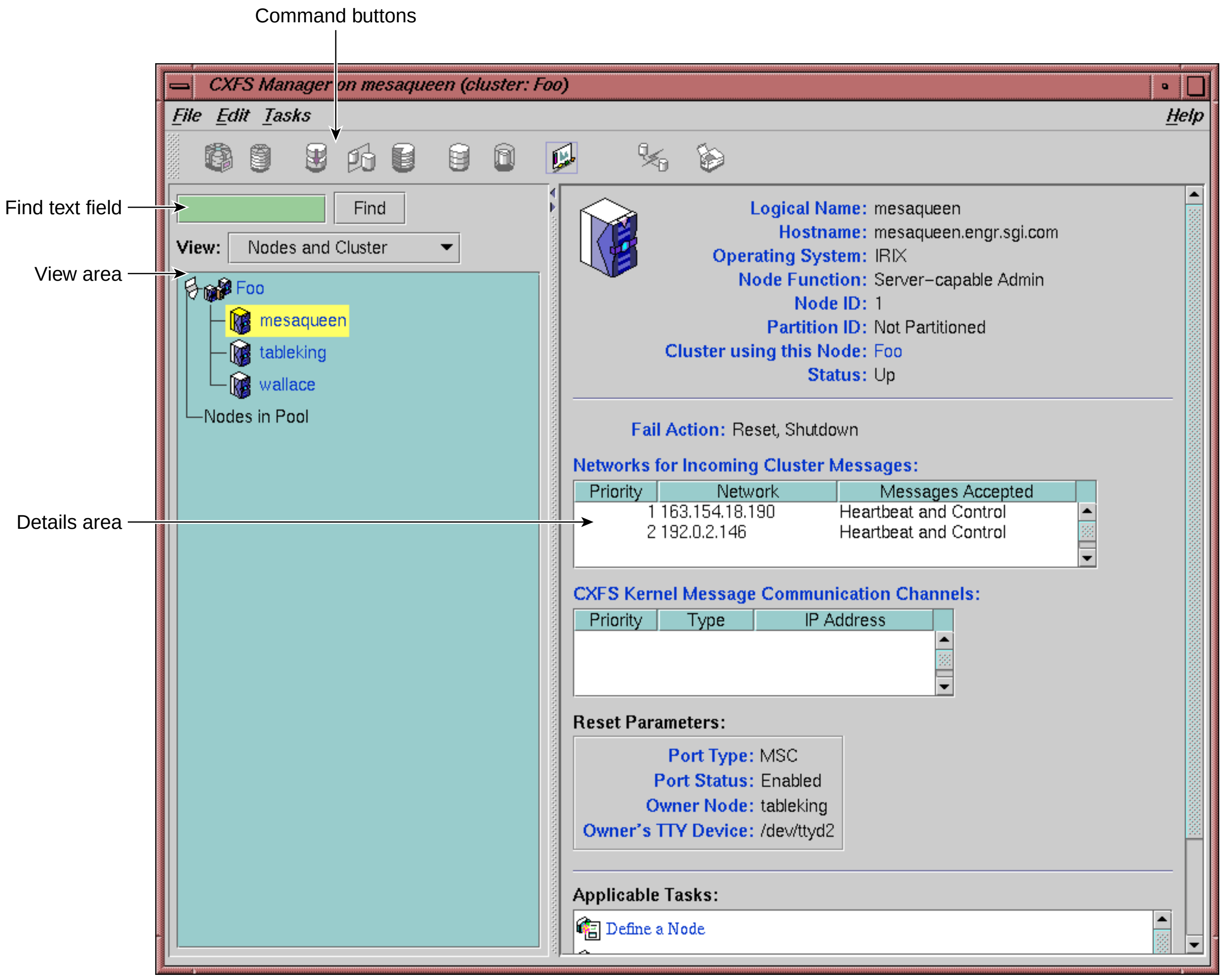
- **INACTIVE** for its own state, because it can determine that the start CXFS services task has not been run
- **UNKNOWN** as the state of all other nodes, because the daemons required to exchange information with other nodes are not running, and therefore state cannot be determined

The following sections provide different methods to monitor node status. Also see "Check Cluster/Node/Filesystem Status with `cxfs_info`", page 313.

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. Figure 15-1, page 315 shows an example of a node that is of type `CXFS only`.



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 15-1 Node Status

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 `cluster_status`

You can use the `cluster_status` command to monitor the status of the nodes in the cluster. For example, the following output shows that all three nodes in the CXFS cluster are up:

```
cxfs6# /var/cluster/cmgr-scripts/cluster_status

+ Cluster=cxfs6-8  FailSafe=Not Configured CXFS=ACTIVE              15:15:33
   Nodes =   cxfs6    cxfs7    cxfs8
FailSafe =
    CXFS =      UP       UP       UP
```

If you toggle the `c` command to `off`, the `CXFS` line will disappear.

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 Serial Hardware 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 6, "IRIX CXFS Installation", page 61.

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 64–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 15-2 shows an example.

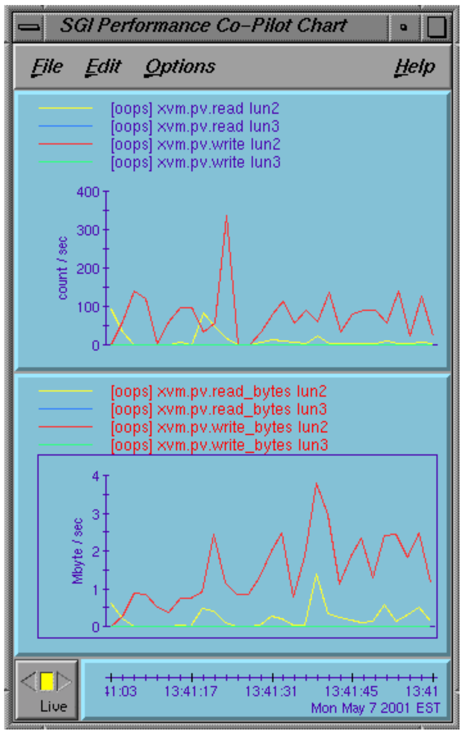


Figure 15-2 pmgxvm chart

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
```

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 evaluation of the `mtcp` expression and prints it out as a decimal value:

```
irix# icrash -e '*(mtcp_stat_t *)&mtcp_stat'
```

For example:

```
typedef struct mtcp_stat {
        lock_t  lock;
        int     discovery_enable;
        cell_time_t discovery;          /* max hz discovery thread slept      */
        cell_time_t multicast;          /* max hz multicast thread slept      */
        cell_time_t monitor;            /* max hz between hb monitors         */
        int     gen_hist[MAX_GEN_BKTS]; /* hb generation dlay histogram       */
        int     alive_small;            /* undersized alive mesgs             */
        int     alive_big;              /* oversized alive mesgs              */
        int     invalid_cell;           /* alive msgs with invalid cell       */
        int     invalid_cluster;        /* alive msgs with wrong cluster id   */
        int     wrong_ipaddr;           /* alive msgs from wrong ipaddr       */
        int     not_configured;         /* alive msgs from cells not in config*/
        int     unknown;                /* alive msgs from cells who haven't  */
                                        /* discovered us                      */
        int     gen;         !          /* generation number                  */
        mtcp_hb_mon_config_t hb_config; /* hb mon config info                 */
        mtcp_cell_hb_stat_t *hb_stats[MAX_CELLS];
```

```
                                /* per cell hb stats                 */
        mtcp_cell_seq_stat_t seq_stats[MAX_CELLS];
                                /* per cell seq # stats              */
        cell_time_t overdue[MAX_CELLS]; /* timestamp when we noticed overdue  */
                                /* heartbeat */
        int     rescues[MAX_CELLS];     /* number of times tcp rescued failure*/
                                /* due to overdue heartbeats */
} mtcp_stat_t;
```

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

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"
- "Avoid Problems", page 333
- "Common Problems", page 345
- "Understanding Error Messages", page 352
- "Corrective Actions", page 372
- "Reporting Problems to SGI", page 379

Administrative tasks must be performed from a node that has the `cluster_admin` software package installed. See the *SGI InfiniteStorage CXFS MultiOS Client-Only Guide* for additional troubleshooting information.

Troubleshooting Strategy

To troubleshoot CXFS problems, do the following:

- "Know the Tools"
- "Avoid Problems", page 333
- "Identify the Cluster Status", page 341
- "Locate the Problem", page 343

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

Understand the following physical storage tools:

- To display the hardware inventory:
  - IRIX:

    ```
    irix# /sbin/hinv
    ```

  - Linux 64-bit (assuming the `sgi-misc` RPM is installed):

    ```
    [root@linux64 root]# /usr/bin/hinv
    [root@linux64 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 64-bit:

  ```
  root@linux64 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", page 136 and Chapter 10, "Reference to GUI Tasks for CXFS", page 129.

  - The `cmgr` command line with prompting:

    ```
    # /usr/cluster/bin/cmgr -p
    ```

    See "`cmgr` Overview", page 196, and Chapter 11, "Reference to `cmgr` Tasks for CXFS", page 195.

- To reinitialize the database, use the `cdbreinit` command:

  ```
  # /usr/cluster/bin/cdbreinit
  ```

  See "Recreating the Cluster Database", page 378.

Cluster Control Tools

Understand the following cluster control tools:

- To start and stop the cluster services daemons:

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

  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.

  See the following:

  – "Cluster Database Membership Quorum Stability", page 334

  – "Restarting CXFS Services", page 373

  – "Clearing the Cluster Database", page 374

  – "Stopping and Restarting Cluster Infrastructure Daemons", page 377

- To start and stop CXFS services, use the GUI or the following `cmgr` commands:

  ```
  cmgr> start cx_services on node hostname for cluster clustername
  cmgr> stop cx_services on node hostname for cluster clustername
  ```

  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`", page 228, and "Cluster Services Tasks with the GUI", page 171.

  Note: In this release, relocation and recovery are supported only when using standby nodes.

- 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", page 175
- "Allow Membership of the Local Node with the GUI", page 175
- "Revoke Membership of the Local Node with `cmgr`", page 234
- "Allow Membership of the Local Node with `cmgr`", page 234

Networking Tools

Understand the following networking tools:

- To send packets to network hosts:
  - IRIX:
    ```
    irix# /usr/etc/ping
    ```
  - Linux 64-bit:
    ```
    [root@linux64 root]# /bin/ping
    ```
- To show network status:
  - IRIX:
    ```
    irix# /usr/etc/netstat
    ```
  - Linux 64-bit:
    ```
    [root@linux64 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", page 108.
- To see cluster and filesystem status, use one of the following:
  - GUI:
    ```
    # /usr/sbin/cxfsmgr
    ```

See "Display a Cluster with the GUI", page 171.

– `cluster_status` command:

```
# /usr/cluster/cmgr-scripts/cluster_status
```

See "Check Cluster Status with `cluster_status`", page 311.

– `clconf_info` command:

```
# /usr/cluster/bin/clconf_info
```

– `cxfs_info` command on an IRIX or Linux 64-bit client-only node:

```
# /usr/cluster/bin/cxfs_info
```

- To see the mounted filesystems:

  – IRIX:

  ```
  irix# /sbin/mount
  irix# /usr/sbin/df
  ```

  – Linux 64-bit:

  ```
  [root@linux64 root]# /bin/mount
  [root@linux64 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 64-bit 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", page 308.

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: The `cxfsdump` command requires access to AIX, IRIX, Linux 64-bit, Linux 32-bit, and Solaris nodes in the cluster via the `rcp` and `rsh` commands. Because these commands are not provided on Windows nodes, the `cxfsdump` command must be run manually on each Windows node.

The `cxfsdump /?` command displays a help message on Windows nodes. The `cxfsdump -help` command displays a help message on other nodes.

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.

If your cluster contains Windows nodes, you must run the command manually on each Windows node.

To gather information about just the local mode, use the `cxfsdump -local` command.

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", page 334
- "Eliminate a Residual Cluster", page 334
- "Cluster Database Membership Quorum Stability", page 334
- "Consistency in Configuration", page 335
- "Define Node Function Appropriately", page 335
- "GUI Use", page 335
- "Log File Names and Sizes", page 336
- "IRIX: Netscape and the Brocade Switch GUI", page 336
- "Performance Problems with Unwritten Extent Tracking and Exclusive Write Tokens", page 336
- "Avoid Excessive Filesystem Activity Caused by the `crontab` File", page 337
- "Use System Capacity Wisely", page 338
- "Reboot Before Changing Node ID or Cluster ID", page 338
- "Remove Unused Nodes", page 339
- "Restart CXFS after a Forced Shutdown", page 339
- "Remove Serial Hardware Reset Lines", page 339

- "Appropriate Use of `xfs_repair`", page 340

Proper Start Up

Ensure that you follow the instructions in "Preliminary Cluster Configuration Steps", page 107, 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 64-bit:

  ```
  [root@linux64 root]# ps -ef | grep cmsd
  ```

If the output shows a `cmsd` kernel thread, force a 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 infrastructure 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:

```
# /etc/init.d/cluster stop
# /etc/init.d/cluster 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", page 83.

Define Node Function Appropriately

Use the appropriate node function definition:

- Use an odd number of server-capable nodes and an odd number of CXFS administration nodes for stability.
- 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", page 130.

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", page 277. 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", page 174, or "Configure Log Groups with `cmgr`", page 231.

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", page 355.

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", page 378.

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", page 169
- "Modify a Cluster with `cmgr`", page 225
- "Delete a Node with `cmgr`", page 218

Restart CXFS after a Forced Shutdown

If you perform a forced shutdown on a node, you must restart CXFS 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. Therefore, you may want to do this after resetting the database or after stopping CXFS services.

For example, enter the following on the 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", page 275.

Remove Serial Hardware Reset Lines

When serial hardware reset is enabled, CXFS requires a `reset successful` message before it moves the metadata server. Therefore, if you have the serial hardware 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", page 163, or "Modify a Node with `cmgr`", page 212.

Note: The reset capability is **mandatory** to ensure data integrity for clusters with only two server-capable nodes, and it is highly recommended for all server-capable nodes. Larger clusters should have an odd number of server-capable and CXFS administration nodes. See "Cluster Environment", page 8.

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 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", page 108.
- 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 `cluster_status`", page 311, "Check Cluster Status with `cmgr`", page 312, and the following files:
  - IRIX: `/var/adm/SYSLOG`
  - Linux 64-bit: `/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", page 324.
  - Are the filesystems mounted on all nodes? Use `mount` and `clconf_info` commands.
  - Which node is the metadata server for each filesystem? Use the `cluster_status` or `clconf_info` commands.

  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 64-bit: `/var/log/messages`

  See "Kernel Status Tools", page 329. 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/*
  ```

Locate the Problem

To locate the problem, do the following:

- Examine the log files (see "Log Files", page 331):
  - Search for errors in all log files. See "Status in Log Files", page 308. 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", page 329.
- 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", page 167.
- 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
  ```

  This command displays the following fields:

  - Node name

   - Node ID
   - Status (up or down)
   - Age (not useful; ignore this field)
   - Incarnation (not useful; ignore this field)
   - Cell ID, which is a number that is dynamically allocated by the CXFS software when you add a node to a cluster (the user does not define a cell ID number). To see the cell ID, use the `clconf_info` command.

  For example:

```
# /usr/cluster/bin/clconf_info
Membership since Fri Sep 10 08:57:36 1999
Node         NodeId     Status    Age   Incarnation     CellId
cxfs6           1001        UP      1             7          0
cxfs7           1002        UP      0             0          1
cxfs8           1003        UP      0             0          2
2 CXFS FileSystems
/dev/xvm/test1 on /mnts/test1  disabled  server   0 clients
/dev/xvm/test2 on /mnts/test2  disabled  server   0 clients
```

- 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 64-bit: `/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
```

Common Problems

The following are common problems and solutions.

Cannot Access Filesystem

If you cannot access a filesystem, check the following:

- Is the filesystem enabled? Check the GUI, `clconf_info` command, and `cluster_status` commands.
- 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", page 107
  - "License Error", page 356
  - Chapter 6, "IRIX CXFS Installation", page 61
- Are the cluster daemons running? See "Verify that the Cluster Daemons are Running", page 108.

- Are the `tcpmux` and `tcpmux/sgi_sysadm` services enabled in the following files?
  - IRIX: `/etc/inetd.conf`
  - Linux 64-bit: `/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", page 277. You may also want to consider choosing a less-verbose log level; see the following:

- "`cad.options` on CXFS Administration Nodes", page 84
- "`fs2d.options` on CXFS Administration Nodes", page 86
- "Configure Log Groups with the GUI", page 174

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", page 106.

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 64-bit: `/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", page 83.

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", page 83.

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`", page 340, 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 after Renaming Cluster

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 64-bit: `/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 64-bit platform.

Note: 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 Brocade Fibre Channel 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 Brocade Fibre Channel 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 Brocade Fibre Channel 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 Brocade Fibre Channel switch, you 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 the "Make Filesystems with the GUI", page 179 task) 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", page 188, and "Delete a CXFS Filesystem with `cmgr`", page 247.

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", page 353
- "`clconfd` Daemon Death", page 355
- "Out of Logical Swap Space", page 355
- "No Cluster Name ID Error", page 355
- "Lost CXFS Membership", page 356
- "License Error", page 356
- "IP Address Error", page 357
- "System Log File Errors", page 358
- "Log File Error Messages", page 367

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
```

Node 0 and node 1 are in the CXFS membership; node 0 has been in the last 21 CXFS memberships, node 1 has been in the last 12. A membership is formed each time a node is added or deleted from the quorum; this number is also incremented when a membership is verified during the quorum change process, therefore the numbers are not sequential. This is an informative message; no corrective action is required.

```
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. Serial hardware reset lines are mandatory for clusters with only two server-capable nodes and highly recommended for all server-capable nodes; larger clusters should have an odd number of server-capable nodes.

```
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 IRIS 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.

`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", page 356.

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 6, "IRIX CXFS Installation", page 61.

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", page 338.

The cluster daemons could also be leaking memory in this case. You may need to restart them:

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

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", page 175, or "Allow Membership of the Local Node with `cmgr`", page 234.

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 the following error on the console when trying to run CXFS:

```
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/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 6, "IRIX CXFS Installation", page 61.

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", page 334. 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 64-bit: `/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 16-1 shows the parts of the preceding message.

Table 16-1 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 Infrastructure Daemons", page 377.

`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", page 374.

If the database exists, restart the cluster daemons; see "Stopping and Restarting Cluster Infrastructure Daemons", page 377.

```
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 Infrastructure Daemons", page 377.

```
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. See "Define a Node with the GUI", page 152, or "Modify a Node with `cmgr`", page 212.

```
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 Infrastructure Daemons", page 377.

If the problem persists, clear the database, reboot, and re-create the database. See "Clearing the Cluster Database", page 374.

`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", page 378.

`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", page 84.

`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", page 84.

`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:

   ```
   # /etc/init.d/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", page 378.

```
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 Infrastructure Daemons", page 377.

If the problem persists, clear the database, reboot, and re-create the database. See "Clearing the Cluster Database", page 374.

```
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 Infrastructure Daemons", page 377.

If the problem persists, clear the database, reboot, and re-create the database. See "Clearing the Cluster Database", page 374.

```
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 Infrastructure Daemons", page 377.

If the problem persists, clear the database, reboot, and re-create the database. See "Clearing the Cluster Database", page 374.

```
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 Infrastructure Daemons", page 377.

If the problem persists, clear the database, reboot, and re-create the database. See "Clearing the Cluster Database", page 374.

```
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", page 378.

```
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", page 173.

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 16-2, page 368, shows the parts in the preceding message.

Table 16-2 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 Infrastructure Daemons", page 377.

```
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 Infrastructure Daemons", page 377.

```
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 Infrastructure Daemons", page 377.

```
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`(1M) 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`(1M) daemon is running.

```
CI_IPCERR_NOPULSE, ipcclnt_pulse_internal():  server failed to
pulse
```

The cluster state could not be determined. Check if the `clconfd`(1M) 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.

Corrective Actions

This section covers the following corrective actions:

- "Restarting CXFS Services", page 373
- "Clearing the Cluster Database", page 374
- "Rebooting", page 375
- "Recovering a Two-Node Cluster", page 375

Restarting CXFS Services

If CXFS services to do not restart after a reboot, it may be that the start flag was turned off by using the stop function of the GUI or the `cmgr` command. In this case, issuing a `/etc/init.d/cluster start` will not restart the services. You must 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", page 171, or "Start CXFS Services with `cmgr`", page 229.

Clearing the Cluster Database

To clear the cluster database on all the nodes of the cluster, do the following, completing each step on each node before moving to the next step:

⚠ **Caution:** This procedure deletes all configuration information.

---

1. Enter the following on all nodes:

```
# /usr/cluster/bin/cmgr -c 'admin cxfs_stop'
```

2. Enter the following on all nodes:

```
# /etc/init.d/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 nodes:

```
# /usr/cluster/bin/cdbreinit
```

   See also "Reboot Before Changing Node ID or Cluster ID", page 338.

4. Enter the following on all nodes:

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

5. Enter the following on all nodes:

```
# /usr/cluster/bin/cmgr -c 'admin cxfs_start'
```

See "Eliminate a Residual Cluster", page 334, 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 64-bit, see the *SGI InfiniteStorage CXFS MultiOS Client-Only Guide*.

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", page 171, and "Start CXFS Services with `cmgr`", page 229.

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", page 400.

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", page 173.
   - See "Set the Tiebreaker Node with `cmgr`", page 230.

2. Revoke the CXFS kernel membership of `node2`:

   - See "Revoke Membership of the Local Node with the GUI", page 175.
   - In `cmgr`, enter:

     ```
     cmgr> admin cxfs_stop
     ```

     See "Revoke Membership of the Local Node with `cmgr`", page 234.

3. Allow CXFS kernel membership of `node2`:

   - See "Allow Membership of the Local Node with the GUI", page 175.
   - In `cmgr`, enter:

     ```
     cmgr> admin cxfs_start
     ```

     See "Allow Membership of the Local Node with `cmgr`", page 234.

4. Unset the CXFS tiebreaker node capability.

Caution: If the CXFS tiebreaker node in a cluster with 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. The serial hardware reset capability is **mandatory** to ensure data integrity for clusters with only two server-capable nodes and highly recommended for all server-capable nodes. Larger clusters should have an odd number of server-capable nodes, or must have serial hardware reset lines or use I/O fencing with switches if only two of the nodes are server-capable nodes.

Use either the GUI or `cmgr`:

- "Set Tiebreaker Node with the GUI", page 173
- "Set the Tiebreaker Node with `cmgr`", page 230

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 `cluster` flag to `chkconfig` controls the other cluster administration daemons and the replicated cluster database. If it 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 cluster daemons are causing serious trouble and prevent the machine from booting, you can recover the node by booting in single-user mode, turning off the `cluster` flag, and booting in multiuser mode:

- IRIX:

  ```
  irix# init 1
  irix# /etc/chkconfig cluster off
  irix# init 2
  ```

- Linux 64-bit:

  ```
  [root@linux64 root]# init 1
  [root@linux64 root]# /bin/chkconfig cluster off
  [root@linux64 root]# init 3
  ```

Stopping and Restarting Cluster Infrastructure Daemons

To stop and restart cluster infrastructure daemons, enter the following:

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

These commands affect the cluster infrastructure daemons only.

Caution: When the cluster infrastructure 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 infrastructure daemons should remain running at all times. Use these commands only as directed.

See also "Restarting CXFS Services", page 373. For general information about the daemons, see "Daemons", page 383.

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", page 338.

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. (The `224.0.0.250` IP address will only respond if CXFS 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
```

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
/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 64-bit Problems

Retain the following information for Linux 64-bit nodes:

- The kernel you are running:

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

- The CXFS packages you are running:

  ```
  [root@linux64 root]# rpm -q cxfs_client cxfs-modules cxfs_utils xvm-cmds
  ```

- The number and types of processors in your machine:

  ```
  [root@linux64 root]# cat /proc/cpuinfo
  ```

- The hardware installed on your machine:

  ```
  [root@linux64 root]# /sbin/lspci
  ```

- Modules that are loaded on your machine:

  ```
  [root@linux64 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", page 329
- 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
    ```

CXFS Software Architecture

This appendix discusses the following for administration nodes:

- "Daemons"
- "Communication Paths", page 386
- "Communication Paths in a Coexecution Cluster", page 391
- "Flow of Metadata for Reads and Writes", page 392

Also see the *SGI InfiniteStorage CXFS MultiOS Client-Only Guide*.

Daemons

The following table lists the CXFS daemons and threads.

Note: CXFS shares with XFS the IRIX `xfsd` and Linux 64-bit `xfsdatad` kernel threads to push buffered writes to disk.

If you are using a coexecution (of type `CXFS and FailSafe`) cluster, see the *SGI InfiniteStorage FailSafe Administrator's Guide*, for information about FailSafe daemons.

Note: On Linux 64-bit, the process names begin with a `*` (such as `[*mtcp_notify]`).

Table A-1 CXFS Daemons and Threads

Layer	Subsystem	Process	Description
Cluster services (CXFS)	`cluster_services`	`clconfd`	CXFS administration cluster configuration daemon. 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-only cluster configuration daemon. 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.

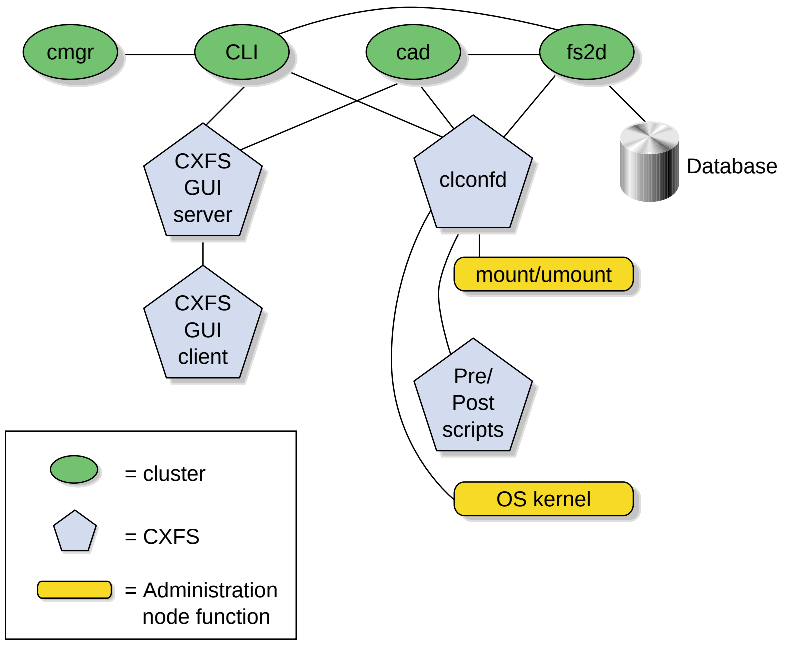


Figure A-1 Communication within One Administration Node

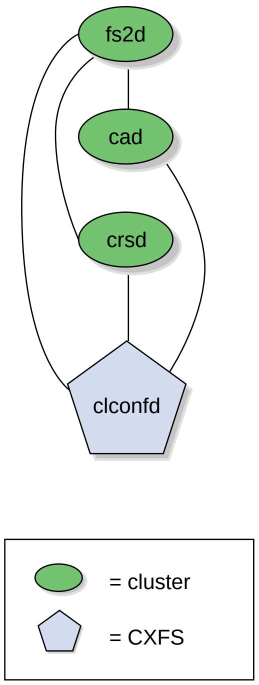


Figure A-2 Daemon Communication within One Administration Node

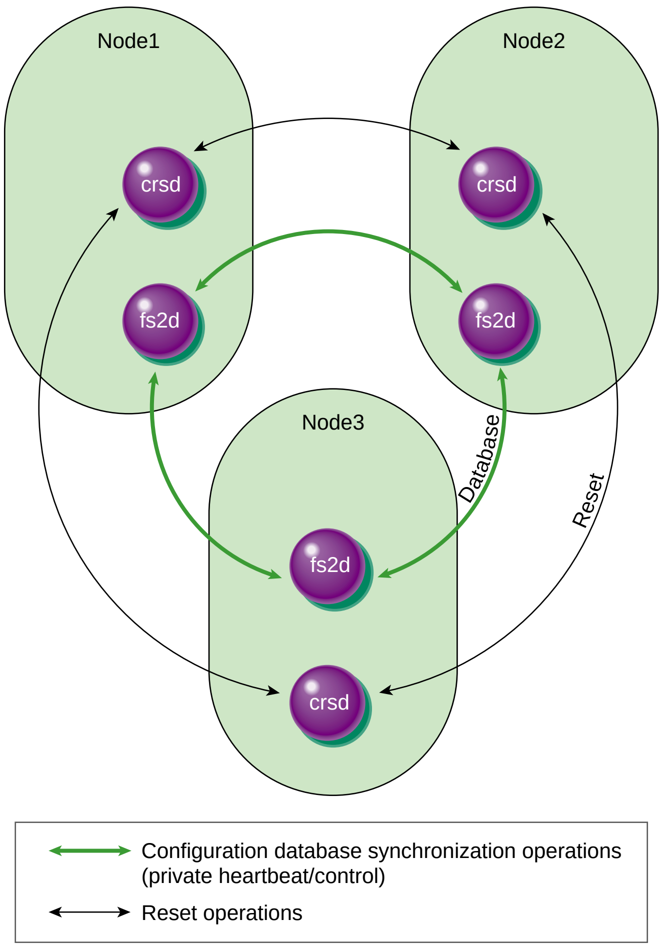

Node1
Node2
Node3
crsd
fs2d
crsd
fs2d
fs2d
crsd
Database
Reset
Configuration database synchronization operations (private heartbeat/control)
Reset operations

Figure A-3 Communication between Nodes in the Pool

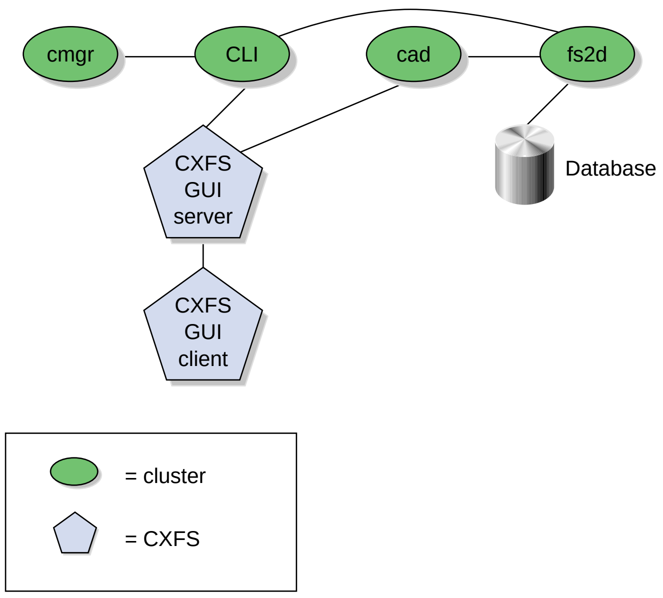


Figure A-4 Communication for an Administration Node Not in a Cluster

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.

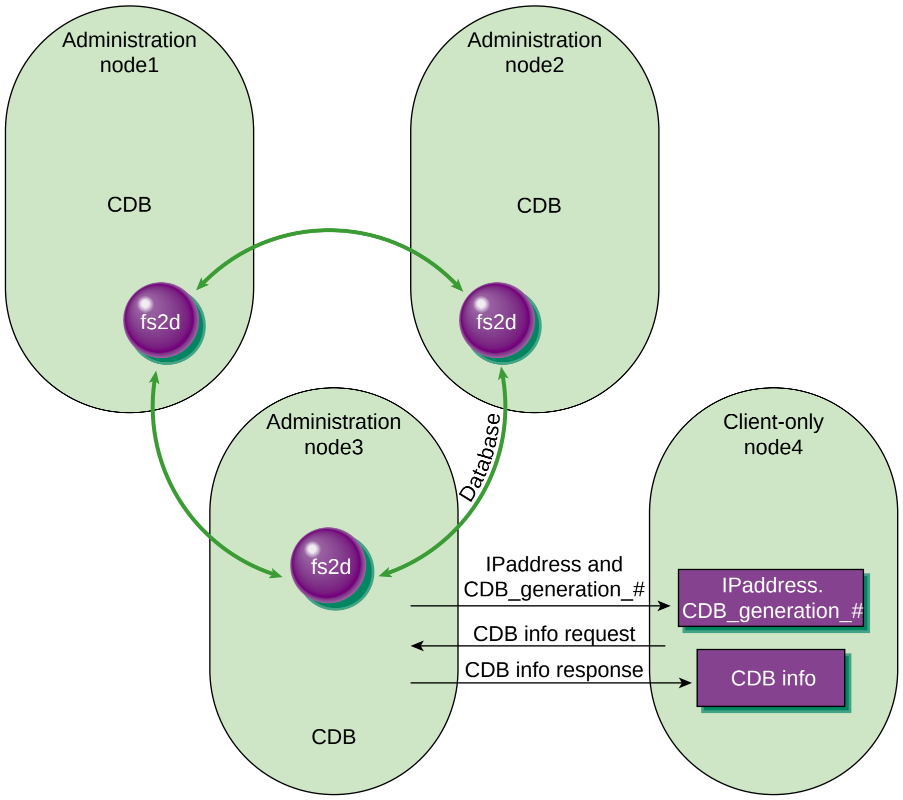


Figure A-5 Communication Among Administration Nodes and Client-Only Nodes

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.

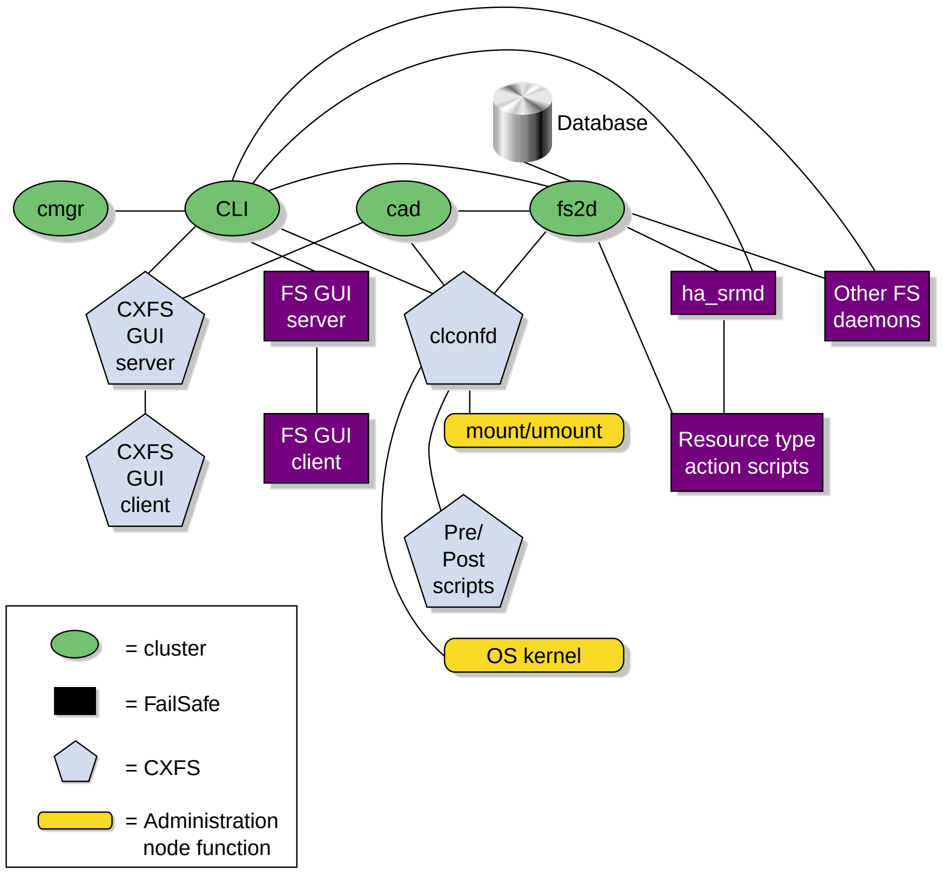


Figure A-6 Administrative Communication within One Administration Node under Coexecution

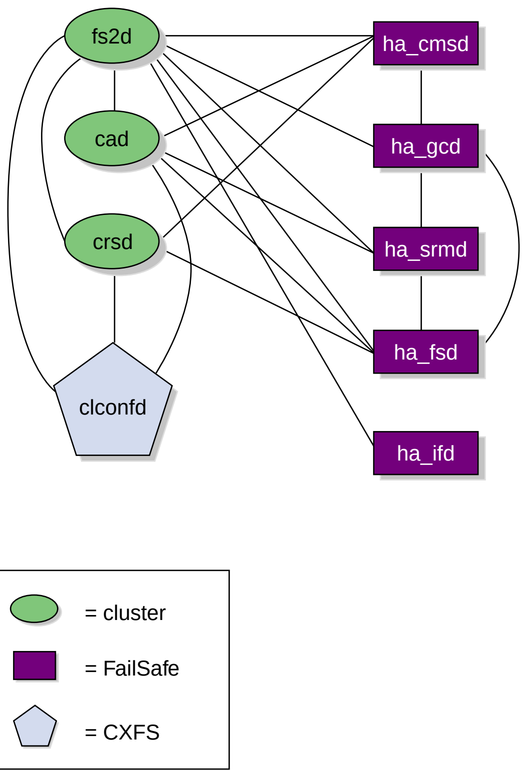


Figure A-7 Daemon Communication within One Administration Node under Coexecution

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.

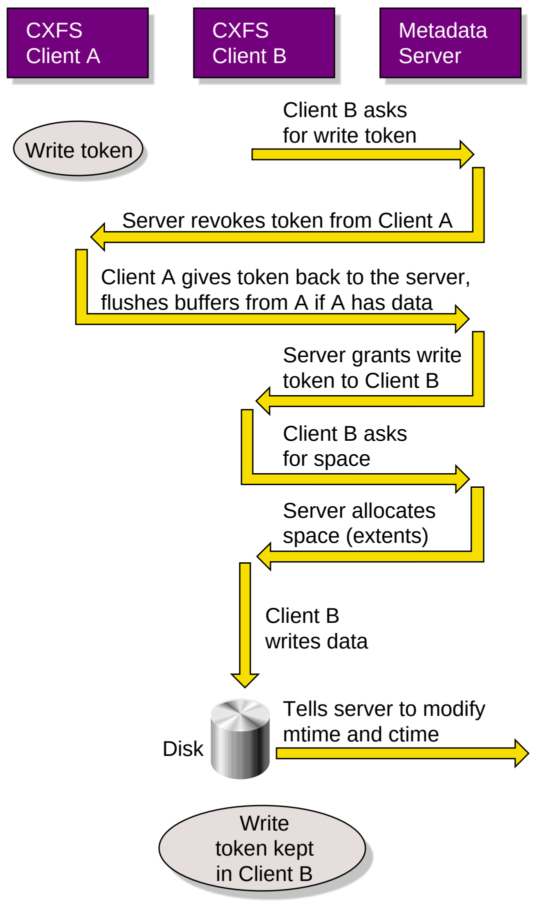

CXFS Client A
CXFS Client B
Metadata Server
Write token
Client B asks for write token
Server revokes token from Client A
Client A gives token back to the server, flushes buffers from A if A has data
Server grants write token to Client B
Client B asks for space
Server allocates space (extents)
Client B writes data
Disk
Tells server to modify mtime and ctime
Write token kept in Client B

Figure A-8 Metadata Flow on a Write

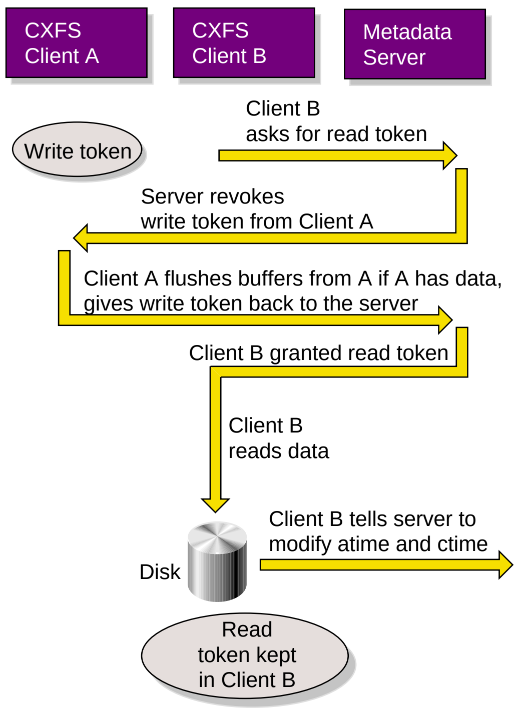

CXFS Client A
CXFS Client B
Metadata Server
Write token
Client B asks for read token
Server revokes write token from Client A
Client A flushes buffers from A if A has data, gives write token back to the server
Client B granted read token
Client B reads data
Disk
Client B tells server to modify atime and ctime
Read token kept in Client B

Figure A-9 Metadata Flow on a Read on Client B Following a Write on Client A

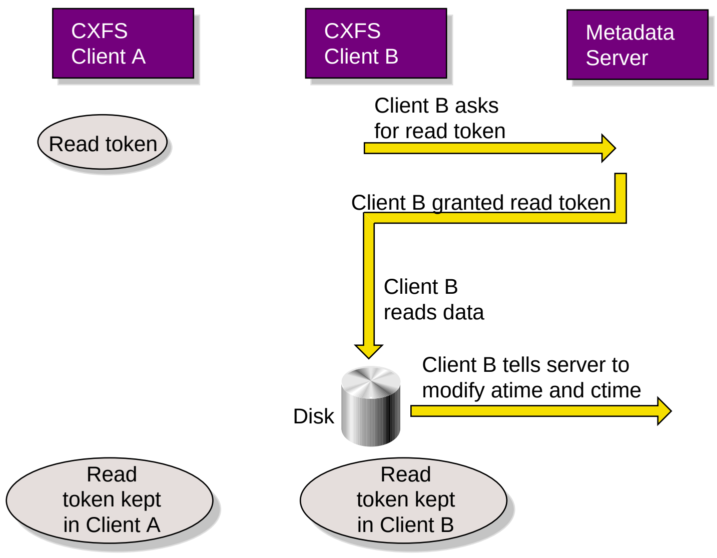

CXFS Client A
CXFS Client B
Metadata Server
Read token
Client B asks for read token
Client B granted read token
Client B reads data
Client B tells server to modify atime and ctime
Disk
Read token kept in Client A
Read token kept in Client B

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 the 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", page 399
- "CXFS Kernel Membership, Quorum, and Tiebreaker", page 400
- "FailSafe Membership, Quorum, and Tiebreaker", page 402

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 (CXFS nodes are limited to a single heartbeat network).

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 quorum allows an initial membership to be formed when a majority (>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", page 403.) The 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.

If you do not use serial hardware reset or I/O fencing to prevent a problem node from accessing I/O devices, you should set a CXFS tiebreaker node to avoid multiple CXFS kernel memberships in the event of a network partition. In CXFS, there is no default tiebreaker. Any node in the cluster can be a CXFS tiebreaker node. You can set the tiebreaker node by using the GUI's **Set Tiebreaker Node** task or by using the `set tie_breaker` command in `cmgr`.

Note: Suppose you have a cluster with only two server-capable nodes with one of them being the CXFS tiebreaker node. If the tiebreaker node fails or if the administrator stops CXFS services on the tiebreaker node, the other node will not be able to maintain a membership and will do a forced shutdown of CXFS services, which unmounts all CXFS filesystems.

If I/O fencing or serial hardware 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 serial hardware reset on a node in the other set. *Serial hardware 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 *SGI InfiniteStorage FailSafe Administrator's Guide*. 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, page 405, 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 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.

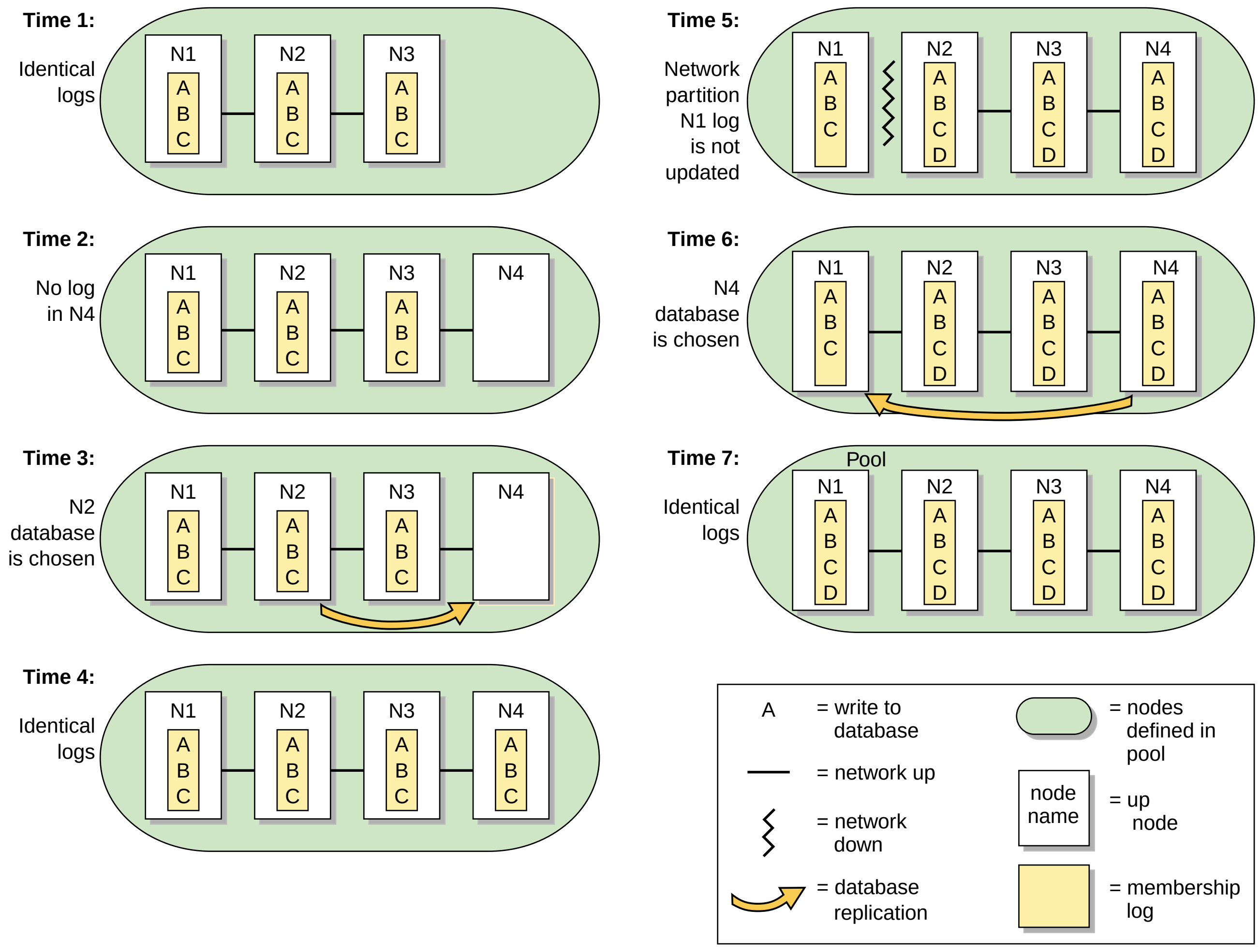


Figure B-1 One Node is Out of Date: Most Recent Log is Replicated

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, page 406 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.

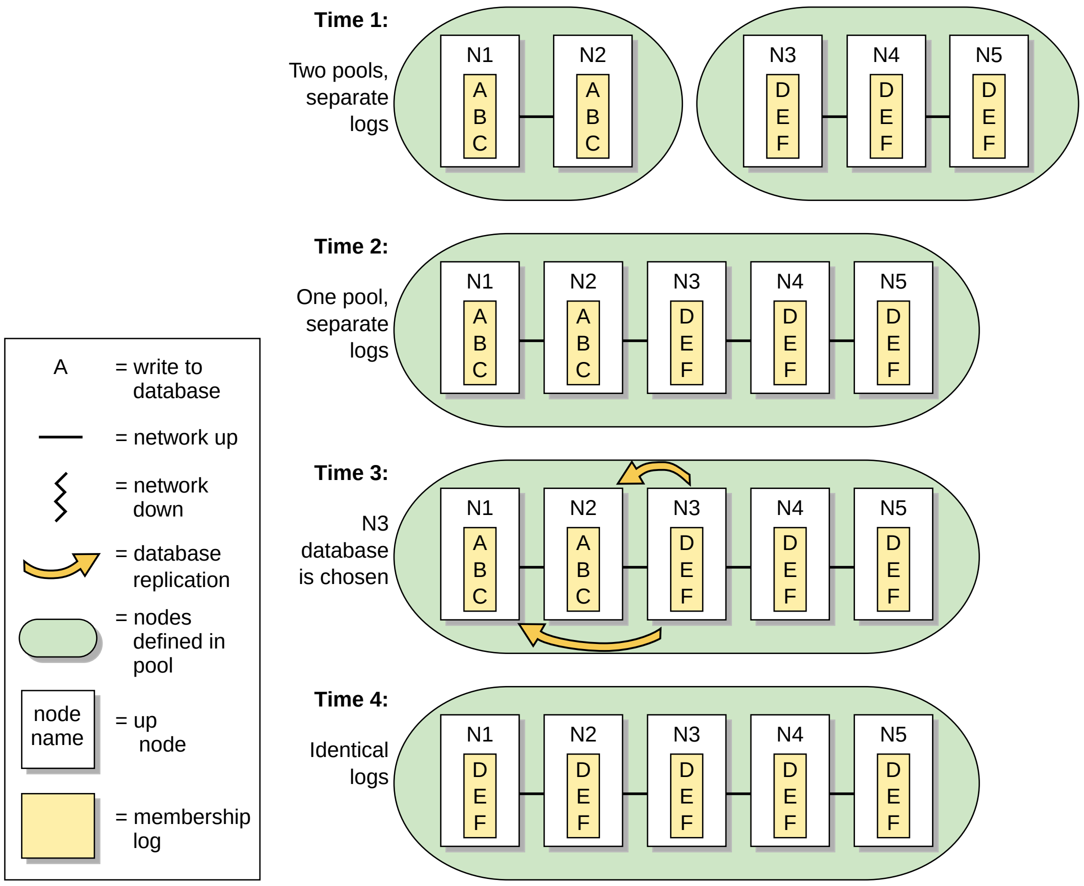


Figure B-2 Unequally Sized Pools are Joined: Log from Larger Pool is Replicated

Figure B-3, page 408, 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.

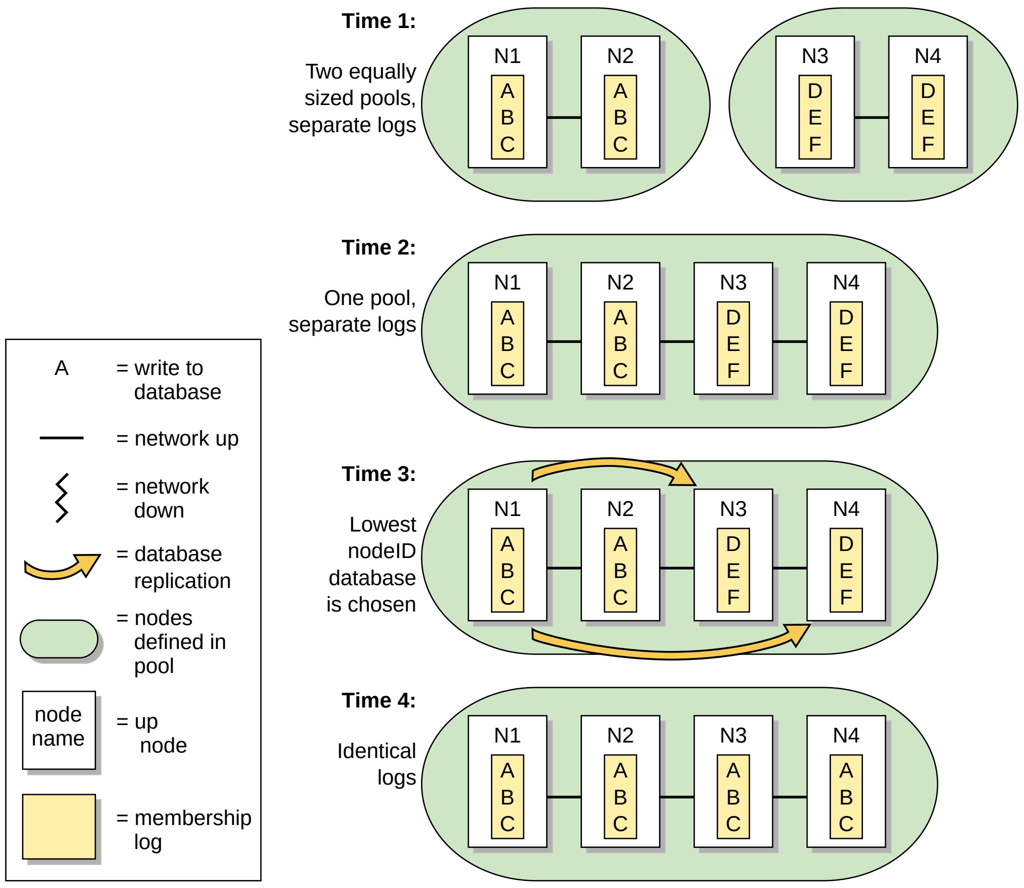


Figure B-3 Equally Sized Pools are Joined: Log from Node with Lowest Node ID is Replicated

Quorum and Tiebreaker Examples

Changing CXFS Kernel Membership Quorum Example

Figure B-4, page 410, 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, page 410, 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.

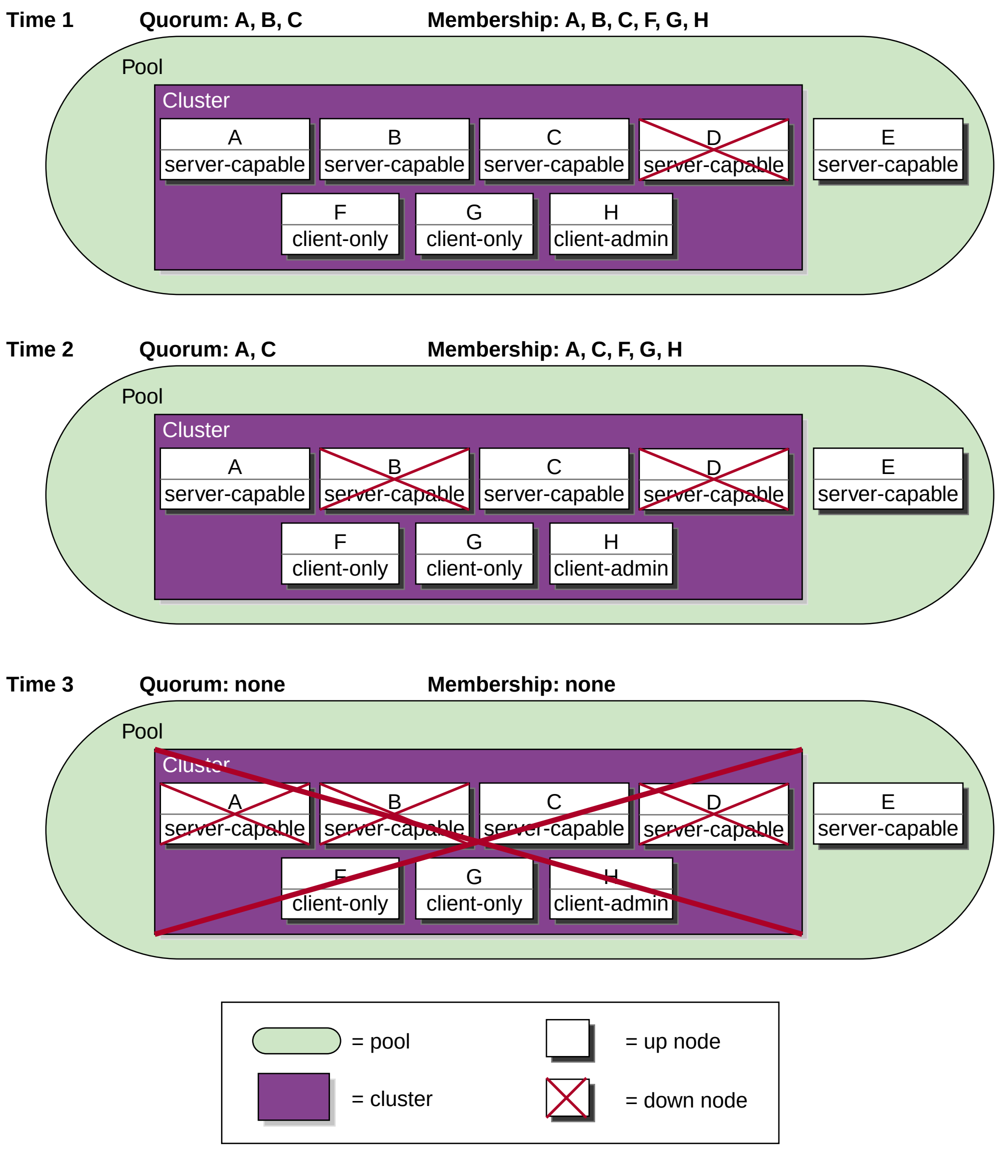
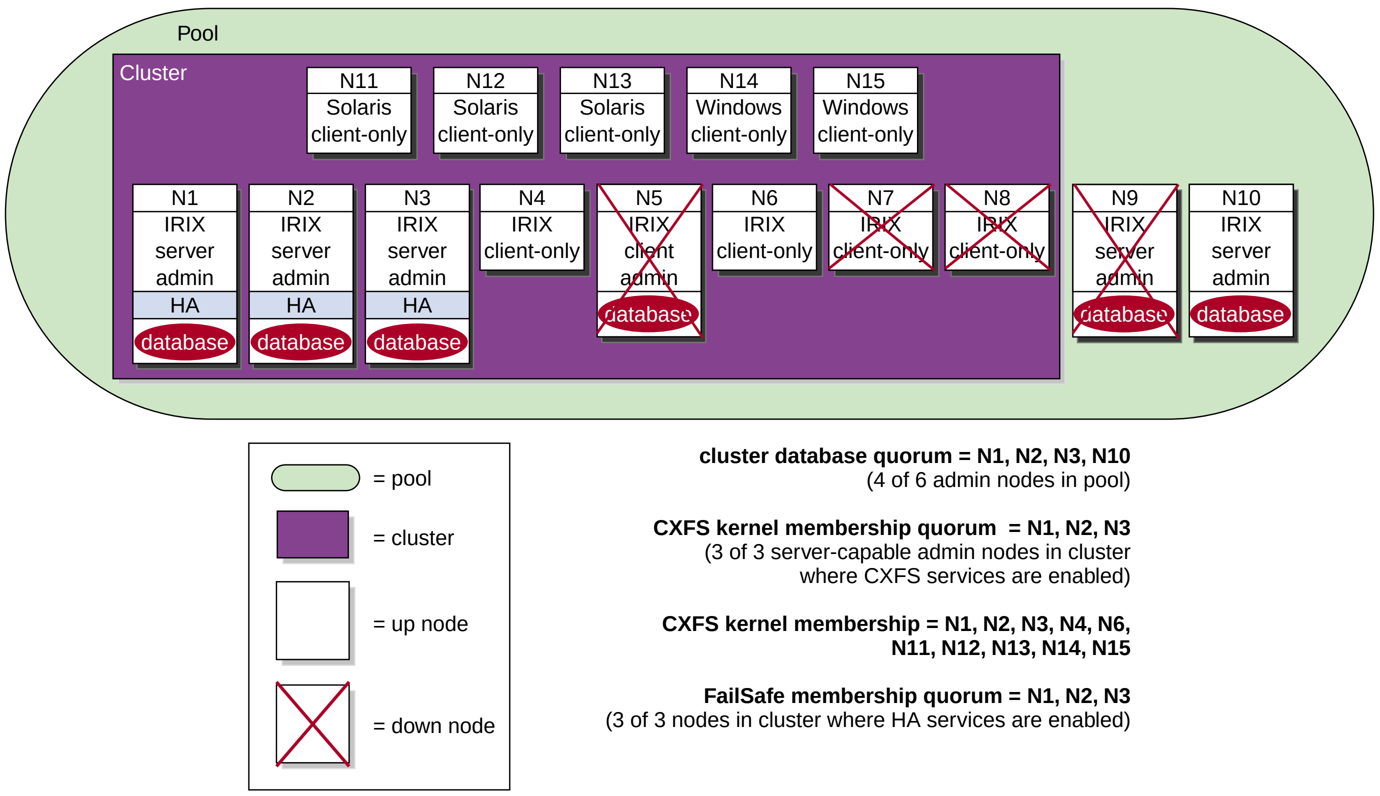


Figure B-4 Changing Quorum for CXFS Kernel Membership

Coexecution Example

Figure B-5, page 412, 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)



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, page 413, 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.

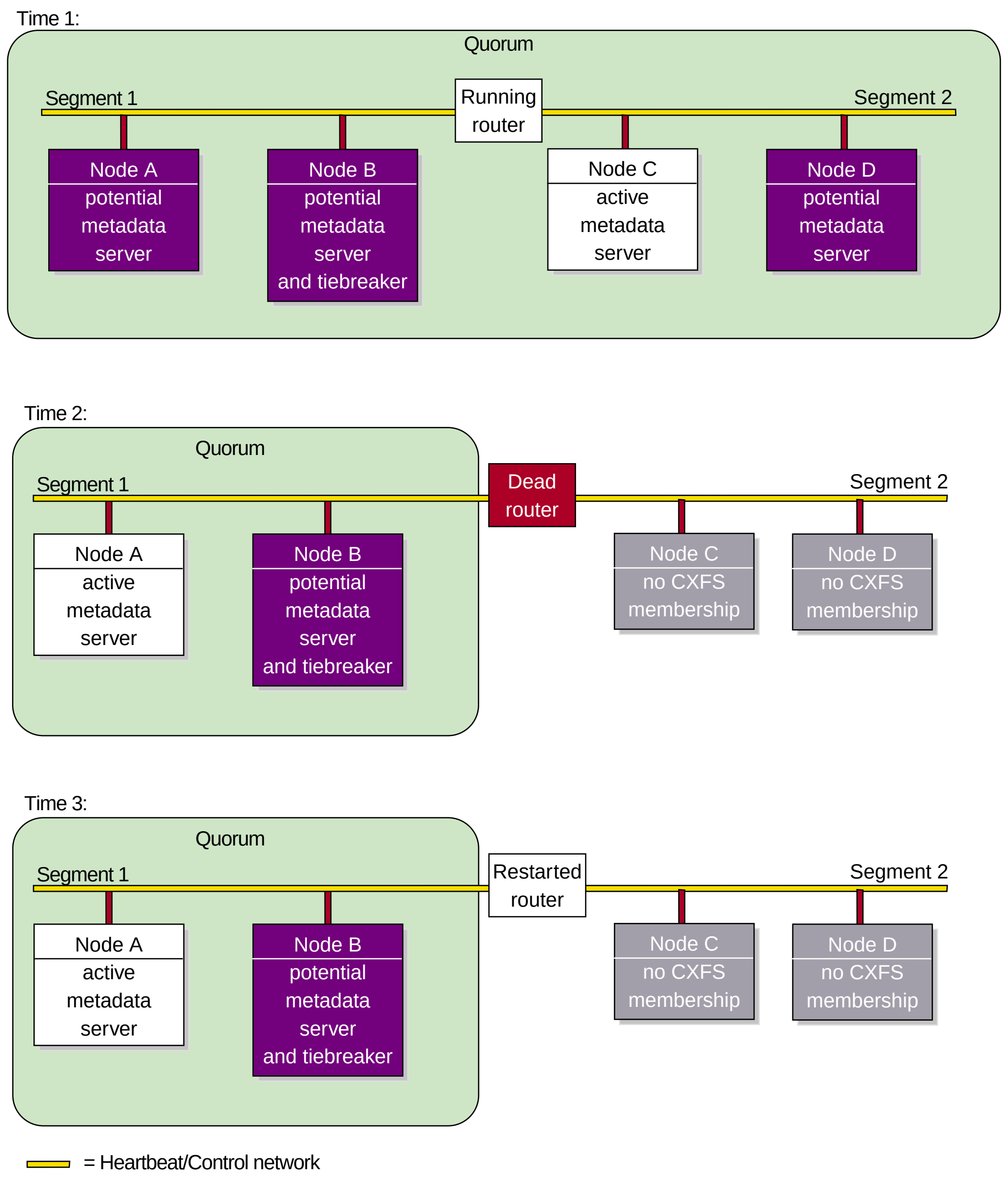

Time 1:
Quorum
Segment 1
Running router
Segment 2
Node A potential metadata server
Node B potential metadata server and tiebreaker
Node C active metadata server
Node D potential metadata server
Time 2:
Quorum
Segment 1
Dead router
Segment 2
Node A active metadata server
Node B potential metadata server 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 and cluster database memberships will continue using just the next priority network, but 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 serial hardware reset lines or I/O fencing to ensure protection of data and guarantee that only one node is running in error conditions. The reset capability or I/O fencing is mandatory to ensure data integrity for clusters with only two server-capable nodes and it is highly recommended for all server-capable nodes. Larger clusters should have an odd number of server-capable nodes, or must have serial hardware reset lines or I/O fencing with switches if only two of the nodes are server-capable.

- 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` flag 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` flag:

   ```
   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
```

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 Chapter 16, "Troubleshooting", page 323, 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", page 2.)

[ ] Connect the SAN hardware. See the following documents (only available to SGI service personnel):

- *SGI Storage Area Network Installation Instructions*
- *SGI TP9400 RAID Installation and Upgrade Guide*
- *SGI Total Performance 9100 Storage System Installation Instructions*
- *IRIS FailSafe Version 2 Installation and Maintenance Instructions* (hardware configuration information largely applies)

[ ] Is there a private network? This is a requirement. SGI recommends a network switch rather than a hub for performance and control.

[ ] Are there serial hardware reset lines? The reset capability is **mandatory** to ensure data integrity for clusters with only two server-capable nodes, and it is highly recommended for all server-capable nodes. Larger clusters should have an odd number of server-capable nodes, or must have serial hardware reset lines or use I/O fencing with switches if only two of the nodes are server capable. See "Isolating Failed Nodes", page 22, and "Serial Hardware Reset", page 27.

[ ] Read this book and any `README` files provided with the release. If you have clients running operating systems other than IRIX or SGI ProPack for Linux 64-bit, also read the *SGI InfiniteStorage CXFS MultiOS Client-Only Guide*.

[ ] Verify that the network is usable. See Chapter 6, "IRIX CXFS Installation", page 61.

[ ] Install the CXFS software. See Chapter 6, "IRIX CXFS Installation", page 61. If you have clients running operating systems other than IRIX or SGI ProPack for Linux 64-bit, also see the *SGI InfiniteStorage CXFS MultiOS Client-Only Guide*.

[ ] Modify the configuration files as needed. Ensure that the contents of the `/etc/sys_id` file matches the primary name in the `/etc/hosts` file. See "Configuring System Files", page 83.

[ ] Perform other system configuration steps and reboot. See:

- [ ] "Configuring System Files", page 83
- [ ] "IRIX: Configuring for Automatic Restart", page 100
- [ ] "Configuring Network Interfaces", page 89
- [ ] "Configuring the Serial Ports for IRIX Administration Nodes", page 91
- [ ] "Rebooting the System", page 92

[ ] Completely configure a **small** cluster (3 nodes). See Chapter 9, "Initial Configuration of the Cluster", page 103.

[ ] 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 16, "Troubleshooting", page 323.

[ ] Set up the filesystems. See Chapter 9, "Initial Configuration of the Cluster", page 103.

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.

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*.

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. Thus, if two clusters will be sharing the same network for communications, then they must have unique cluster IDs. In the case of multiOS clusters, both the names and IDs must be unique if the clusters share a network.

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 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 that share a network must have unique names and IDs.

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 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.

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", page 38.

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.

A node's control networks should not be set to accept control messages if the node is not a dedicated CXFS node. Otherwise, end users who run other jobs on the machine can have their jobs killed unexpectedly when CXFS resets the node.

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.

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.

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*.

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 *SGI InfiniteStorage FailSafe Administrator's Guide*.

CXFS shutdown

The failure action that stops CXFS kernel-based services on the node in response to a loss of CXFS filesystem membership. The surviving cluster delays the beginning of recovery to allow the node time to complete the 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. The CXFS tiebreaker differs from the FailSafe tiebreaker; see *SGI InfiniteStorage FailSafe Administrator's Guide*.

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 *SGI InfiniteStorage FailSafe Administrator's Guide*.

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.

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.

multiOS

A cluster that is running multiple operating systems, such as IRIX and Solaris.

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 the metadata server moves from one node to another due to an interruption in services on the first node.

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

See *serial hardware reset*.

server-capable administration node

A node that is installed with the `cluster_admin` product and is also capable of coordinating CXFS metadata.

serial hardware reset

The failure action that performs a system reset via a serial line connected to the system controller. This failure action hierarchy choice applies only to nodes with system controllers; see "Requirements", page 33.

shutdown

See *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 serial hardware 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 controller port information is optional for a node in the pool, but is required if the node will be added to a cluster; otherwise resources running on that node never will be highly available.

system log file

Log files in which system messages are stored:

- IRIX: `/var/adm/SYSLOG`
- Linux 64-bit: `/var/log/messages`

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