

CXFS™ Software Installation and Administration Guide

007-4016-011

CONTRIBUTORS

Written by Lori Johnson

Edited by Rick Thompson and Susan Wilkening

Illustrated by Chrystie Danzer and Chris Wengelski

Production by Glen Traefald

Contributions by François Barbou des Places, Ken Beck, Felix Blyakher, Laurie Costello, Dave Ellis, Brian Gaffey, Dean Jansa, Erik Jacobson, Dennis Kender, Chris Kirby, Ted Kline, Dan Knappe, Kent Koeninger, Linda Lait, Bob LaPreze, Steve Lord, Troy McCorkell, LaNet Merrill, Terry Merth, Nate Pearlstein, Alain Renaud, John Relph, Elaine Robinson, Dean Roehrich, Wesley Smith, Kerm Steffenhagen, Paddy Sreenivasan, Andy Tran, Connie Waring, Geoffrey Wehrman

COPYRIGHT

© 1999, 2001 Silicon Graphics, Inc. All rights reserved; provided portions may be copyright in third parties, as indicated elsewhere herein. No permission is granted to copy, distribute, or create derivative works from the contents of this electronic documentation in any manner, in whole or in part, without the prior written permission of Silicon Graphics, Inc.

LIMITED RIGHTS LEGEND

The electronic (software) version of this document was developed at private expense; if acquired under an agreement with the USA government or any contractor thereto, it is acquired as "commercial computer software" subject to the provisions of its applicable license agreement, as specified in (a) 48 CFR 12.212 of the FAR; or, if acquired for Department of Defense units, (b) 48 CFR 227-7202 of the DoD FAR Supplement; or sections succeeding thereto. Contractor/manufacturer is Silicon Graphics, Inc., 1600 Amphitheatre Pkwy 2E, Mountain View, CA 94043-1351.

TRADEMARKS AND ATTRIBUTIONS

Silicon Graphics, IRIS, IRIX, O2, Octane, and Onyx2 are registered trademarks and SGI, CXFS, FailSafe, FDDIXPress, IRIXconsole, IRIS FailSafe, Origin, Trusted IRIX, and XFS are trademarks of Silicon Graphics, Inc.

Adaptec is a trademark of Adaptec, Inc. Brocade is a trademark of Brocade Communication Systems, Inc. Digi is a trademark of Digi International, Inc. FLEXlm is a trademark of GLOBEtrotter, Inc. Java is a registered trademark of Sun Microsystems, Inc. Linux is a registered trademark of Linus Torvalds. Legato NetWorker is a trademark of Legato Systems, Inc. Netscape is a trademark of Netscape Communications Corporation. QLogic is a trademark of QLogic Corporation. Sun is a trademark of Sun Microsystems, Inc. VERITAS is a trademark of VERITAS Software Corporation. Windows is a trademark of Microsoft Corporation.

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

New Features in This Guide

Note: The implementation of relocation and recovery is deferred.

This update contains the following:

- The graphical user interface (GUI) has been improved. The separate cluster view (the `cxdetail` command) and task manager (the `cxtask` command) have been streamlined into one window, the **CXFS Manager**. Both the `cxtask` and `cxdetail` commands are kept for historical purposes; this document refers to just `cxtask` for simplicity.

The new GUI provides the following features:

- Access to tasks through the menu bar or by clicking the right mouse button within the tree view
- Faster filesystem status and cluster status updates
- Access to the `salog(4)` file, which shows every command run from the GUI
- A **Find** textfield helps you find components within the displayed tree-view

See "GUI Overview", page 47.

- Information about the use of `xfstool` and CXFS filesystems; see "Appropriate Use of `xfstool`", page 224.



Caution: Do not use `xfstool` on a CXFS filesystem unless you are certain there is a problem.

- Information about using `cmgr(1M)`:
 - Invoking subcommands directly on the command line with the `-c` option
 - Using template scripts provided in the `/var/cluster/cmgr-templates` directory

See "`cmgr(1M)` Overview", page 51.

- Information about MAC labels in a mixed Trusted IRIX (Trix) and IRIX cluster; see "Hardware and Software Requirements and Support", page 35.
- The structure of the CXFS filesystem configuration was changed with the release of IRIX 6.5.13. Backward compatibility with earlier versions is no longer maintained as of IRIX 6.5.14, because all nodes in the cluster must be running the same or adjacent releases.

If you are upgrading from 6.5.13f entirely to 6.5.14f, there is no further impact.

If you intend to run a mixture of 6.5.13f and 6.5.14f nodes, you must turn off backward compatibility.

If you are upgrading from 6.5.12f or earlier without first installing and running 6.5.13f, then you must perform a one-time manual conversion of your CXFS filesystem definitions.

See "Convert Filesystem Definitions for Upgrades", page 79.

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.

Contents

About This Guide	xxiii
Related Publications	xxiii
Obtaining Publications	xxiv
Conventions	xxiv
Reader Comments	xxv
1. Introduction to CXFS	1
Comparison of XFS and CXFS	1
Supported XFS Features	2
When to Use CXFS	3
When to Use XFS	4
Comparison of Networked and CXFS Filesystems	4
Networked Filesystems	4
CXFS Filesystems	5
Features	5
Restrictions	6
Cluster Environment	7
Terminology	7
Node	7
Cluster Database	7
Pool	8
Cluster	8
Membership	9
Quorum	10
Private Network	10
007-4016-011	vii

Metadata	11
Metadata Server and Metadata Client	11
Relocation	14
Recovery	15
Hardware Components	16
Membership Quorums	19
CXFS Membership Quorum	19
Quorum Calculation and Node Weights	19
Changing Quorum Example	21
Network Partition Example	21
fs2d Database Membership Quorum	23
Hardware Resets	24
Metadata Client/Server Model	25
System View	26
Daemons	27
Communication Paths	29
Flow of Metadata for Reads and Writes	33
Hardware and Software Requirements and Support	35
Coexecution of CXFS and IRIS FailSafe	37
CXFS Resource Type for FailSafe	37
Cluster Type	39
Size of the Cluster	40
Node Types	40
Separate GUIs	40
Conversion	40
Network Interfaces	41
CXFS Tie-Breaker Node	41
Metadata Servers and Failover Domain	41

Communication Paths in a Coexecution Cluster	42
Recovery and Relocation Issues in a Cluster with Only Two Weighted Nodes	43
Cluster Manager Tools	44
CXFS Manager GUI	45
Starting the GUI	45
GUI Overview	47
Viewing the Cluster Components	48
Viewing Component Details	48
Performing Tasks	48
Screens	50
cmgr(1M) Overview	51
Getting Help	52
Using Prompt Mode	52
Completing Actions and Cancelling	53
Using Script Files	53
Invoking a Shell from within cmgr	55
Entering Subcommands on the Command Line	55
Template Scripts	56
2. Installation of CXFS Software and System Preparation	59
Install Software	60
Configure System Files	64
Hostname Resolution: /etc/sys_id, /etc/hosts, /etc/nsswitch.conf	64
/etc/services	67
/etc/config/cad.options	68
/etc/config/fs2d.options	69
Example 1	70
Example 2	71

<i>(Optional)</i> / .rhosts	71
<i>(Optional)</i> /etc/exports	72
<i>(Optional)</i> Configure for Automatic Restart	72
Configure Network Interfaces	73
Configure the Serial Ports	75
Reboot the System	76
Test the System	76
Private Network Interface	76
Serial Reset Connection	77
Convert Filesystem Definitions for Upgrades	79
Upgrading from 6.5.12f or Earlier	79
Running with a Mixture of 6.5.13f and 6.5.14f Nodes	79
Running with All Nodes Upgraded to 6.5.14f	80
3. Initial Configuration of the Cluster	81
Preliminary Steps	81
Verify the License	82
Start the Cluster Daemons	82
Verify that the Cluster Daemons are Running	82
Determine the Hostname of the Node	83
Verify that the chkconfig Flags are On	84
Configuring with the GUI	84
Start the GUI	84
Set Up a New Cluster with the GUI	87
Set Up a New Filesystem with the GUI	88
Configuring with the cmgr(1M) Command	89
4. GUI Reference	101

Guided Configuration Task Sets	101
Set Up an Existing FailSafe Cluster for CXFS with the GUI	102
Make Changes to Existing Cluster	102
Fix or Upgrade Cluster Nodes	103
Node Tasks with the GUI	103
Define a Node with the GUI	104
Examples of Defining a Node with the GUI	108
Add/Remove Nodes in the Cluster with the GUI	109
Reset a Node with the GUI	110
Modify a Node with the GUI	111
Convert a FailSafe Node to CXFS with the GUI	113
Delete a Node with the GUI	114
Display a Node with the GUI	114
Cluster Tasks with the GUI	114
Define a Cluster with the GUI	114
Modify a Cluster Definition with the GUI	116
Convert a FailSafe Cluster to CXFS with the GUI	116
Delete a Cluster with the GUI	117
Display a Cluster with the GUI	117
CXFS Services Tasks with the GUI	117
Start CXFS Services with the GUI	118
Stop CXFS Services (Normal CXFS Shutdown) with the GUI	118
Set Tie-Breaker Node with the GUI	119
Set Log Configuration with the GUI	119
Display Log Group Definitions with the GUI	120
Configure Log Groups with the GUI	120
Filesystem Tasks with the GUI	121
Define a Filesystem with the GUI	122

Mount a Filesystem with the GUI	123
Unmount a Filesystem with the GUI	123
Modify a Filesystem with the GUI	124
Relocate a Metadata Server for a Filesystem with the GUI	125
Delete a Filesystem with the GUI	125
Diagnostic Tasks and Error Recovery with the GUI	125
Test Connectivity with the GUI	126
Revoke Membership of the Local Node with the GUI	126
Allow Membership of the Local Node with the GUI	127
5. cmgr Reference	129
Set Configuration Defaults with cmgr	130
Node Tasks with cmgr	130
Define a Node with cmgr	131
Modify a Node with cmgr	136
Example of Modifying the Node Weight with cmgr	136
Example of Partitioning	137
Reset a Node with cmgr	139
Convert a Node to CXFS or FailSafe with cmgr	139
Delete a Node with cmgr	140
Display a Node with cmgr	142
Cluster Tasks with cmgr	144
Define a Cluster with cmgr	144
Modify a Cluster with cmgr	147
Convert a Cluster to CXFS or FailSafe with cmgr	147
Delete a Cluster with cmgr	148
Display a Cluster with cmgr	150

CXFS Services Tasks with <code>cmgr</code>	150
Start CXFS Services with <code>cmgr</code>	150
Stop CXFS Services with <code>cmgr</code>	151
Set the Tie-Breaker Node with <code>cmgr</code>	151
Set Log Configuration with <code>cmgr</code>	153
Display Log Group Definitions with <code>cmgr</code>	153
Configure Log Groups with <code>cmgr</code>	153
Modify Log Groups with <code>cmgr</code>	154
Filesystem Tasks with <code>cmgr</code>	155
Define a Filesystem with <code>cmgr</code>	155
Mount a Filesystem with <code>cmgr</code>	161
Unmount a Filesystem with <code>cmgr</code>	162
Modify a Filesystem with <code>cmgr</code>	162
Relocate the Metadata Server for a Filesystem with <code>cmgr</code>	166
Delete a Filesystem with <code>cmgr</code>	167
Diagnostic and Error Recovery Tasks with <code>cmgr</code>	167
Test Network Connectivity with <code>cmgr</code>	167
Testing the Serial Connections with <code>cmgr</code>	168
Revoke Membership of the Local Node with <code>cmgr</code>	168
Allow Membership of the Local Node with <code>cmgr</code>	168
Script Example	169
6. Administration and Maintenance	173
Executing Scripts Around Mount Operations	174
Example <code>cxfs-pre-mount</code>	175
Example <code>cxfs-post-mount</code>	175
Example <code>cxfs-pre-umount</code>	176

Example <code>cxfs-post-umount</code>	176
Using <code>fsr(1M)</code>	176
Using <code>find(1)</code> and <code>crontab(1)</code>	176
Using Hierarchical Storage Management (HSM) Products	177
Discovering the Metadata Server for a Filesystem	177
Metadata Server Discovery with the GUI	177
Metadata Server Discovery with <code>cluster_status</code>	179
Metadata Server Discovery with <code>clconf_info</code>	179
Metadata Server Recovery	179
Shutdown of the Database and CXFS	180
Cluster Configuration Database Shutdown	181
Node Status and Cluster Configuration Database Shutdown	181
If the Node is the Metadata Server	182
Restart the Cluster Configuration Database	182
Normal CXFS Shutdown	182
Node Status and Normal CXFS Shutdown	183
When You Should Not Perform a Normal CXFS Shutdown	183
Rejoining the Cluster after a Normal CXFS Shutdown	184
Forced CXFS Shutdown: Revoke Membership of Local Node	184
Node Status and Forced CXFS Shutdown	185
Rejoining the Cluster after a Forced CXFS Shutdown	185
Reset Capability and Forced CXFS Shutdown	186
Avoiding a Restart of the Database at Reboot	186
Log File Management	186
Rotating All Log Files	186
Rotating Large Log Files	187
Volume Management	187

Disk Management	188
Backups	188
NFS	188
Quotas	188
SAMBA	188
Filesystem Maintenance	189
Mounting Filesystems	189
Unmounting Filesystems	190
Growing Filesystems	191
Dump and Restore	191
Cluster Database Backup and Restore	192
chkconfig Flags	193
System Tunable Parameters	194
7. Monitoring Status	195
Log Files	195
Cluster Status	196
Key to Icons and Colors	196
Check Cluster Status with the GUI	198
Check Cluster Status with <code>cluster_status</code>	198
Check Cluster Status with <code>clconf_info</code>	199
Check Cluster Status with <code>cmgr</code>	200
Node Status	200
Monitoring Node Status with the GUI	201
Querying Node Status with <code>cmgr</code>	202
Monitoring Node Status with <code>cluster_status</code>	203
Pinging the System Controller with <code>cmgr</code>	203
Monitoring Reset Serial Line with <code>cmgr</code>	203

XVM Statistics	204
8. Troubleshooting	207
Troubleshooting Strategy	207
Know the Tools	207
Physical Storage Tools	208
Cluster Configuration Tools	208
Cluster Control Tools	209
Networking Tools	210
Cluster/Node Status Tools	211
Performance Monitoring Tools	212
Kernel Status Tools	212
Log Files	213
Identify the Cluster Status	214
Locate the Problem	215
Avoiding Problems	217
Proper Start Up	218
Eliminating a Residual Cluster	218
fs2d Membership Quorum Stability	219
Consistency in Configuration	219
Weight Nodes Appropriately	219
GUI Use	220
Log File Names and Sizes	220
Netscape and the Brocade Switch GUI	220
Performance Problems with Unwritten Extent Tracking and Exclusive Write Tokens	220
Unwritten Extent Tracking	221
Exclusive Write Tokens	221
Avoiding Excessive Filesystem Activity Caused by the crontab File	222

Using System Capacity Wisely	222
Reboot Before Changing Node ID or Cluster ID	223
Remove Unused Nodes	223
Restarting CXFS after a Forced Shutdown	223
Removing Reset Lines	224
Appropriate Use of <code>xfstool repair</code>	224
Common Problems	225
Cannot Access Filesystem	225
GUI Will Not Run	226
Log Files Consume Too Much Disk Space	226
Unable to Define a Node	226
System is Hung	227
Node is Detected but Never Joins Membership	227
Cell ID Count and “Membership Delivered” Messages	227
You Cannot Log In	228
Understanding Error Messages	228
Normal Messages	228
<code>clconfd</code> Daemon Death	230
Out of Logical Swap Space	231
No Cluster Name ID Error	231
Lost CXFS Membership	232
License Error	232
IP Address Error	232
SYSLOG Errors	233
<code>cli</code> Errors	234
<code>clconfd</code> Errors	235
<code>crsd</code> Errors	238
<code>cmond</code> Errors	239

Contents

fs2d Errors	240
General Messages	241
Log File Errors	242
cad Messages	243
cli Messages	245
crsd Errors	246
fs2d Errors	247
Corrective Actions	247
Restarting CXFS Services	248
Clearing the Cluster Configuration Database	248
Rebooting	249
Recovering a Two-Node Cluster	249
Rebooting without Rejoining the Cluster	251
Stopping and Restarting Cluster Infrastructure Daemons	252
Recreating the Cluster Configuration Database	252
Appendix A. Initial Configuration Checklist	253
Glossary	255
Index	263

Figures

Figure 1-1	Pool and Cluster Concepts	9
Figure 1-2	Evenly Distributed Metadata Servers	12
Figure 1-3	Multiple Metadata Servers	13
Figure 1-4	One Metadata Server	14
Figure 1-5	Relocation versus Recovery	15
Figure 1-6	Example of a Four-Node Cluster Using Reset	17
Figure 1-7	Example of a Two-Node Cluster Using Reset	18
Figure 1-8	Changes in Quorum and Active Metadata Server due to Network Partitioning	22
Figure 1-9	Minimum CXFS and <code>fs2d</code> Database Membership Concepts	23
Figure 1-10	Administrative Communication within One Node	29
Figure 1-11	Daemon Communication within One Node	30
Figure 1-12	Communication between Nodes in the Pool	31
Figure 1-13	Communication for a Node Not in a Cluster	32
Figure 1-14	Metadata Flow on a Write	33
Figure 1-15	Metadata Flow on a Read on Client B Following a Write on Client A	34
Figure 1-16	Metadata Flow on a Read on Client B Following a Read on Client A	35
Figure 1-17	Administrative Communication within One Node under Coexecution	42
Figure 1-18	Daemon Communication within One Node under Coexecution	43
Figure 1-19	Initial CXFS Manager Window	50
Figure 1-20	GUI Showing Details for a Node	51
Figure 3-1	CXFS Manager	86
Figure 4-1	Example Node Definition	108
Figure 4-2	Example System Controller Settings	109

Figure 6-1	Window Showing the Metadata Server	178
Figure 7-1	Node Status	202
Figure 7-2	pmgxvm chart	205

Tables

Table 1-1	CXFS Daemons and Threads	27
Table 2-1	fs2d.options File Options	69
Table 5-1	System Controller Types	133
Table 6-1	System Tunable Parameters	194
Table 7-1	Key to Icons	197
Table 7-2	Key to Colors	197
Table 8-1	SYSLOG Error Message Format	233
Table 8-2	Log Error Message Format	243

About This Guide

This publication documents CXFS running on a storage area network (SAN). It assumes that you are already familiar with the XFS file system documented in *IRIX Admin: Disks and Filesystems* and that you have access to the *XVM Volume Manager Administrator's Guide*.

You should read through this entire book, especially Chapter 8, "Troubleshooting", page 207, before attempting to install and configure a CXFS cluster.

Related Publications

The following documents contain additional information:

- *EL Serial Port Server Installation Guide* (provided by Digi International)
- *EL Serial Port Server Installation Guide Errata*
- *FDDIXPress Administration Guide*
- *IRISconsole Administrator's Guide*
- *IRIS FailSafe Version 2 Administrator's Guide*
- *IRIX 6.5 Installation Instructions*
- *IRIX Admin: Disks and Filesystems*
- *NIS Administrator's Guide*
- *Performance Co-Pilot User's and Administrator's Guide*
- *Performance Co-Pilot Programmer's Guide*
- *Personal System Administration Guide*
- *SGI TP 9400 RAID Owner's Guide*
- *SGI TP9400 RAID Administration Guide*
- *SGI Total Performance 9100 Storage System Owner's Guide*
- *XVM Volume Manager Administrator's Guide*

The following man pages are provided with CXFS:

- `cdbBackup(1M)`
- `cdbRestore(1M)`
- `cmond(1M)`
- `cluster_mgr(1M)`
- `cluster_status(1M)`
- `crsd(1M)`

Obtaining Publications

To obtain SGI documentation, go to the SGI Technical Publications Library at:

<http://techpubs.sgi.com>.

Conventions

The following conventions are used throughout this document:

Convention	Meaning
<code>command</code>	This fixed-space font denotes literal items such as commands, files, routines, path names, signals, messages, and programming language structures.
<i>variable</i>	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 *FailSafe* as an abbreviation for *IRIS FailSafe*.

Reader Comments

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

You can contact us in any of the following ways:

- Send e-mail to the following address:

`techpubs@sgi.com`

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

`http://techpubs.sgi.com`

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

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

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

We value your comments and will respond to them promptly.

Introduction to CXFS

CXFS allows groups of computers to coherently share large amounts of data while maintaining high performance. It runs on storage area network (SAN) disks, such as Fibre Channel. CXFS and IRIS FailSafe share the same infrastructure.

This introduction discusses the following:

- "Comparison of XFS and CXFS"
- "Comparison of Networked and CXFS Filesystems", page 4
- "Cluster Environment", page 7
- "Flow of Metadata for Reads and Writes", page 33
- "Hardware and Software Requirements and Support", page 35
- "Coexecution of CXFS and IRIS FailSafe", page 37
- "Recovery and Relocation Issues in a Cluster with Only Two Weighted Nodes", page 43
- "Cluster Manager Tools", page 44

Note: You should read through this entire book, especially Chapter 8, "Troubleshooting", page 207, before attempting to install and configure a CXFS cluster.

Comparison of XFS and CXFS

CXFS is based on SGI's high-performance XFS filesystem. CXFS uses the same filesystem structure as XFS. A CXFS filesystem is initially created using the same `mkfs(1M)` 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(1M)` command directly, by the system during boot via an entry in `/etc/fstab`, or by way of the Filesystem Manager.
 - A filesystem resides on only one system.
 - The `/etc/fstab` file contains static information about filesystems.
- In CXFS:
 - Filesystem are mounted using the `cmgr(1M)` command or the graphical user interface (GUI). CXFS filesystems are mounted across the cluster by CXFS management software.
 - A filesystem resides on all systems (nodes) in a cluster. A *metadata server* coordinates 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.
 - Information is **not** stored in the `/etc/fstab` file. (However, the CXFS filesystems do show up in the `/etc/mtab` file.)

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 bandwidths, high transaction rates, and fast metadata operations).
- Dynamically allocated metadata space.
- Quotas. (The quota mount options must be the same on all mounts of the filesystem. You can administer quotas anywhere in the cluster just as if this were a regular XFS filesystem.)
- Filesystem reorganizer (defragmenter), which must be run from the CXFS metadata server. See the `fsr_xfs(1M)` man page.

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/Os 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.

For more information about XFS features, see *IRIX Admin: Disks and Filesystems*.

When to Use CXFS

CXFS performs best under the following conditions:

- Data I/O operations are greater than 16 KB
- Reads from and writes to a file that is opened by only one process
- Reads from and writes to a file where all processes with that file open reside on the same host
- Reads from a file where multiple processes on multiple hosts read the same file
- Reads from and writes to a file using direct-access I/O for multiple processes on multiple hosts
- Large files and file accesses are being used

For most filesystem loads, the scenarios above represent the bulk of the file accesses. Thus, CXFS tends to deliver 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.

When to Use XFS

Some operations can be slower in CXFS than in local XFS filesystems. For example, metadata operations can take longer to complete through the metadata server 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 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.

Applications that need frequent and high-performance shared, distributed reads and writes can use direct-access I/O in place of buffered I/O for fast local-file performance. For example, distributed parallel databases would tend to use direct-access I/O.

Comparison of Networked and CXFS Filesystems

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

Networked 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* (I/O per second) and the *bandwidth* (megabytes per second). These delays are difficult to avoid in the client-server architecture of LAN-based networked 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. For example, 10- to 1000-Mbit Ethernet have limits of 1 to 100 MB per second, respectively. This is 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 file server can be a bottleneck for performance when multiple clients wait their turns for data, which must pass through the centralized file server. For example, network file system (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 system. 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 networked filesystems, but with significant performance and functionality advantages. CXFS runs on top of a storage area network (SAN), where each computer system 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 computer systems in the cluster. Each system can read and write the disks at near-local disk speeds; the data passes directly from the disks to the system requesting the I/O, without passing through a data server or over a local area network (LAN). For the data path, each system 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 IRIX systems, the bandwidth can scale to gigabytes and even tens of gigabytes per second. Compare this with a networked 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 system 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 file system and is not dependent on any particular host. The path name is a normal POSIX path name; 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 symmetric multiprocessor (SMP) host and processes distributed across multiple hosts have the same view of the filesystem, with performance similar on each host.

This differs from typical networked filesystems, which tend to include the name of the fileserver in the path name. 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 IRIX 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.

The following XFS features are not supported in CXFS:

- Real-time filesystems
- Guaranteed-rate I/O
- Swap to a file
- Nesting of mount points; that is, you cannot mount a filesystem on top of a CXFS filesystem

Cluster Environment

This section discusses the following:

- "Terminology"
- "Hardware Components", page 16
- "Membership Quorums", page 19
- "Hardware Resets", page 24
- "Metadata Client/Server Model", page 25
- "System View", page 26
- "Daemons", page 27
- "Communication Paths", page 29

Terminology

This section defines the terminology necessary to understand CXFS. Also see the Glossary, page 255.

Node

A *node* is an operating system (OS) image, usually an individual computer. All nodes must be running adjacent levels of the IRIX OS; for example, 6.5.13f and 6.5.14f (this applies as of 6.5.12f). The nodes are connected to a storage area network (SAN) that connects the storage systems to the nodes in the cluster. A node can belong to only one cluster.

This use of the term *node* does not have the same meaning as a node in an Origin system.

Cluster Database

The *cluster database* contains configuration information about all nodes and the cluster. The `fs2d` daemon manages the distribution of the cluster configuration database (CDB) across the nodes in the pool.

Pool

The *pool* is the entire set of nodes that are coupled to each other by networks and are defined as nodes in the cluster database. The nodes are usually close together and should always serve a common purpose. A replicated cluster configuration database is stored on each node in the pool.

All nodes that can be added to a cluster are part of the pool, but not all nodes in the pool must be part of the cluster. There is only one pool. (Other pools may exist, but each is disjoint from the other. They share no node or cluster definitions.)

Cluster

The *cluster* is the set of nodes in the pool that have been defined as a cluster. The cluster is identified by a simple name; this name must be unique within the pool. (For example, you cannot use the same name for the cluster and for a node.)

All nodes in the cluster are also in the pool. However, all nodes in the pool are not necessarily in the cluster; that is, the cluster may consist of a subset of the nodes in the pool. There is only one cluster per pool.

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

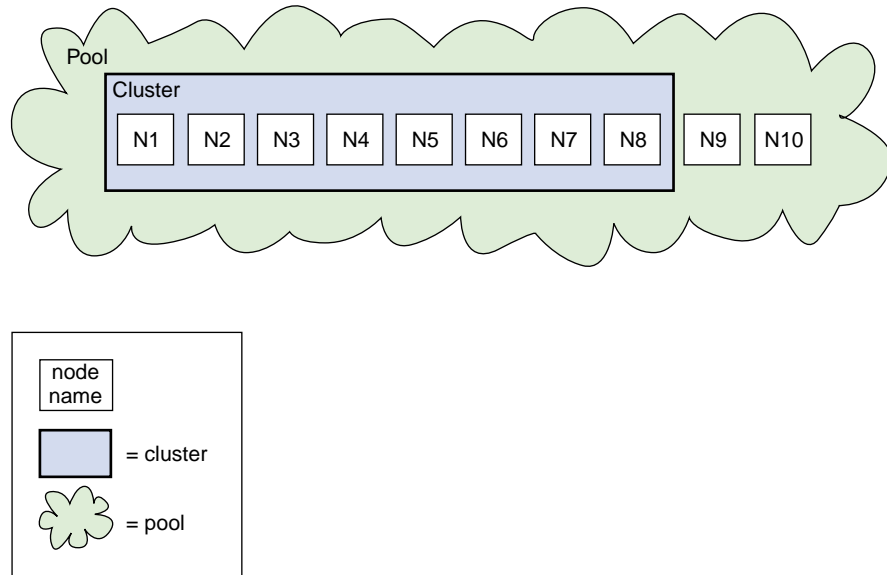


Figure 1-1 Pool and Cluster Concepts

Membership

There are the following types membership:

- *CXFS membership* (also known as *kernel-space membership*) is the group of CXFS nodes in the **cluster** that can actively share filesystems in the cluster; this may be a subset of the nodes defined in a cluster. During the boot process, a node applies for CXFS membership. Once accepted, the node can actively share the filesystems of the cluster.
- *fs2d database membership* (also known as *user-space membership*) is the group of nodes in the **pool** that are accessible to fs2d. The fs2d daemon manages the distribution of the cluster database across the nodes in the pool; nodes available to fs2d are able to receive cluster configuration database updates, and are therefore part of the fs2d database membership; this may be a subset of the nodes defined in the pool.

CXFS membership and fs2d database membership differ from *FailSafe membership*. For more information about FailSafe, see *IRIS FailSafe Version 2 Administrator's Guide*.

Quorum

The *quorum* is the number of nodes required to form a cluster, which differs according to membership:

- For CXFS membership:
 - A majority (>50%) of the weighted nodes in the cluster are required to **form** an initial membership
 - Half (50%) of the weighted nodes in the cluster are required to **maintain** an existing membership
- For fs2d database membership, 50% of the **nodes in the pool** are required to form and maintain a cluster. (No weighting is used.)

For more complete information, see "Membership Quorums", page 19.

Private Network

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

- The cluster software uses the private network to send the heartbeat/control messages necessary for the cluster configuration to function. If there are delays in receiving heartbeat messages, the cluster software may determine that a node is not responding and have its CXFS membership revoked, causing it to either be reset or disconnected, depending upon the configuration.
- Rebooting network equipment can cause the nodes in a cluster to lose communication; the cluster will move into a degraded state 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, because the messaging protocol does not prevent snooping or spoofing, a network with restricted access is safer than one with user access.

Therefore, because the performance and security characteristics of a public network could cause problems in the cluster and because heartbeat is very timing-dependent (even small variations can cause problems), **a private network is required**.

In addition, SGI recommends that all nodes be on the same local network segment.

Note: 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 "Network Partition Example", page 21.

Metadata

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

Metadata Server and Metadata Client

The *metadata server* coordinates the metadata for a filesystem on behalf of the other nodes in the cluster; the other nodes are known as *metadata clients*. Data, as opposed to metadata, moves directly between the hosts and disks (peer-to-disk).

Note: Do not confuse *metadata server* and *metadata client* with the traditional data-path client/server model used by networked 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 metadata client via the fibre channel connection.

Each filesystem may have a list of potential metadata servers, but there is only one **active** metadata server per filesystem. There can be multiple active metadata servers in the cluster (one per filesystem). The following figures show different possibilities.

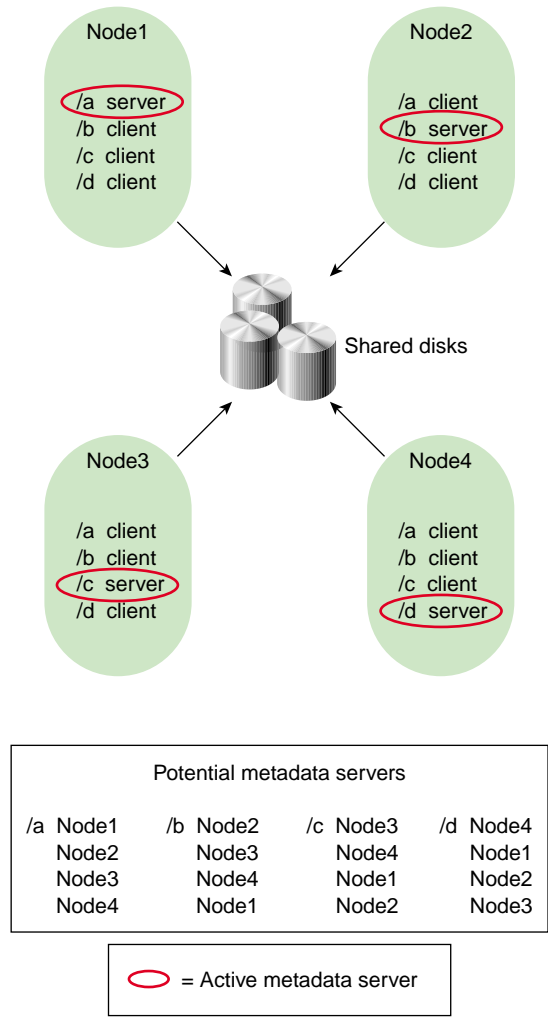


Figure 1-2 Evenly Distributed Metadata Servers

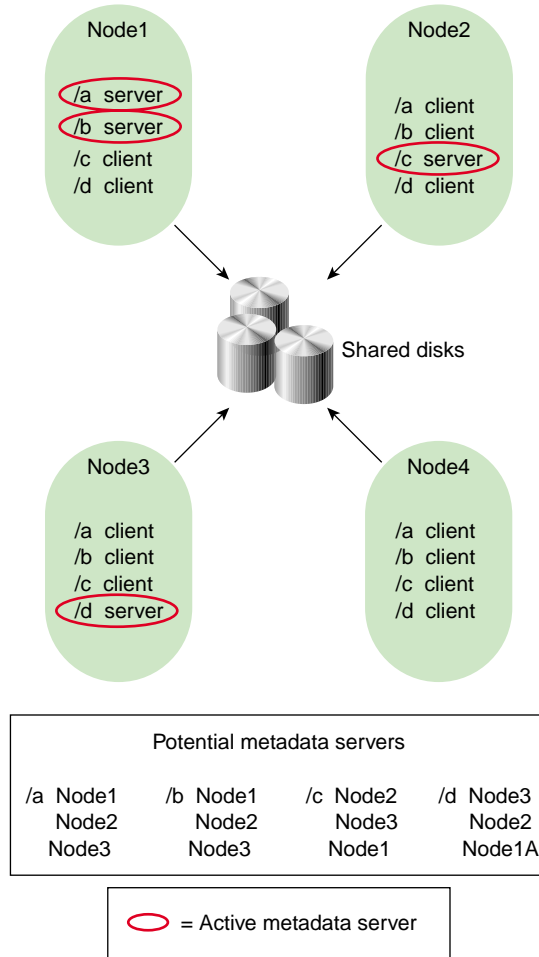


Figure 1-3 Multiple Metadata Servers

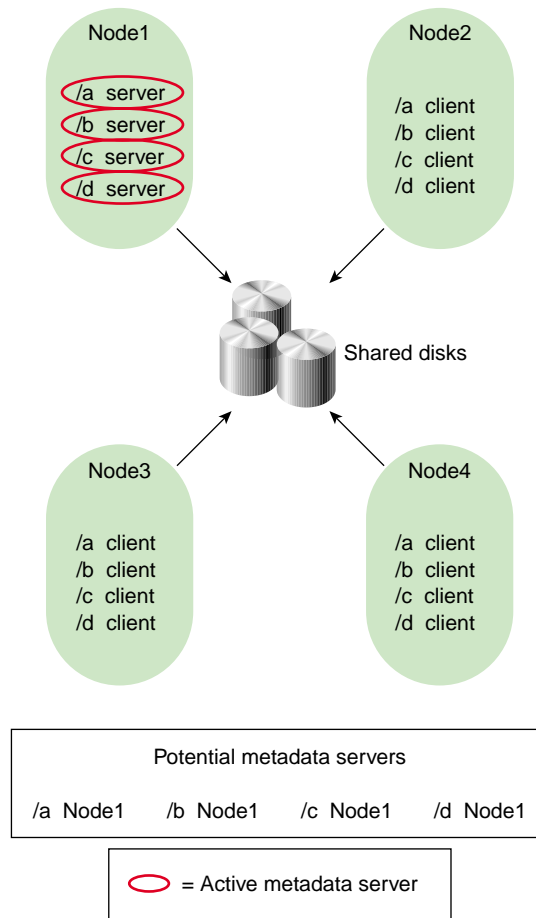


Figure 1-4 One Metadata Server

Relocation

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

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

Recovery

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

Figure 1-5 describes the difference between relocation and recovery.

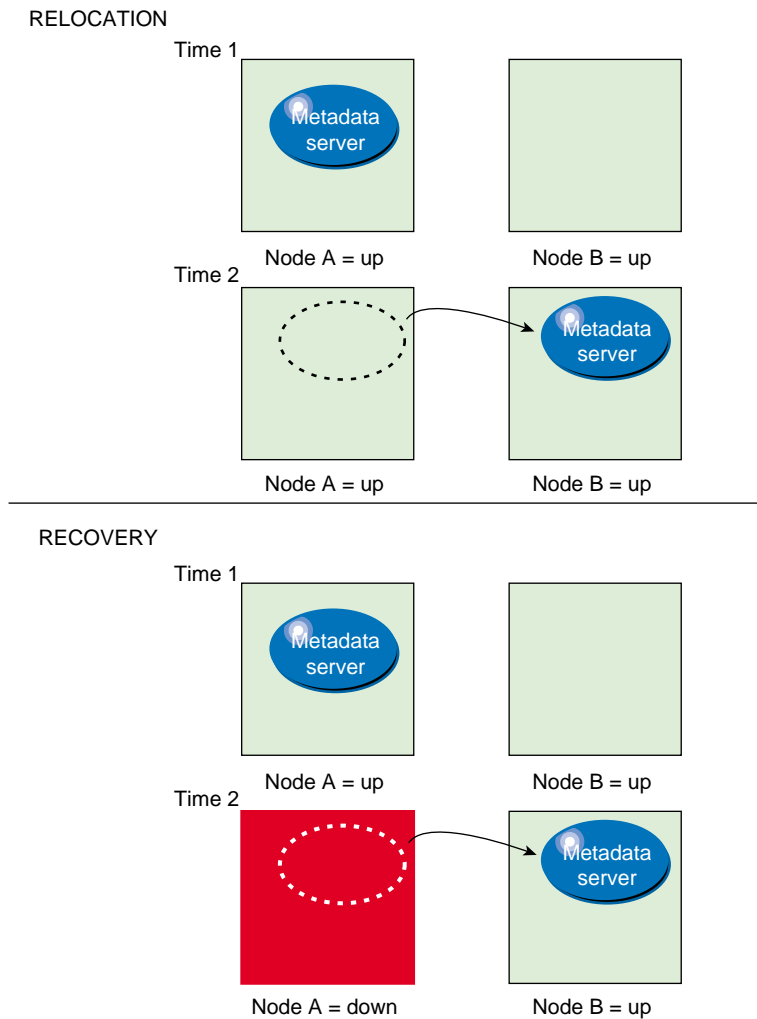


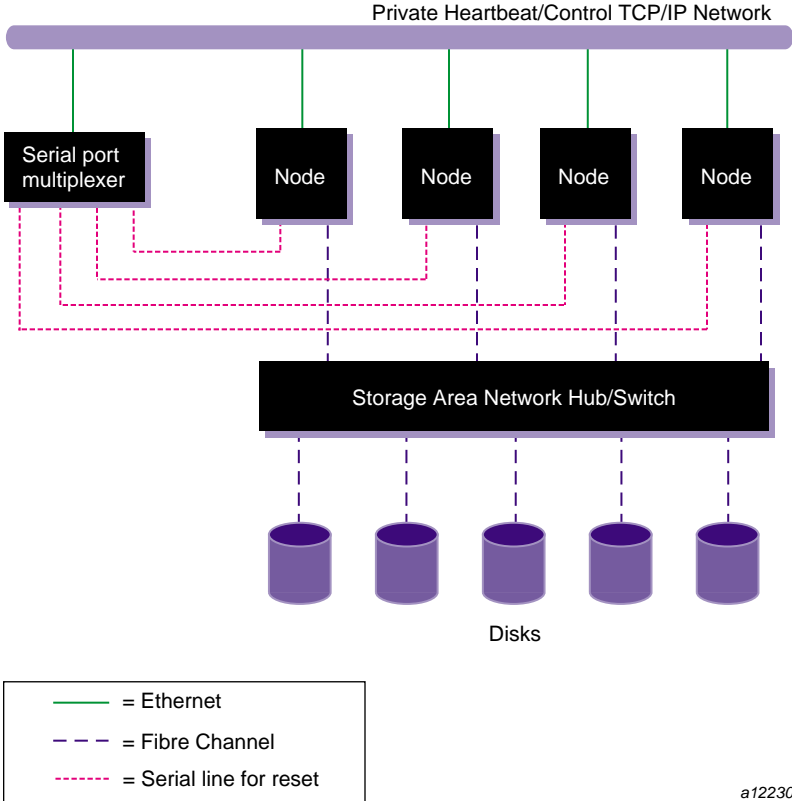
Figure 1-5 Relocation versus Recovery

Hardware Components

Figure 1-6 shows an example of the CXFS hardware components for a four-node cluster using the reset capability and an Ethernet serial port multiplexer. SGI recommends a switch rather than a hub for performance and control. (The user network is not shown.)

Note: The reset capability is **highly** recommended to ensure data integrity, especially for clusters with only two weighted nodes; reset is required for IRIS FailSafe. (See "Recovery and Relocation Issues in a Cluster with Only Two Weighted Nodes", page 43.)

A SCSI multiplexer can be used in place of the Ethernet multiplexer with a similar configuration. The reset connection has the same connection configuration as IRIS FailSafe; for more information, contact SGI professional or managed services.



a12230

Figure 1-6 Example of a Four-Node Cluster Using Reset

Figure 1-7 shows a cluster with two weighted nodes using direct serial lines for reset. (The user network is not shown.)

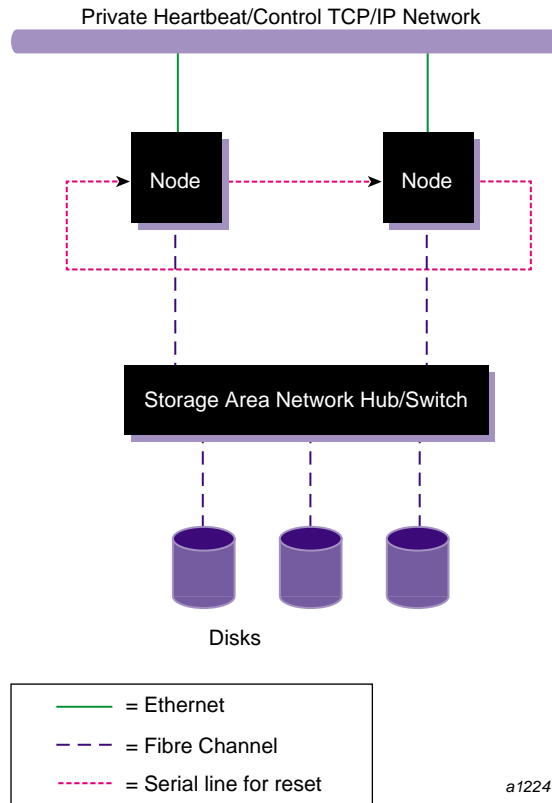


Figure 1-7 Example of a Two-Node Cluster Using Reset

With CXFS, each node has direct access to the logical volumes on the SAN, which enhances performance. Each node is more tightly coupled to the logical volumes than in a network file system (NFS) solution. To ensure data integrity, each filesystem has a metadata server; see "Metadata Client/Server Model", page 25. CXFS software provides the infrastructure that allows the metadata server to decide CXFS membership and access questions.

Membership Quorums

There are separate quorum mechanisms for each membership type:

- "CXFS Membership Quorum"
- "fs2d Database Membership Quorum", page 23

CXFS Membership Quorum

By design, there can only be one active metadata server per filesystem. However, a problem can develop if there are problems in the heartbeat/control network, which is used to transport metadata information between clients and the metadata server. If the heartbeat/control network is somehow split in half (for example, due to a network failure), the network can become two smaller networks (segments). If this happens, the CXFS membership quorum ensures that only one metadata server is writing the metadata portion of the CXFS filesystem over the storage area network.

Quorum Calculation and Node Weights

The **CXFS membership quorum** is calculated based on the combined weight of nodes attempting to participate in the CXFS membership compared to the total weight of all nodes defined in the cluster.

Nodes have a default weight of 1. When all nodes have a weight of 1, an initial CXFS membership quorum calculation essentially becomes a majority (>50%) of the number of nodes.

In cases where a cluster consists of one or more CXFS metadata servers and multiple CXFS client-only nodes, you may want to configure the node weights such that the quorum consists of only the possible metadata servers, regardless of the state of the clients. In this case, you should define a weight of 1 for the metadata servers and 0 for the clients. If clients are weighted 0, the clients will be unable to form a CXFS membership by themselves. In this scenario, a CXFS membership will always require a quorum of the possible metadata servers.

Note: At least one node must have a membership weight greater than 0. All possible metadata servers must have a membership weight greater than 0.

Membership weight values other than 0 or 1 are not recommended.

For the initial CXFS membership quorum, a majority (>50%) of the weighted nodes must be available to bring up a cluster; a large cluster in which all nodes are weighted 1 will require more nodes to be available before a cluster can be formed than if most of the nodes are weighted 0. To maintain the existing CXFS membership quorum requires half (50%) of the weighted nodes defined in the cluster.

If you do not use hardware reset, you should set a *CXFS tie-breaker node* to avoid multiple clusters (also known as *split-brain syndrome*) in the event of a network partition.

Note: The reset capability is **highly** recommended to ensure data integrity, especially for clusters with only two weighted nodes; reset is required for IRIS FailSafe.

A CXFS tie-breaker node is a node that is identified for CXFS to use in the process of computing CXFS 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 tie-breaker.

However, if the CXFS tie-breaker node in a cluster with only two weighted nodes fails or if the administrator stops CXFS services, the other node will do a forced shutdown, which unmounts all CXFS filesystems.

The CXFS tie-breaker node must have weight greater than 0 so that it can be a metadata server.

If the network being used for heartbeat/control is divided in half, only the portion of the network that has the quorum of metadata servers and the CXFS tie-breaker node will remain in the cluster. Nodes on any portion of the heartbeat/control network that is not part of the quorum will shut down from the cluster. 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.

The CXFS membership monitors itself with normal messaging and heartbeating. If a failure occurs, the offending nodes are removed from the CXFS membership or are reset to prevent further access to shared resources by these nodes. A node that discovers it has been eliminated from the CXFS membership (due to a communication failure) will forcibly terminate access to the shared disks and stop CXFS services, if it is not first reset.

The number of nodes possible in the CXFS membership can be changed to either expand the cluster to include new nodes or to remove nodes that have left the CXFS

membership. Removal of a node that is down and will remain unavailable reduces the weight required to form a quorum if the node has a weight of 1.

Changing Quorum Example

The following is an example of a changing CXFS membership quorum.

Consider a pool of 6 nodes (A, B, C, D, E, and F) on one private network, each node having a weight of 1:

- C is a CXFS tie-breaker node
- A, B, C, D, and E are members of the CXFS cluster

Given this, the minimum number of nodes needed for an initial CXFS membership quorum is three nodes (>50%).

If B were to shut down or leave the CXFS membership, then the remaining number of nodes in the cluster are A, C, D and E. The cluster would still be available in this case because the cluster still satisfies the requirements to maintain the CXFS membership quorum (50%).

Network Partition Example

Continuing the previous example, Figure 1-8 displays a situation in which a router dies and the heartbeat/control network is effectively split in two. The nodes on network segment 2 (nodes D and E) will disconnect because they do not contain the CXFS tie-breaker node, and therefore do not have a quorum. On network segment 1, one of the other two non-active metadata servers will become active and the cluster will only include the systems on network segment 1. Even after the router is repaired, the nodes that were on network segment 2 will remain disconnected until cluster services are restarted on them.

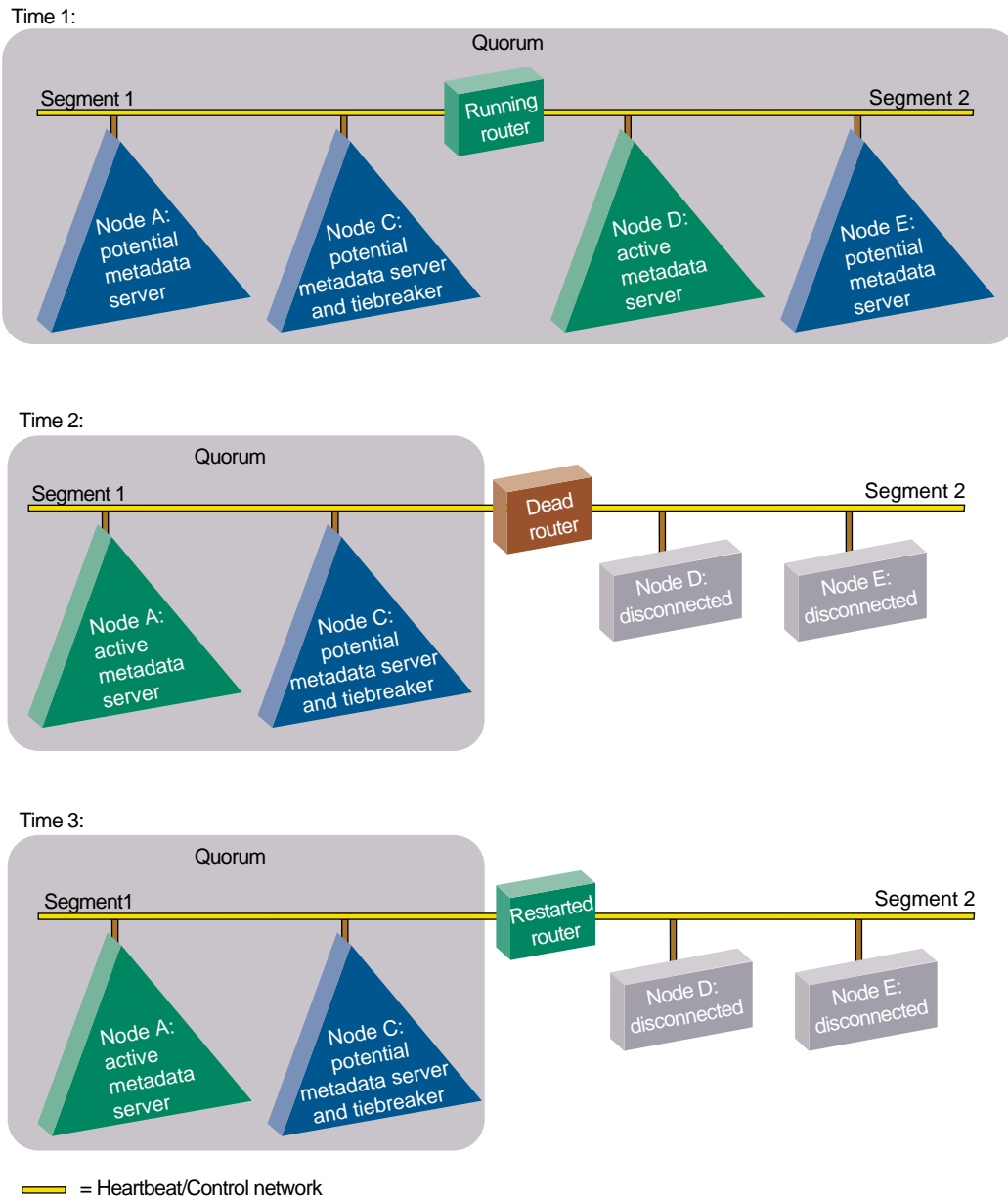


Figure 1-8 Changes in Quorum and Active Metadata Server due to Network Partitioning

fs2d Database Membership Quorum

The **fs2d database membership** quorum allows an initial cluster to be formed when **half** (50%) of the nodes in the **pool** are available to the fs2d daemon (and can therefore receive cluster configuration database updates). Weighting is not used when calculating the fs2d database membership quorum.

Note: This differs from the **CXFS membership**, which requires that a majority (>50%) of the weighted nodes in the cluster are available before an initial cluster can start. The CXFS membership can be maintained with only 50% of the weighted nodes.

Figure 1-9 shows the concepts of minimum CXFS and fs2d database memberships.

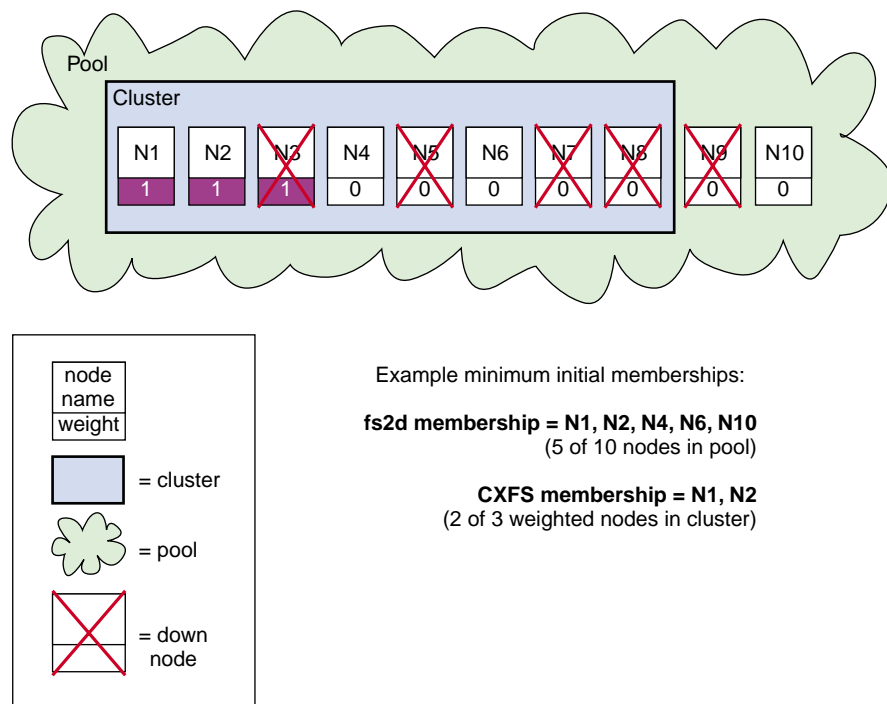


Figure 1-9 Minimum CXFS and fs2d Database Membership Concepts

Hardware Resets

Nodes that detect that they have lost contact with the cluster will forcibly terminate access to shared disks. In the absence of 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 remote reset, especially for clusters with an even number of weighted nodes, particularly for a cluster of only two weighted nodes. All possible metadata servers must be weighted, and weighted nodes should have reset lines. Reset is required for IRIS 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 reset hardware 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 membership splits into two halves (each with half the total weight), the following will happen:

- If the CXFS tie-breaker and reset are configured, the half with the tie-breaker node will reset the other half. The side without the tie-breaker will attempt to forcibly shut down CXFS services.
- If there is no CXFS tie-breaker node but reset is configured, each half will attempt to reset the other half using a delay heuristic. One half will succeed and continue. The other will lose the reset race and be rebooted.
- If there is no CXFS tie-breaker node and reset is not configured, then both sides will delay, each assuming that one will win the race and reset the other. Both sides

will then continue running because neither will have been reset, leading to likely data corruption.

To avoid this situation, you should always have at least the tie-breaker node or reset capability configured. However, if the tie-breaker node (in a cluster with only two weighted 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 membership, it will have only half the weight and be unable to form an initial CXFS membership, preventing two CXFS memberships in a single cluster.

The remote reset connections take the following forms:

- Clusters of two nodes can be directly connected with serial lines.
- Clusters of three or more nodes should be connected with an Ethernet multiplexer or SCSI 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.

Metadata Client/Server Model

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

- Metadata logging
- File locking
- Buffer coherency

All CXFS requests for metadata are routed over a TCP/IP network and through a 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 in the cluster can be defined as a potential metadata server, as long as it has weight.
- A single node in the cluster can act as the metadata server for multiple filesystems at once.
- There can be multiple nodes acting as metadata servers, each with different sets of filesystems. However, a given filesystem has a single active metadata server on a single node.
- Although you can configure multiple 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 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 179.)
- 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 "Executing Scripts Around Mount Operations", page 174.

For more information, see "Flow of Metadata for Reads and Writes", page 33.

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 path names 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 file server. It combines the speed of near-local disk access with the flexibility, scalability, and reliability of clustering.

Daemons

Table 1-1 lists the CXFS daemons and threads.

Note: CXFS shares with XFS the `xfsd` kernel threads to push buffered writes to disk.

If you are using a coexecution (of type CXFS and FailSafe) cluster, see the *IRIS FailSafe Version 2 Administrator's Guide*, for information about FailSafe daemons.

Table 1-1 CXFS Daemons and Threads

Layer	Subsystem	Process	Description
Cluster services (CXFS)	<code>cluster_services</code>	<code>clconfd</code>	CXFS cluster configuration daemon. Reads the cluster configuration from the CDB database and manages the local kernel's CXFS membership services accordingly.
Cluster software infrastructure (cluster administrative processes)	<code>cluster_admin</code>	<code>cad</code>	Cluster administration daemon. Provides administration services.
	<code>cluster_control</code>	<code>crsd</code>	Node control daemon. Monitors the serial connection to other nodes. Has the ability to reset other nodes.
		<code>cmond</code>	Daemon that manages all other daemons. This process starts other processes in all nodes in the cluster and restarts them on failures.
		<code>fs2d</code>	Manages the database and keeps each copy in synchronization on all nodes in the pool.

Layer	Subsystem	Process	Description
Kernel Threads	stthreads	cmsd	Manages CXFS 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.
		dcshake	Purges idle CXFS vnodes on the CXFS client.
	xthreads	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.
		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.

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.

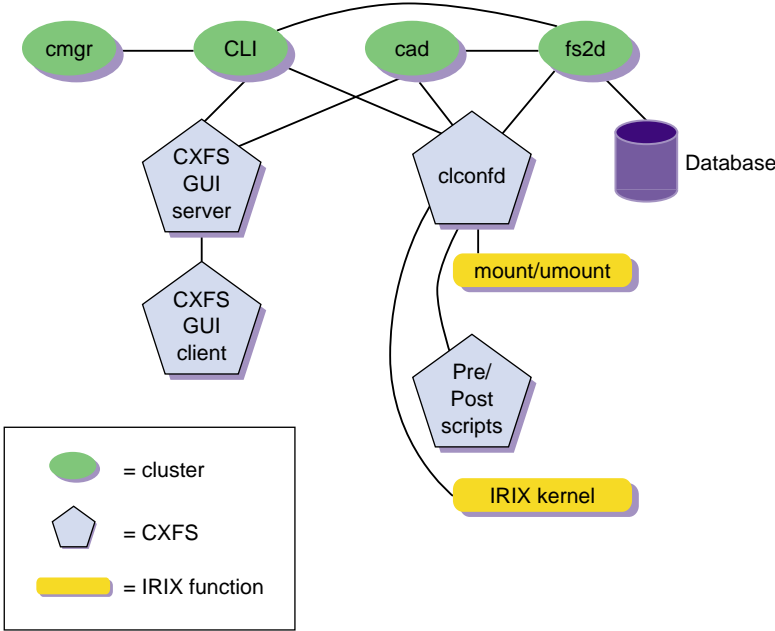


Figure 1-10 Administrative Communication within One Node

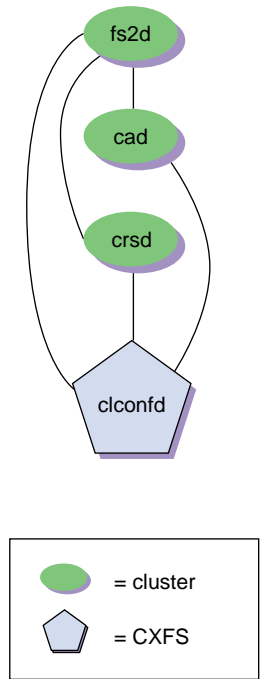


Figure 1-11 Daemon Communication within One Node

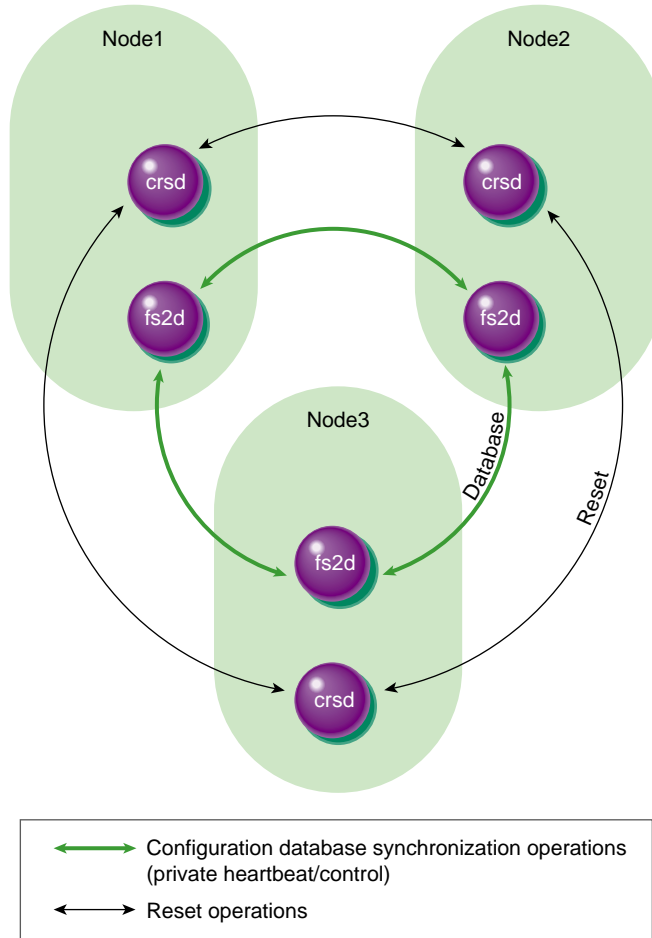


Figure 1-12 Communication between Nodes in the Pool

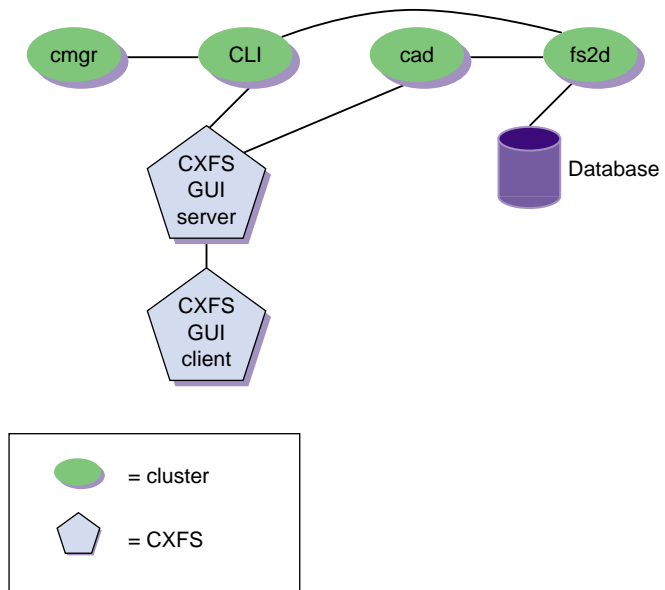


Figure 1-13 Communication for a Node Not in a Cluster

Flow of Metadata for Reads and Writes

The following figures show examples of metadata flow.

Note: A token protects a file. There can be multiple read tokens for a file at any given time, but only one write token.

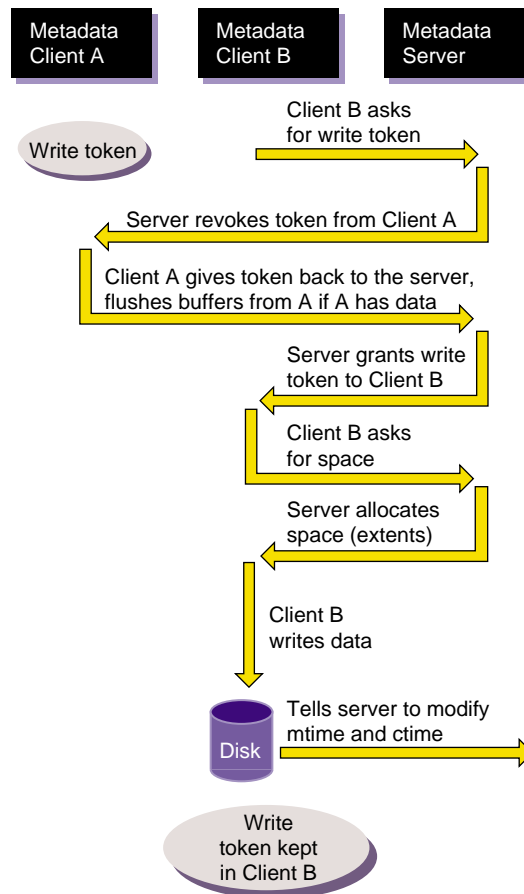


Figure 1-14 Metadata Flow on a Write

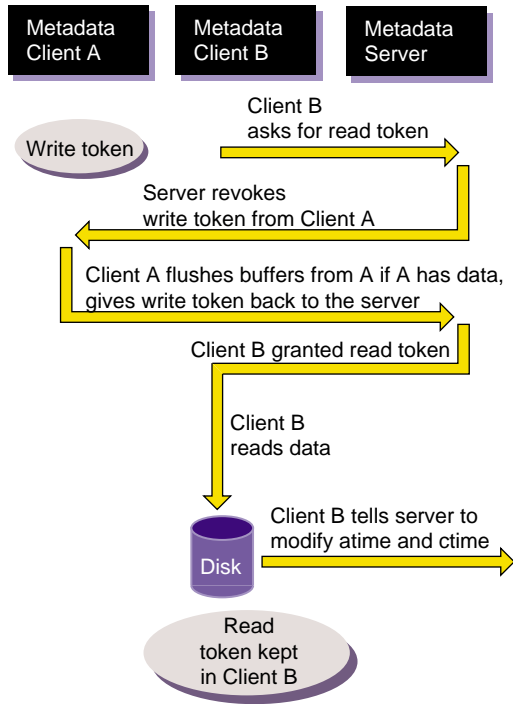


Figure 1-15 Metadata Flow on a Read on Client B Following a Write on Client A

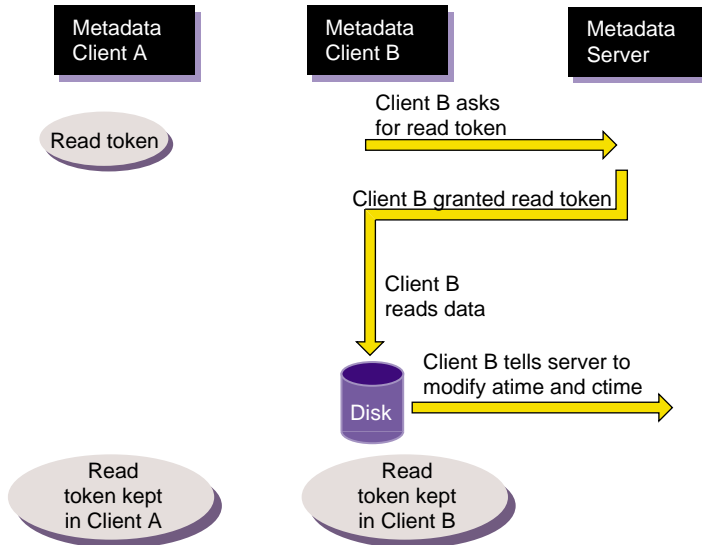


Figure 1-16 Metadata Flow on a Read on Client B Following a Read on Client A

Hardware and Software Requirements and Support

CXFS requires the following:

- A CXFS-only cluster is supported with as many as 16 nodes. The CXFS architecture is designed to support up to 64 nodes; a future release will support this number.
A cluster in which both CXFS and IRIS FailSafe 2.1 or later are run (known as *coexecution*) is supported with a maximum of 16 nodes, as many as 8 of which can be FailSafe; see "Coexecution of CXFS and IRIS FailSafe", page 37, for further configuration details.
- There is only one pool and one cluster.
- All nodes must be running the same or adjacent levels of the IRIX operating system (OS); for example, 6.5.12f and 6.5.13f (this applies as of 6.5.12f).
- FLEXlm license key for CXFS.
- A private TCP/IP network connected to each CXFS metadata server and client.
- Origin 3000, Origin 2000, Origin 200, Onyx2, or Onyx3 platforms.

- A supported SAN hardware configuration other than a Silicon Graphics O2 workstation. This includes the following platforms:
 - Origin 2000
 - Origin 200
 - Octane
- XVM volume manager, which is provided as part of the IRIX release.

CXFS works with the following:

- Trusted IRIX/CMW. CXFS has now been qualified in an SGI Trix cluster with the Data Migration Facility (DMF) and Tape Management Facility (TMF). If you want to run CXFS and Trusted IRIX, SGI recommends that 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.

In a mixed Trusted-IRIX/IRIX cluster, an IRIX client will require but not have a mandatory access control (MAC) label associated with its credentials when it attempts access a Trix 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 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: `msen low, mint high`.
- IRIS FailSafe. See "Coexecution of CXFS and IRIS FailSafe", page 37, and *IRIS FailSafe Version 2 Administrator's Guide*.
- IRISconsole; see *IRISconsole Administrator's Guide*.
- Ethernet multiplexer or SCSI multiplexer (used for the reset capability in clusters of three or more nodes).

Coexecution of CXFS and IRIS FailSafe

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, which is known as *coexecution*. This allows you to have application-level high availability and a clustered filesystem.

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(1M)` 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 consist of the CXFS metadata server and the metadata 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(1M)` 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 *IRIS FailSafe Version 2 Administrator's Guide*.

Cluster Type

The cluster can be one of three types:

- FailSafe. In this case, all nodes will also be of type FailSafe.
- CXFS. In this case, all nodes will be of type CXFS.
- CXFS and FailSafe (coexecution). In this case, all nodes will be a mix of type CXFS and type CXFS and FailSafe, 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.

Size of the Cluster

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 can be configured with a minimum of three weighted nodes (CXFS weight) and a maximum of 16 nodes. (A cluster with reset cables and only two weighted nodes is supported, but there are inherent issues with this configuration; see "Recovery and Relocation Issues in a Cluster with Only Two Weighted Nodes", page 43.) All the nodes in the cluster must run CXFS. As many as 8 nodes can also run IRIS FailSafe.

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(1M)` (`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(1M)`, 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 102.

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 139, and "Convert a Cluster to CXFS or FailSafe with `cmgr`", page 147.

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

Do not use a CXFS tie-breaker 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).

Communication Paths in a Coexecution Cluster

The following figures show the communication paths within one node in a coexecution cluster.

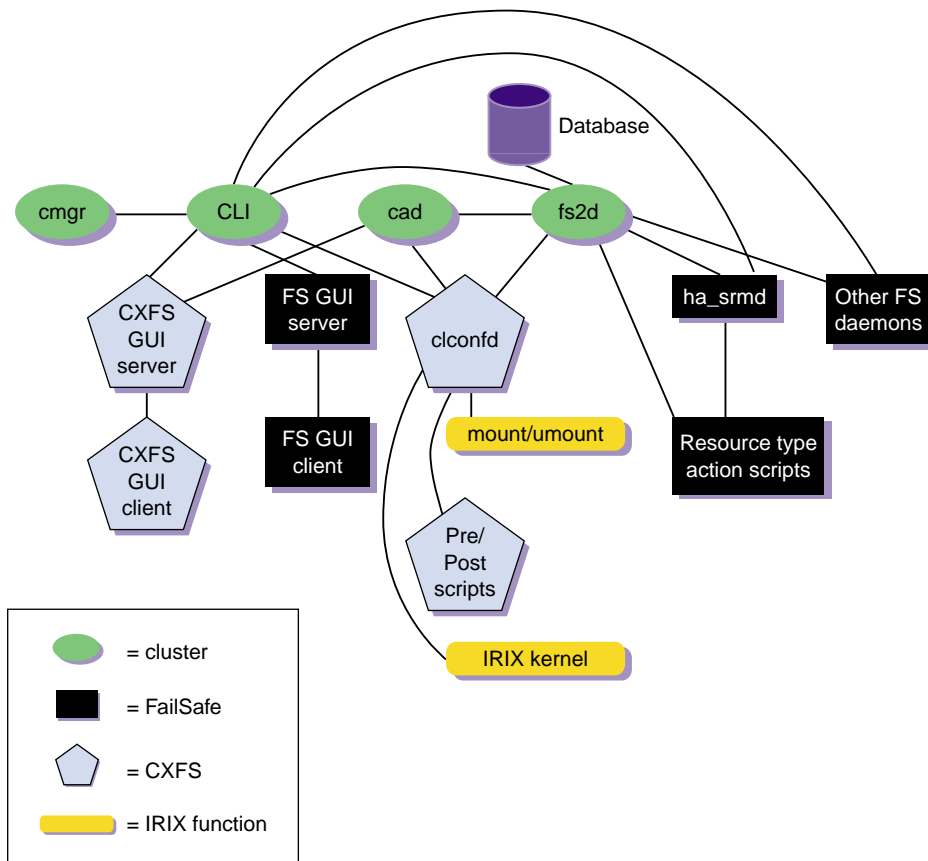


Figure 1-17 Administrative Communication within One Node under Coexecution

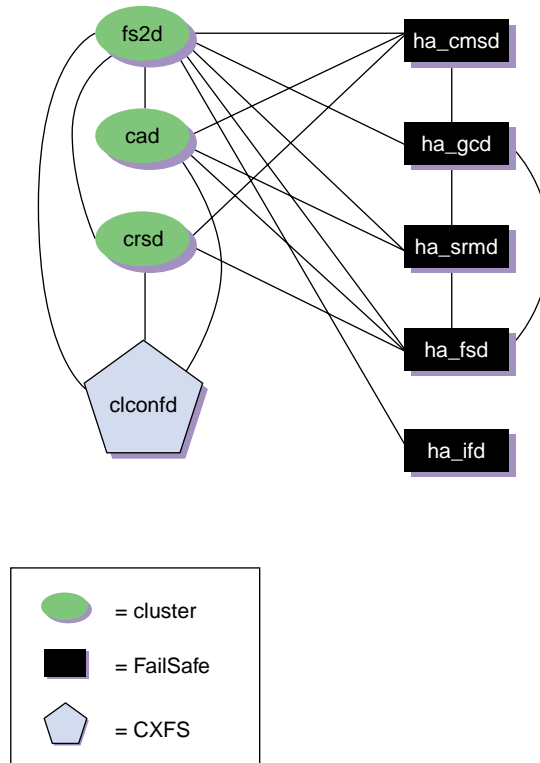


Figure 1-18 Daemon Communication within One Node under Coexecution

Recovery and Relocation Issues in a Cluster with Only Two Weighted Nodes

A cluster of at least three weighted nodes is recommended for a production environment (that is, one requiring relocation and recovery of the metadata server).

If you use a production cluster with an even number of weighted nodes (especially only two weighted nodes), you must do one of the following:

- Use reset lines to ensure protection of data and guarantee that only one node is running in error conditions (reset lines are recommended for all CXFS clusters and required for use with IRIS FailSafe).

- Weight one node as 1 and the other as 0.
- Set a CXFS tie-breaker node.

However, even with these methods, there are recovery and relocation issues inherent to a cluster with only two weighted nodes.

Suppose you have a cluster with reset lines and only two weighted nodes, and one of the nodes has a problem. The following situations may occur:

- Both nodes are weighted 1, no CXFS tie-breaker:

The client node will survive a server panic or reset and might survive a loss of connection with the metadata server on the private network. If there is a connection loss, both the client and the metadata server will attempt to reset the other node. The node that succeeds first will survive.

- Both nodes are weighted 1, metadata server is the CXFS tie-breaker:

The client node should survive a failure of the metadata server because the **existing** CXFS membership quorum can be maintained with only 50% of the weight present.

- Both nodes are weighted 1, client is the CXFS tie-breaker:

For both a server panic or reset and a loss of connection with the metadata server on the private network, the client node will survive. If the client was listed as a secondary metadata server, it will become the active metadata server. If it was not listed, then the file system will be unmounted.

Note: Usually, you want the primary metadata server to be the CXFS tie-breaker.

For more information, see "Hardware Components", page 16, and "Quorum Calculation and Node Weights", page 19.

Cluster Manager Tools

You can perform CXFS configuration tasks using either the CXFS Manager graphical user interface (GUI) or the `cmgr(1M)` cluster manager command (also known as `cluster_mgr`). 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 and task sets 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 machine, or any computer that has a Java-enabled Web browser, including Windows and Linux computers and laptops.

The `cmgr(1M)` command is more limited in its functions. It enables you to configure and administer a cluster system only on an IRIX system. 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.

CXFS Manager GUI

This section provides an overview of the CXFS Manager graphical user interface (GUI).

Starting the GUI

When CXFS daemons have been started, you must be sure to connect to a node that is running all of the CXFS daemons to obtain the correct cluster status. When CXFS daemons have not yet been started in a cluster, you can connect to any node in the pool.

The node from which you run the GUI affects your view of the cluster. You should wait for a change to appear in the tree view before making another change; the change is not guaranteed to be propagated across the cluster until it appears in the tree view. The entire cluster status information is sent each time a change is made to the cluster database; therefore, the larger the configuration, the longer it will take.

To ensure that the required privileges are available for performing all of the tasks, you should log in to the GUI as `root`. However, some or all privileges can be granted to any other user by the system administrator using the Privilege Manager, part of the IRIX Interactive Desktop System Administration (`sysadmdesktop`) product. For more information, see the *Personal System Administration Guide*.

Note: You should only make changes from one instance of the GUI running at any given time; changes made by a second GUI instance (a second invocation of `cxtask`) may overwrite changes made by the first instance. However, multiple windows accessed via the **File** menu are all part of a single GUI instance; you can make changes from any of these windows.

To start the GUI, use one of the following methods:

- Enter the following command line:

```
# /usr/sbin/cxtask
```

If you invoke `cxtask` before you have defined a cluster, you will get an error message. If you are in the process of defining a cluster, you can ignore it.

The `cxdetail` command performs the identical function as `cxtask`; both commands are kept for historical purposes.

- Choose the following from the Toolchest:

```
CXFS  
> CXFS Manager
```

Note: 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 &
```

In order for this to take effect, `sysadm_cxfs.sw.desktop` must be installed on the client system.

A dialog box will appear prompting you to log in to a host.

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

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

You can use this method of launching the GUI if you want to run it from a non-IRIX system. If you are running the GUI on an IRIX system, the preferred method is to use the Toolchest or the `/usr/sbin/cxtask` command.

GUI Overview

The **CXFS Manager** GUI allows you to administer the entire CXFS cluster from a single point. It provides access to the tasks and task sets that help you set up and administer your CXFS cluster:

- *Tasks* let you set up and monitor individual components of a CXFS cluster.
- *Guided configuration task sets* 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.

The **File** menu lets you display multiple windows for this instance of the GUI, the `/var/adm/SYSLOG` system log file, and the `/var/sysadm/salog` system administration log file (which shows the commands accessed by the GUI). It also lets you close the current window and exit the GUI completely.

The **Edit** menu lets you expand and collapse the contents of the tree view. You can also choose to automatically expand the display to reflect new nodes added to the pool or cluster.

The **Tasks** menu contains the following:

- **Find Tasks**, which lets you use keywords to search for a specific task
- **Guided Configuration**, which contains the task sets to set up your cluster, define filesystems, modify an existing cluster, and check status
- **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 tie-breaker node, and set the log configuration
- **Filesystems**, which contains tasks to define and manage filesystems and relocate a metadata server.

Note: Relocation is deferred in this release.

- **Diagnostics**, which contains the task to test the cluster and nodes for configuration problems
- **Error Recovery**, which contains the tasks that allow you to recover from CXFS membership errors

By default, the window is divided into two sections: the *tree view* and the *item view*. You can use the arrows the the middle of the window to shift the display. To find a component in the current tree, enter the name of the component and click the **Find** button.

To deselect an item in the tree view, click anywhere in the tree view except on the name of an item.

Viewing the Cluster Components

Choose what you want to view from the **View** selection: the nodes in the cluster, the nodes in the pool (all defined nodes), or filesystems in the cluster.

Viewing Component Details

To view the details on any component, click on its icon name in the tree view. The configuration and status details will appear in the item view to the right. To see the details about an item in the item view, 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 tree view. For example:

Task

- > **Guided Configuration**
- > **Set Up a New Cluster**

The task or task set window appears.

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.

Screens

Figure 1-19 shows the **CXFS Manager** window. Figure 1-20 shows details in the item view.

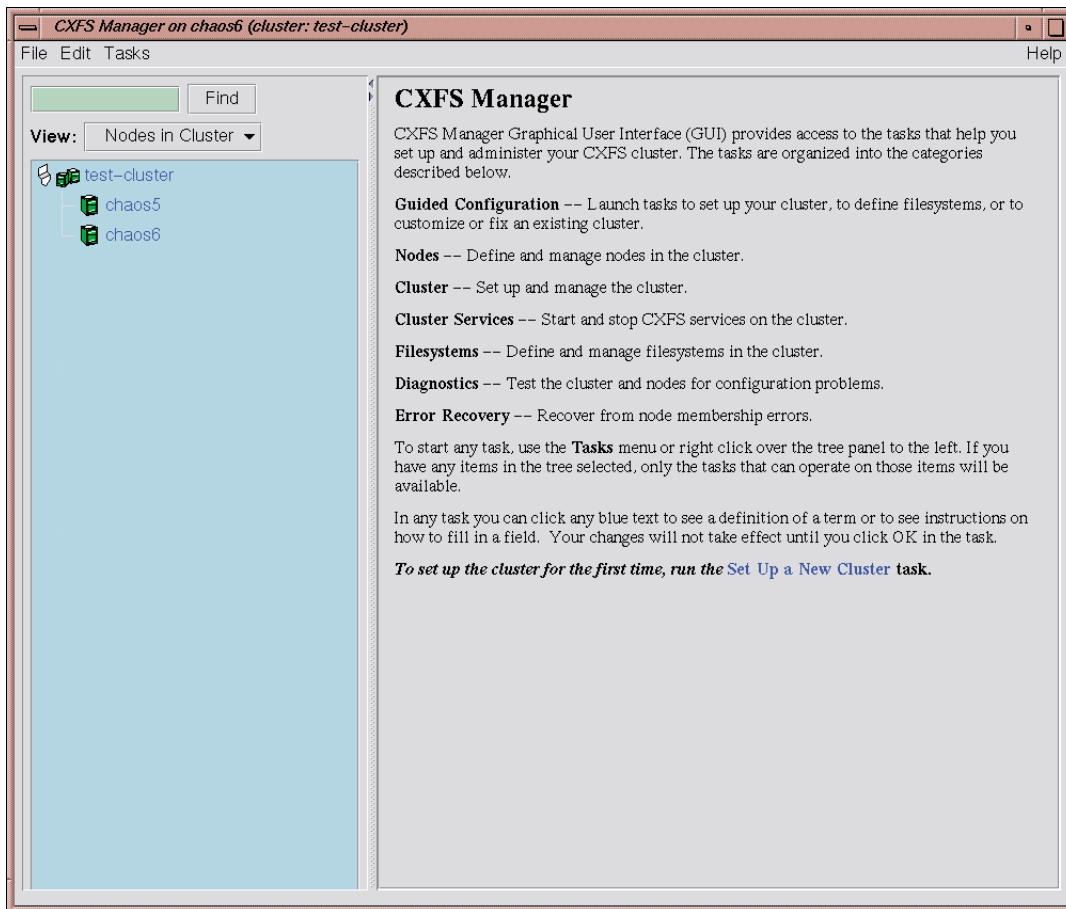


Figure 1-19 Initial CXFS Manager Window

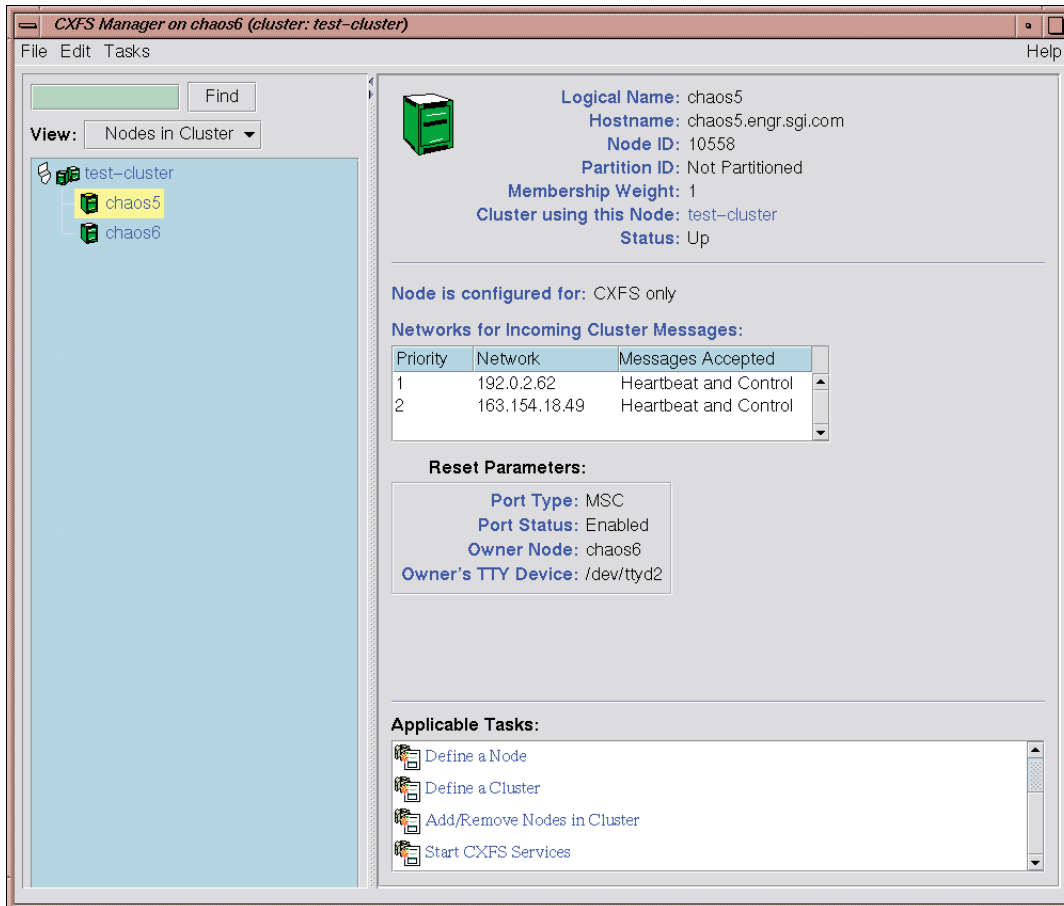


Figure 1-20 GUI Showing Details for a Node

cmgr(1M) Overview

To use the `cmgr(1M)` command, you must be logged in as `root`. Then enter either of the following:

```
# /usr/cluster/bin/cluster_mgr
```

or

```
# /usr/cluster/bin/cmgr
```

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>
```

For a complete discussion of the available `cmgr` commands, see Chapter 5, "cmgr Reference", page 129, and the `cmgr(1M)` man page.

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(1M)` 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 asked in prompting mode:

```
cmgr> define node nodename
```

```
Enter commands, you may enter "done" or "cancel" at any time to exit
```

```
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> ?
```

```

Node ID ? [optional]
Partition ID ? [optional] (0)
Do you wish to define system controller info[y/n]:
Reset type <powerCycle> ? (powerCycle)
Do you wish to define system controller info[y/n]:
Sysctrl Type <msc|mmsc|l2>? (msc)
Sysctrl Password[optional] ? ( )
Sysctrl Status <enabled|disabled> ?
Sysctrl Owner ?
Sysctrl Device ?
Sysctrl Owner Type <tty> ? (tty)
Number of Network Interfaces ? (1)
NIC 1 - IP Address ?
NIC 1 - Heartbeat HB (use network for heartbeats) <true|false> ?
NIC 1 - (use network for control messages) <true|false> ?
NIC 1 - Priority <1,2,...> ?
Node Weight ? (1)

```

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 commits 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 169.

Invoking a Shell from within `cmgr`

To invoke a shell from within `cmgr(1M)`, 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:

```
cluster_mgr -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 Chapter 5, "cmgr Reference", page 129, and the `cmgr(1M)` 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.

Installation of CXFS Software and System Preparation



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.

This chapter covers the following steps:

1. "Install Software", page 60
2. "Configure System Files", page 64
3. "(Optional) Configure for Automatic Restart", page 72
4. "Configure Network Interfaces", page 73
5. "Configure the Serial Ports", page 75
6. "Reboot the System", page 76
7. "Convert Filesystem Definitions for Upgrades", page 79

After completing these steps, see Chapter 3, "Initial Configuration of the Cluster", page 81.

You should read through this entire book, especially Chapter 8, "Troubleshooting", page 207, before attempting to install and configure a CXFS cluster. If you are using coexecution with IRIS FailSafe, see *IRIS FailSafe Version 2 Administrator's Guide*.

Install Software

Installing the CXFS base CD requires approximately 30.3 MB of space. To install the required software, do the following:

1. On each node in the pool, upgrade to IRIX 6.5.14f 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** 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:

```
CXFS not properly licensed for this host. Run
      '/usr/cluster/bin/cxfslicense -d'
for detailed failure information.
```

If you increase the number of CPUs in your system, you may need a new license. However, repartitioning an existing Origin 3000 system does not require a new license.

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

- b. Insert CD-ROM #2 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(1M)` 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 CXFS software:

```
Inst> install cluster*
Inst> install cxfs*
Inst> install sysadm_base*
Inst> install sysadm_cxfs*
```

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
cxfs.books.CXFS_AG
cxfs.sw.cxfs
cxfs.sw.xvm_cell
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
```

When `sysadm_base` is installed, `tcpmux` service is enabled.

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:

```
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 `sysadm_base.sw.client`, `sysadm_cxfs.sw.client`, and `sysadm_cxfs.sw.web` subsystems must be installed on the pool nodes that you will connect to (by means of a Java-enabled Web browser running on any platform) for performing administrative operations. These subsystems are part of the default software that was installed in step 3e.

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 the workstation from which you want to perform CXFS administration (the *administrative workstation*, which can be a node in the cluster or outside of the cluster) runs the GUI client from an IRIX desktop, install the following subsystems:

```
Inst> keep *
Inst> install java_eoe.sw (SGI version 3.1.1)
Inst> install java_eoe.sw32
Inst> install sysadm_base.man
Inst> install sysadm_base.sw.client
Inst> install sysadm_cxfs.man
Inst> install sysadm_cxfs.sw.client
Inst> install sysadm_cxfs.sw.desktop
Inst> go
```



Caution: The GUI only operates with Java 1.1.8. This is the version of Java that is provided with the IRIX 6.5.xf release.

The SGI Web site also contains Java 2. However, you **cannot** use this version of Java with the GUI. Using a Java version other than 1.1.8 will cause the GUI to fail.

6. If the administrative workstation is an IRIX machine that launches the GUI client from a Web browser that supports Java, install the `java_plugin` from the CXFS CD. (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.)

If you try to install all subsystems in `java_plugin`, `inst` reports incompatible subsystems (`java_plugin.sw.swing101`, `java_plugin.sw.swing102`, and `java_plugin.sw.swing103`). Do not install these three subsystems because the GUI does not use them.

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

For a non-IRIX workstation, download the Java Plug-in from the above URL. If the Java plug-in is not installed when the GUI is run from a browser, the browser is redirected to this site.

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

8. 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 "Reboot the System", page 76.

Configure 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.

This section describes the required and optional changes you must make.

Hostname Resolution: `/etc/sys_id`, `/etc/hosts`, `/etc/nsswitch.conf`



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

The following hostname resolution rules and recommendations apply to CXFS clusters:

- Hostnames cannot begin with an underscore (`_`) or include any whitespace characters.
- The value of the `/etc/sys_id` file must match the node's primary hostname in the `/etc/hosts` file (that is, the first field after the node's IP address in `/etc/hosts`). This field can be either the hostname or the fully qualified domain name.

You must use the name in `/etc/sys_id` at the following times:

- In the **Server** field in the GUI login window when logging in to the CXFS GUI for the first time
- In the **Hostname** field when defining the first node in the pool

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
```

SGI recommends that you provide two networks; although only the primary network is used by CXFS for heartbeat/control, the reset daemon can fail over to the second network if necessary. The reset daemon forwards a reset request to the appropriate daemon on the failing node and accesses the reset hardware if it is accessible via the network.

For example, suppose your `/etc/hosts` contains the following:

```
# The public interface:
128.2.3.4 color-green.sgi.com color-green green

# The private interface:
192.0.1.1 color-green-private.sgi.com color-green-private green-private
```

The `/etc/sys_id` file could contain either the hostname `color-green` or the fully qualified domain name `color-green.sgi.com`. It cannot contain the alias `green`.

In this case, you would enter the hostname `color-green` or the fully qualified domain name `color-green.sgi.com` for the **Server** field in the login screen and for the **Hostname** field in the **Define a new node** window.

Note: Using the value of `/etc/sys_id` is only necessary when logging in to the **first** node during initial configuration; aliases such as `green` may be used for subsequent nodes.

- If you use the `nsd(1M)` name service daemon, you must configure your system so that local files are accessed before either the network information service (NIS) or

the domain name service (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.)

The `/etc/config/netif.options` file must have one of the interfaces be equal to the value of `/etc/sys_id` (`$HOSTNAME`).

For more information about the Unified Name Service (UNS) and the name service daemon, see the `nsd(1M)` man page.

- If you change the `/etc/nsswitch.conf` or `/etc/hosts` files, you must restart `nsd` by using the `nsadmin restart` command, which also flushes its cache.

The reason you must restart `nsd(1M)` after making a change to these files is that the `nsd` name service daemon actually takes the contents of `/etc/hosts` and places the contents in its memory cache in a format that is faster to search. Thus, you must restart `nsd` in order to see that change and place the new `/etc/hosts` information into RAM cache. If `/etc/nsswitch.conf` is changed, `nsd` must re-read this file so that it knows what type of files (for example, `hosts` or `passwd`) to manage, what services it should call to get information, and in what order those services should be called.

The IP addresses on a running node in the cluster and the IP address of the first node in the cluster cannot be changed while cluster services are active.

- You should be consistent when using fully qualified domain names in the `/etc/hosts` file. If you use fully qualified domain names in `/etc/sys_id` 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 host 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.

- If you change hostname resolution settings in the `/etc/nsswitch.conf` file after you have defined the first node (which creates the cluster database), you must recreate the database. See "Recreating the Cluster Configuration Database", page 252.

- When using coexecution with IRIS FailSafe, never add an `/etc/hosts` entry that associates the value of `/etc/sys_id` with an IP address alias. You must use the primary address.

`/etc/services`

Edit the `/etc/services` file on each node so that it contains entries for `sgi-cad` and `sgi-crsd` before you install the `cluster_admin` product on each 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
```

/etc/config/cad.options

The `/etc/config/cad.options` file contains the list of parameters that the cluster administration daemon reads when the `cad` process is started. `cad` provides cluster information.

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

<code>--append_log</code>	Append <code>cad</code> logging information to the <code>cad</code> log file instead of overwriting it.
<code>--log_file filename</code>	<code>cad</code> log filename. Alternately, this can be specified as <code>-lf filename</code> .
<code>-vvvv</code>	Verbosity level. The number of <code>v</code> characters indicates the level of logging. Setting <code>-v</code> logs the fewest messages; setting <code>-vvvv</code> 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(1M)` 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(1M)` daemon; the GUI will prompt you to reconnect.

For information about licensing, see "Install Software", page 60.

/etc/config/fs2d.options

The `/etc/config/fs2d.options` file on each 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 configuration database (CDB) across the nodes in the pool.

Table 2-1 shows the options can that can be set in the `fs2d.options` file.

Table 2-1 `fs2d.options` File Options

Option	Description
<code>-logevents <i>event name</i></code>	Log selected events. The following event names may be used: <code>all</code> , <code>internal</code> , <code>args</code> , <code>attach</code> , <code>chandle</code> , <code>node</code> , <code>tree</code> , <code>lock</code> , <code>datacon</code> , <code>trap</code> , <code>notify</code> , <code>access</code> , <code>storage</code> . The default is <code>all</code> .
<code>-logdest <i>log destination</i></code>	Set log destination. The following log destinations may be used: <code>all</code> , <code>stdout</code> , <code>stderr</code> , <code>syslog</code> , <code>logfile</code> . If multiple destinations are specified, the log messages are written to all of them. If <code>logfile</code> is specified, it has no effect unless the <code>-logfile</code> option is also specified. The default is <code>logfile</code> .
<code>-logfile <i>filename</i></code>	Set log filename. The default is <code>/var/cluster/ha/log/fs2d_log</code> .
<code>-logfilemax <i>maximum size</i></code>	Set log file maximum size (in bytes). If the file exceeds the maximum size, any preexisting <code>filename.old</code> will be deleted, the current file will be renamed to <code>filename.old</code> , and a new file will be created. A single message will not be split across files. If <code>-logfile</code> is set, the default is 10000000.
<code>-loglevel <i>loglevel</i></code>	Set log level. The following log levels may be used: <code>always</code> , <code>critical</code> , <code>error</code> , <code>warning</code> , <code>info</code> , <code>moreinfo</code> , <code>freq</code> , <code>morefreq</code> , <code>trace</code> , <code>busy</code> . The default is <code>info</code> .
<code>-trace <i>trace_class</i></code>	Trace selected events. The following trace classes may be used: <code>all</code> , <code>rpcs</code> , <code>updates</code> , <code>transactions</code> , <code>monitor</code> . If you specify this option, you must also specify <code>-tracefile</code> and/or <code>-tracelog</code> . No tracing is done, even if it is requested for one or more classes of events, unless either or both of <code>-tracefile</code> or <code>-tracelog</code> is specified. The default is <code>transactions</code> .
<code>-tracefile <i>filename</i></code>	Set trace filename. There is no default.

Option	Description
<code>-tracefilemax <i>maximum_size</i></code>	Set trace file maximum size (in bytes). If the file exceeds the maximum size, any preexisting <code>filename.old</code> will be deleted, the current file will be renamed to <code>filename.old</code> , and a new file will be created.
<code>-[no]tracelog</code>	[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 <code>-tracelog</code> .
<code>-[no]parent_timer</code>	[Do not] exit when parent exits. The default is <code>-noparent_timer</code> .
<code>-[no]daemonize</code>	[Do not] run as a daemon. The default is <code>-daemonize</code> .
<code>-l</code>	Do not run as a daemon.
<code>-h</code>	Print usage message.
<code>-o help</code>	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 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(1M)` 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 `/var/adm/SYSLOG`.

- 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.
```

(Optional) `/.rhosts`

If you want to use the connectivity diagnostics provided with CXFS, ensure that the `/.rhosts` file on each node allows all the hosts in the cluster to have access to each other in order to run remote commands such as `rsh(1)`. The connectivity tests execute a `ping(1)` 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(1)` (user `root`). For example, suppose you have a cluster with three nodes: `cxfs0`, `cxfs1`, and `cxfs2`. The `/.rhosts` file on each 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
```

(Optional) /etc/exports

The /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 unless the filesystem is re-exported after the CXFS mount.

(Optional) Configure 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 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
```

Configure 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.

To configure 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 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.

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.

See "Hostname Resolution: `/etc/sys_id`, `/etc/hosts`, `/etc/nsswitch.conf`", page 64.

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`. See "Hostname Resolution: `/etc/sys_id`, `/etc/hosts`, `/etc/nsswitch.conf`", page 64.

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` 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 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)

6. Configure an e-mail alias on each node that sends e-mail notifications of cluster transitions to a user outside the CXFS cluster and to a user on the other nodes in the cluster. For example, suppose there are two nodes called `cxfs1` and `cxfs2`. On `cxfs1`, add the following line to the `/etc/aliases` file:

```
cxfs_admin:operations@console.xyz.com,admin_user@cxfs2.xyz.com
```

On `cxfs2`, add the following line to the `/usr/lib/aliases` file:

```
cxfs_admin:operations@console.xyz.com,admin_user@cxfs1.xyz.com
```

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

Note: You must run the `newaliases(1M)` command after editing `/usr/lib/aliases`.

7. If FDDI is being used, finish configuring and verifying the new FDDI station, as explained in the FDDIXPress release notes and the *FDDIXPress Administration Guide*.

Configure the Serial Ports

If a node is configured to reset another machine, you must turn off the `getty` process for the tty ports to which the reset serial cables are connected. You must do this on the node performing the reset (not the node receiving the reset). To do this, perform the following steps on each node:

1. Determine which port is used for the reset serial line.
2. Open the file `/etc/inittab` for editing.
3. Find the line for the port by looking at the comments on the right for the port number from step 1.
4. Change the third field of this line to `off`. For example:

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

5. Save the file.

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

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

Note: If you configure a multinode 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.

Reboot the System

Execute the following command on each node to reboot it:

```
# reboot
```

The shutdown process then runs `autoconfig(1M)` to generate the kernel with your changes.

Test the System

This section discusses the following:

- "Private Network Interface"
- "Serial Reset Connection", page 77

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:

```
# /usr/etc/ping -c 3 nodeIPAddress
```

Typical `ping(1)` 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 ttl=254 time=3 ms
64 bytes from 190.x.x.x: icmp_seq=1 ttl=254 time=2 ms
64 bytes from 190.x.x.x: icmp_seq=2 ttl=254 time=2 ms
```

If ping fails, follow these steps:

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

```
# /usr/etc/ifconfig ec3
ec3: flags=c63<UP,BROADCAST,NOTRAILERS,RUNNING,FILTMULTI,MULTICAST>
inet 190.x.x.x netmask 0xffffffff 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 Reset Connection

To test the serial reset connections, do the following:

1. Ensure that the nodes and the serial multiplexer are powered on.
2. Start the `cmgr(1M)` command on one of the nodes in the pool:

```
# cmgr
```

3. Stop CXFS services on each node:

```
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 ...
```

2: Installation of CXFS Software and System Preparation

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, entering 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.

Convert 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. If you intend to run a mixture of 6.5.13f and 6.5.14f, you must ensure that backward-compatibility is not enforced.

Upgrading from 6.5.12f or Earlier

Note: If you are upgrading from 6.5.13f, you do not need to follow the instructions in this section. Your filesystems definitions were automatically and transparently converted while you were 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 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 a Mixture of 6.5.13f and 6.5.14f Nodes

If you intend to run a mixture of 6.5.13f and 6.5.14f nodes in the cluster, you must ensure that any nodes running 6.5.13f do not try to be backward compatible with the old CXFS filesystem definitions. (Otherwise, only some nodes in the cluster would enforce backward compatibility and that could result in loss or corruption of the filesystem definitions.)

To turn off backwards compatibility for the 6.5.13f nodes, run the following command once from any of the nodes in the cluster:

```
# /var/cluster/cmgr-scripts/cxfs_backwards_fs_compat off
```

Running with All Nodes Upgraded to 6.5.14f

Once all the nodes in the cluster have been upgraded to 6.5.14f, 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 a summary of the steps required to initially configure a cluster using either the graphical user interface (GUI) or the `cmgr(1M)` command. You may also wish to use the worksheet provided in Appendix A, "Initial Configuration Checklist", page 253. If you are converting from an existing IRIS FailSafe cluster, see "Set Up an Existing FailSafe Cluster for CXFS with the GUI", page 102.

This chapter assumes that you have already performed the steps in Chapter 2, "Installation of CXFS Software and System Preparation", page 59. It points to detailed descriptions in the task reference chapters and in *XVM Volume Manager Administrator's Guide*.

For the initial installation, SGI **highly** recommends that you use the GUI guided configuration task sets. See "Configuring with the GUI", page 84. You should also read through the entire book, including Chapter 8, "Troubleshooting", page 207, 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 "Install Software", page 60.

Preliminary Steps

Complete the following steps to ensure that you are ready to configure the initial cluster:

- "Verify the License", page 82
- "Start the Cluster Daemons", page 82
- "Verify that the Cluster Daemons are Running", page 82
- "Determine the Hostname of the Node", page 83
- "Verify that the `chkconfig` Flags are On", page 84

During the course of configuration, you will see various information-only messages in the log files. See "Normal Messages", page 228.

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 `/var/flexlm/license.dat` file on each node.

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

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

Start the Cluster Daemons

Enter the following 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 252, and "Daemons", page 27.

Determine the Hostname of the Node

When you are initially configuring the cluster, you must use the IP address or the value of `/etc/sys_id` (which must match the primary name of the IP address for the node in `/etc/hosts`) when logging in to the GUI and when defining the first node in the pool. The value of `/etc/sys_id` is displayed by the `hostname(1)` command. For example:

```
# hostname
cxfs6
```

Also, if you use `nsd(1M)`, you must configure your system so that local files are accessed before the network information service (NIS) or the domain name service (DNS). See "Hostname Resolution: `/etc/sys_id`, `/etc/hosts`, `/etc/nsswitch.conf`", page 64.



Caution: It is critical that these files are configured properly and that you enter the primary name for the first node in the pool; aliases may be used for subsequent node definitions. See "Install Software", page 60.

Verify that the `chkconfig` Flags are On

Ensure that the output from `chkconfig(1M)` shows the following flags set to on:

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

If they are not, set them to on and reboot.

For example:

```
# init 1
# /etc/chkconfig cluster on
# /etc/chkconfig cxfsc_cluster on
# init 2
```

Configuring with the GUI

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

- "Preliminary Steps", page 81
- "Start the GUI", page 84
- "Set Up a New Cluster with the GUI", page 87
- "Set Up a New Filesystem with the GUI", page 88

The node from which you run the GUI affects your view of the cluster. You should wait for a change to appear in the tree view before making another change; the change is not guaranteed to be propagated across the cluster until it appears in the tree view. 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/cxtask
```

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



Caution: For the first node in the pool, it is critical that you use the value of `/etc/sys_id` in the **Server** field. See "Hostname Resolution: `/etc/sys_id`, `/etc/hosts`, `/etc/nsswitch.conf`", page 64.

Because you have not yet defined a cluster, you will be notified that a cluster is not registered, which is an expected message at this point.

Figure 3-1 shows the CXFS Manager window.

3: Initial Configuration of the Cluster

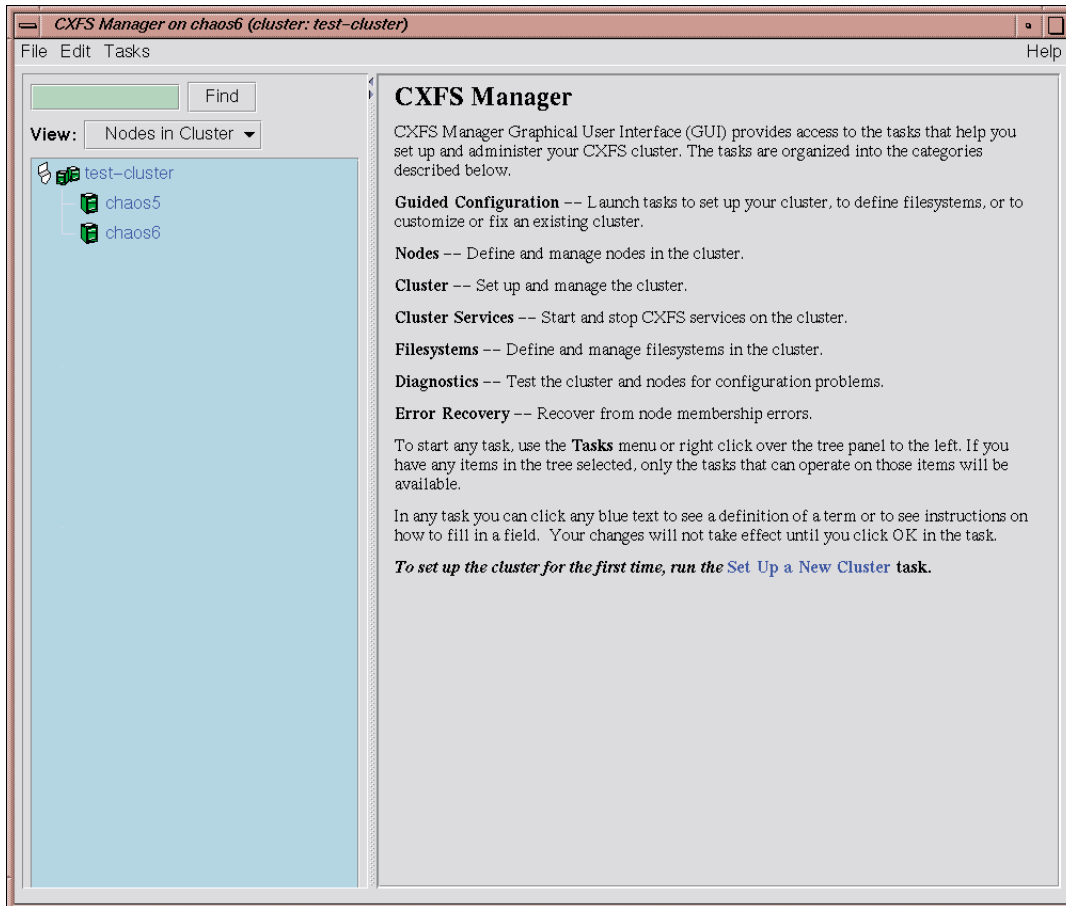


Figure 3-1 CXFS Manager

There are other methods of starting the GUI. For more information, see "Cluster Manager Tools", page 44.

Set Up a New Cluster with the GUI

Note: Within the CXFS tasks, you can click on 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 set 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 on **Define a Node** to define the node to which you connected. The hostname that appears in `/etc/sys_id` is used for this first node definition; see "Determine the Hostname of the Node", page 83. See "Define a Node with the GUI", page 104.
 2. (*Optional*) **After** the first node icon appears in the tree view on the left, click on 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 104.
-

Note: Do not define another node until this node appears in the tree view. 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, SGI recommends that you define only the first three weighted nodes and then continue on to the next step; add the remaining nodes after you have a successful small cluster.

3. Click on **Define a Cluster** to create the cluster definition. See "Define a Cluster with the GUI", page 114. Verify that the cluster appears in the tree view. Choose **View: Nodes in Cluster** in the tree view.
4. After the cluster icon appears in the tree view, click on **Add/Remove Nodes in Cluster** to add the nodes to the new cluster. See "Add/Remove Nodes in the Cluster with the GUI", page 109.

Click on **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 Connectivity with the GUI", page 126. (This test requires the

proper configuration of the `/etc/.rhosts` file; see "(Optional) `.rhosts`", page 71.)

6. Click on **Start CXFS Services**. See "Start CXFS Services with the GUI", page 118. The cluster is operational when the cluster and nodes appear in green in the tree view.
7. Click on **Close**. Clicking on **Close** exits the task set; it does not undo the task.

Set Up a New Filesystem with the GUI

Note: Within the CXFS tasks, you can click on 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 Filesystem** task set 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 on **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 118.
2. Obtain a shell window for one of the nodes in the cluster and use the `fx(1M)` command to create volume headers on the disk drives you will use in the volume. (You do not use the GUI for this step.) For information, see the `fx(1M)` man page and the chapter about performing disk administration procedures in *IRIX Admin: Disks and Filesystems*.
3. In the shell window, create the XVM logical volumes. You can use the `xvm(1M)` `probe` command to display the list of unlabeled volumes. For information, see the *XVM Volume Manager Administrator's Guide*.
4. In the shell window, make the CXFS filesystems using the `mkfs(1M)` command. See the *IRIX Admin: Disks and Filesystems* guide.
5. Returning to the GUI, click on **Define a Filesystem**. See "Define a Filesystem with the GUI", page 122.
6. Click on **Mount a Filesystem**. See "Mount a Filesystem with the GUI", page 123.

Repeat these steps for each filesystem.

Configuring with the `cmgr(1M)` Command

Note: For the initial installation, SGI highly recommends that you use the GUI guided configuration task sets. See "Configuring with the GUI", page 84. For details about `cmgr(1M)` commands, see the man page.

To initially configure the cluster with the `cmgr(1M)` command, do the following:

1. Follow the directions in "Preliminary Steps", page 81.
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 131.

For large clusters, SGI recommends that you define only the first three 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 ID[optional]?
Partition ID[optional] ? (0)
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
```

3: Initial Configuration of the Cluster

```
Sysctrl Owner ? cxfs8
Sysctrl Device ? /dev/ttyd2
Sysctrl Owner Type <tty> ? (tty)
Number of Network Interfaces ? (1)
NIC 1 - IP Address ? cxfs6
NIC 1 - Heartbeat HB (use network for heartbeats) <true|false> ? true
NIC 1 - (use network for control messages) <true|false> ? true
NIC 1 - Priority <1,2,...> ? 1
Node Weight ? (1)
```

Successfully defined node cxfs6

To define the second node, named cxfs7, enter the following:

```
cmgr> define node cxfs7
Enter commands, you may enter "done" or "cancel" at any time to exit
```

```
Hostname[optional] ?
Is this a FailSafe node <true|false> ? false
Is this a CXFS node <true|false> ? true
Node ID[optional] ?
Partition ID[optional] ? (0)
Reset type <powerCycle> ? (powerCycle)
Do you wish to define system controller info[y/n]:y
Sysctrl Type <msc|mmsc|12> ? (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
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
Node Weight ? (1)
```

Successfully defined node cxfs7

To define the third node, named `cxfs8`, enter the following:

```
cmgr> define node cxfs8
Enter commands, you may enter "done" or "cancel" at any time to exit

Hostname[optional] ?
Is this a FailSafe node <true|false> ? false
Is this a CXFS node <true|false> ? true
Node ID[optional] ?
Partition ID[optional] ? (0)
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
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
Node Weight ? (1)
```

Successfully defined node `cxfs8`

You now have three nodes defined in the pool. To verify this, enter the following:

```
cmgr> show nodes in pool

3 Machine(s) defined
    cxfs6
    cxfs7
    cxfs8
```

To show the contents of node `cxfs6`, enter the following:

```
cmgr> show node cxfs6

Logical Machine Name: cxfs6
Hostname: cxfs6.americas.sgi.com
Node Is FailSafe: false
```

```
Node Is CXFS: true
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
Node Weight: 1
```

3. Define the cluster and add the nodes to it. See "Define a Cluster with cmgr", page 144.

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
```

For more information, see "Define a Cluster with cmgr", page 144.

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 169.

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(1M)`, 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(1M)` 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(1M)` 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 150.

5. Obtain a shell window for one of the nodes in the cluster and use the `fx(1M)` command to create a volume header on the disk drive. For information, see *IRIX Admin: Disks and Filesystems*.
6. In the shell window, create the XVM logical volumes. For information, see *XVM Volume Manager Administrator's Guide*.
7. In the shell window, use the `mkfs(1M)` command to make the filesystems. For information, see *XVM Volume Manager Administrator's Guide*.
8. Mount the filesystems by using the `define cxfs_filesystem` subcommand to `cmgr(1M)`. See "Filesystem Tasks with `cmgr`", page 155.

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/dks2d76s0
```

```
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.

3: Initial Configuration of the Cluster

- 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/dks2d77s0**

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.

3: Initial Configuration of the Cluster

- 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
```


GUI Reference

This chapter tells you how to use the CXFS Manager graphical user interface (GUI) to perform the following tasks:

- "Guided Configuration Task Sets"
- "Node Tasks with the GUI", page 103
- "Cluster Tasks with the GUI", page 114
- "CXFS Services Tasks with the GUI", page 117
- "Filesystem Tasks with the GUI", page 121
- "Diagnostic Tasks and Error Recovery with the GUI", page 125

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 "Install Software", page 60.

Guided Configuration Task Sets

There are the following guided configuration task sets:

- "Set Up a New Cluster with the GUI", page 87
- "Set Up a New Filesystem with the GUI", page 88
- "Set Up an Existing FailSafe Cluster for CXFS with the GUI", page 102
- "Check Cluster Status with the GUI", page 198
- "Make Changes to Existing Cluster", page 102
- "Fix or Upgrade Cluster Nodes", page 103

Set Up an Existing FailSafe Cluster for CXFS with the GUI

Note: Within the CXFS tasks, you can click on 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 an Existing FailSafe Cluster for CXFS** task set 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 set appears on the CXFS GUI 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 on **Convert a FailSafe Cluster to CXFS**. This will change the cluster type to CXFS and FailSafe. See "Convert a FailSafe Cluster to CXFS with the GUI", page 116.
2. Stop high availability (HA) services on the nodes to be converted using the FailSafe GUI. See *IRIS FailSafe Version 2 Administrator's Guide*.
3. Add the second heartbeat and control network to the node definitions using the CXFS GUI. See "Modify a Node with the GUI", page 111.
4. Click on **Convert a FailSafe Node to CXFS** to convert the local node (the node to which you are connected). A converted node can be of type CXFS and FailSafe or CXFS. See "Convert a FailSafe Node to CXFS with the GUI", page 113.
5. Click on **Convert a FailSafe Node to CXFS** to convert another node. Repeat this step for each node you want to convert.
6. Click on **Start CXFS Services**.

Make Changes to Existing Cluster

This task set lists different ways to edit an existing cluster. You can make most changes while the CXFS services are active, such as changing the way the cluster administrator is notified of events; however, you 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 116
- "Set Up a New Filesystem with the GUI", page 88
- "Modify a Filesystem with the GUI", page 124
- "Define a Node with the GUI", page 104
- "Test Connectivity with the GUI", page 126
- "Add/Remove Nodes in the Cluster with the GUI", page 109

Fix or Upgrade Cluster Nodes

This task set lead you through steps to remove a node from a cluster in order to perform maintenance on it.

It covers the following steps:

- "Stop CXFS Services (Normal CXFS Shutdown) with the GUI", page 118.
- Perform the necessary maintenance on the node. Only if required, see "Reset a Node with the GUI ", page 110.
- "Start CXFS Services with the GUI", page 118.
- Monitor the state of the cluster components in the tree view. See "Check Cluster Status with the GUI", page 198.

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 set leads you through the tasks required to set up the cluster and nodes. See "Set Up a New Cluster with the GUI", page 87.

Define a Node with the GUI

The first node you define must be the node that you have logged into, in order to perform cluster administration. You **must** use the hostname that appears in `/etc/sys_id` for the first node defined.

Note: Within the CXFS tasks, you can click on 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**.

To define a node, do the following:

1. **Hostname:** Enter the hostname of the node you are defining, such as `mynode.company.com` (this can be abbreviated to `mynode` if it is resolved on all nodes). You **must** use the hostname that appears in `/etc/sys_id` for the first node defined in the pool; this is provided for you by default for the first node. Use the `hostname(1)` command to display the hostname as it appears in the `/etc/sys_id` file. See "Hostname Resolution: `/etc/sys_id`, `/etc/hosts`, `/etc/nsswitch.conf`", page 64.
 2. **Logical Name:** Enter the same as the hostname, or an abbreviation of the hostname (such as `lilly`), 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.
-

Note: If you want to rename a node, you must delete it and then define a new node.

3. **Membership Weight:** Either 0 or 1, which specifies how much weight to give the node in CXFS membership decisions. Use 0 if you do not want the node considered when determining the CXFS membership. The default is 1. At least one node must have a weight of 1. Values other than 0 or 1 are not recommended.

An **initial** cluster must have >50% of the weighted nodes to form a CXFS membership quorum; to maintain the cluster, 50% of the weighted nodes must stay up. For more information, see "Quorum Calculation and Node Weights", page 19.

4. **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.) The priorities of the

networks must be the same for each node in the cluster. For information about using the hostname, see "Hostname Resolution: /etc/sys_id, /etc/hosts, /etc/nsswitch.conf", page 64. For information about why a private network is required, see "Private Network", page 10.

Although the GUI will allow you to enter multiple networks, only the priority 1 network will be used by CXFS; there is no failover of the network. However, you may want to define an additional network that the cluster infrastructure can use for resets, even if the priority 1 network fails.

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

Note: Although the GUI will allow you to specify just **Heartbeat** or just **Control**, you should not choose these settings because they are not permitted in CXFS; CXFS uses a single network for both 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 7.

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

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

If you want to delete a network from the list, click on the network in the list to select it, then click on **Delete**.

5. **Node ID:** (*Optional*) 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. The default ID is a 5-digit number based on the machine's serial number and other machine-specific information; it is not sequential. You must not change the node ID number after the node has been defined. (There is no longer a default CXFS tie-breaker; for more information, see "Membership Quorums", page 19.)
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 `mkpart(1M)` command to determine the partition ID value (see below).

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

7. You can choose whether or not to use the system controller port to reset the node. If you want CXFS to be able to use the system controller to reset the node, you select the **Set Reset Parameters** checkbox and provide the following information:

- This node:
 - **Port Type:** Select **L2**, **MSC** (module system controller), or **MMSC** (multimodule system controller), from the pull-down list. The port type depends upon the node type; for more information about the port type used for your node, click on the blue **Port Type** text.
 - **Port Password:** The password for the system controller port, **not** the node's `root` password or PROM password. On some nodes, the system administrator may not have set this password. If you wish to set or change the system controller port password, consult the hardware manual for your node.
 - **Temporarily Disable Port:** If you want to provide reset information now but do not want to allow the reset capability at this time, check this box. If this box is checked, CXFS cannot reset the node.
- Owner (node that sends reset command):
 - **Logical Name:** Name of the node that sends the remote reset command. Serial cables must physically connect the node being defined and the owner node through the system controller port. At run time, the node must be defined in the CXFS pool.

You can select a logical name from the pull-down list or enter the logical name of a node that is not yet defined. However, you must define the node in CXFS before you run the node connectivity diagnostics task.
 - **TTY Device:** Name of the terminal port (TTY) on the owner node to which the system controller is connected, such as `/dev/ttyd2`. The other end of the cable connects to this node's system controller port, so the node can be controlled remotely by the other node.

If you do not want to use the reset function at all, click on the **Set System Controller Parameters** box to deselect (uncheck) it.

8. Click on **OK**.

You can use the hostname or the IP number as input to the network interface field. However, using the hostname requires DNS on the nodes; therefore, you may want to use the actual IP number.

Note: Do not add a second node until the first node icon appears in the tree view. The entire cluster status information is sent each time a change is made to the configuration database; therefore, the larger the configuration, the longer it will take.

You can use the `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)

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.

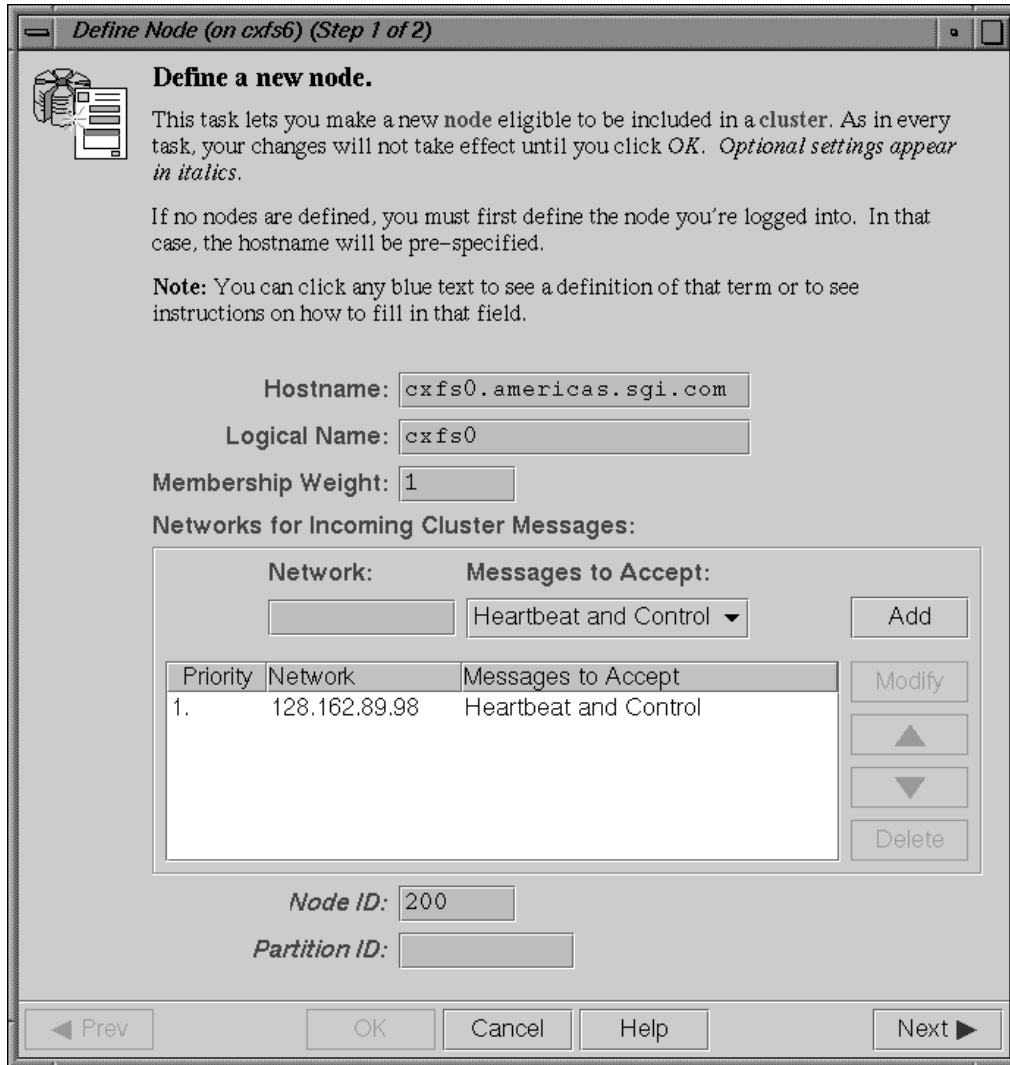


Figure 4-1 Example Node Definition

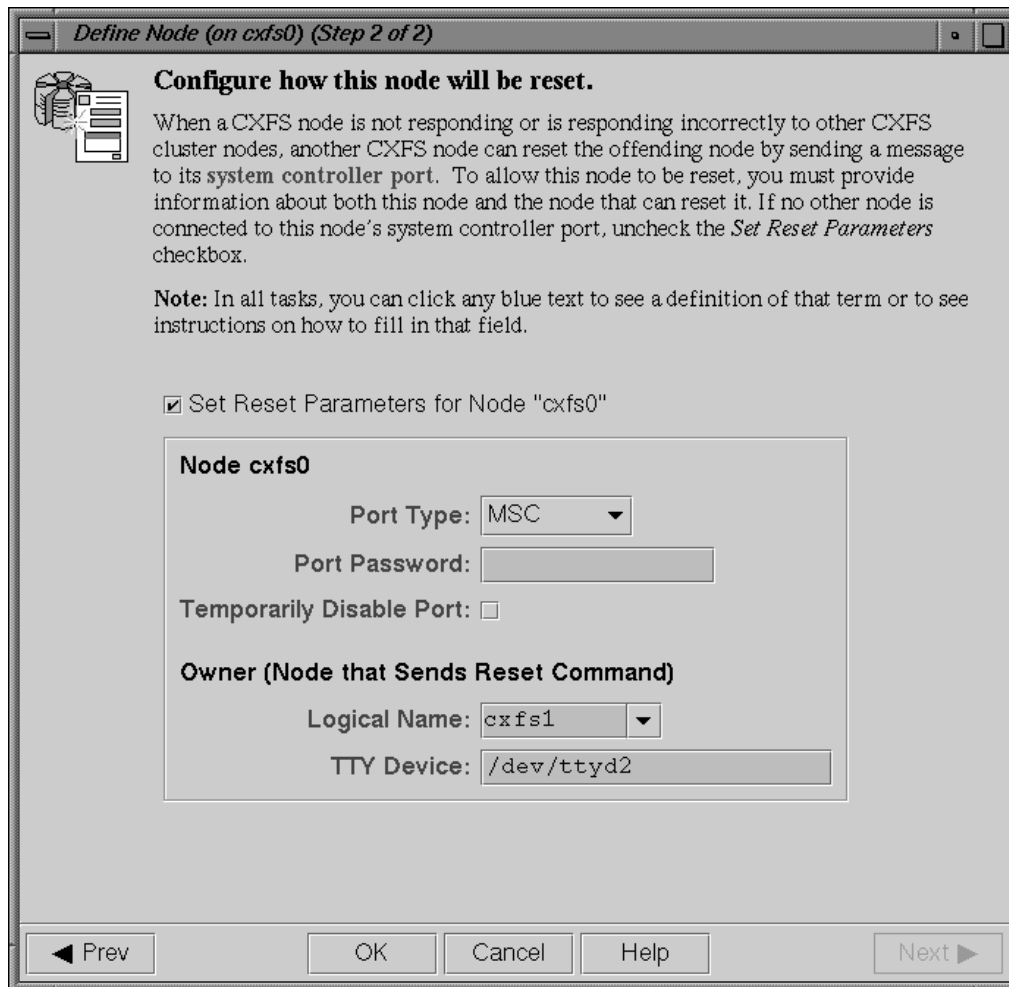


Figure 4-2 Example System Controller Settings

Add/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 tree view; set the tree view to **Nodes in 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 on **Add**. The node name will appear in the **Nodes to Go into Cluster** list. To select all of the available nodes, click on **Add All**.
 - To delete a node, click on its logical name in the **Nodes to Go into Cluster** screen. (The logical name will be highlighted.) Then click on **Remove**.
2. Click on **OK**.

Reset a Node with the GUI

You can use the GUI to reset nodes in a cluster. This sends a reset command to the system controller port on the specified node. When the node is reset, other nodes in the cluster will detect the change and remove the node from the active cluster. When the node reboots, it will rejoin the CXFS 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 on **OK**.

Modify a Node with the GUI

After you have defined a node, you can modify it. If you want to rename a node, you must delete it and then define a new node.

To modify 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. **Membership Weight:** If you want to change the CXFS membership weight, click on in the text entry area and use the backspace key to delete the current value. Then add the correct value. Only use a value of 0 or 1. For more information, see "Quorum Calculation and Node Weights", page 19.

Note: CXFS will not immediately take a change to the CXFS membership weight or networks if the node is currently active in the cluster. Changes will be taken into account the next time the node joins the cluster (that is, after a reboot or after stopping and restarting CXFS services).

3. **Networks for Incoming Cluster Messages:** Although the GUI will allow you to enter multiple networks, only the priority 1 network will be used by CXFS; there is no failover of the network. However, you may want to define an additional network that the cluster infrastructure can use for resets, even if the priority 1 network fails. The priorities of the networks must be the same for each node in the cluster.
 - **Network:** If you want to add a network for incoming cluster messages, enter the IP address or hostname into the **Network** text field and click on **Add**.
 - If you want to modify a network that is already in the list, click on the network in the list in order to select it. Then click on **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 on **Add**.
 - If you want to delete a network, click on the network in the priority list in order to select it. Then click on **Delete**.
 - If you want to change the priority a network, click on the network in the priority list in order to select it. Then click on the up and down arrows in order to move it to a different position in the list.

Note: Although the GUI will allow you to specify just **Heartbeat** or just **Control**, you must not choose these settings because they are not permitted in CXFS; CXFS uses a single network for both 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 7.

- **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 `mkpart(1M)` command to determine the partition ID value; see below.

4. Click on **Next** to move to the next page. Change the values as needed.
5. Click on **OK**.

You can use the `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
```

Convert a FailSafe Node to CXFS with the GUI

This task appears on the CXFS GUI only if you also have FailSafe installed.

To convert an existing IRIS FailSafe node (of type `FailSafe`) to either type `CXFS` and `FailSafe` or type `CXFS`, do the following:

1. Stop HA services on the node to be converted using the FailSafe GUI. See *IRIS FailSafe Version 2 Administrator's Guide*.
2. Add the second **Heartbeat and Control** network to the node definition using the CXFS GUI. See "Modify a Node with the GUI", page 111.
3. Enter the following information:
 - **Logical Name:** Choose the logical name of the node from the pull-down list.
 - **Membership Weight:** If you want to change the CXFS membership weight, click on in the text entry area and use the backspace key to delete the current value. Then add the correct value. Only use a value of 0 or 1. For more information, see "Quorum Calculation and Node Weights", page 19.
 - **Keep FailSafe Settings:**
 - To convert to type `CXFS` and `FailSafe`, click the checkbox
 - To convert to type `CXFS`, leave the checkbox blank
 - Click on **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 with the GUI", page 111. 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(1M)` command or the FailSafe GUI. For information about the FailSafe GUI, see *IRIS FailSafe Version 2 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 116.

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 on **OK**.

Display a Node with the GUI

After you define nodes, you can use the **View** selection in the tree view to display the following:

- **Nodes in Pool** shows all nodes that have been defined
- **Nodes in Cluster** shows only those nodes that are members of a specific cluster

Click on any name or icon to view detailed status and configuration information.

Cluster Tasks with the GUI

This section tells how to perform the individual GUI tasks to define, modify, delete, and display a cluster.

Note: The **Set Up a New Cluster** guided configuration task set leads you through the tasks required to set up the cluster and nodes. See "Set Up a New Cluster with the GUI", page 87.

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.
- **Cluster ID:** A unique number within your network in the range 1 through 128. The cluster ID is used by the IRIX 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.
- **Cluster Mode:** Usually, you should set the cluster to the default `Normal` mode.

Setting the mode to `Experimental` turns off heartbeating in the CXFS membership code for CXFS 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 on **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 114, for information about `Experimental` mode.
 - **Notify Administrator:** Select the desired notification. For more information, see "Define a Cluster with the GUI", page 114.
2. Click on **OK**.

To modify the nodes that make up a cluster, see "Add/Remove Nodes in the Cluster with the GUI", page 109.

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.

Convert a FailSafe Cluster to 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 IRIX 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 on **OK**.

The cluster will apply to both IRIS FailSafe and CXFS. To modify the nodes that make up a cluster, see "Add/Remove Nodes in the Cluster with the GUI", page 109.

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 "Modify a Cluster Definition with the GUI", page 116.

To delete a cluster, do the following:

1. **Cluster to Delete:** The name of the cluster is selected for you.
2. Click on **OK**.

Display a Cluster with the GUI

From the **View** selection, you can choose elements to examine. To view details of the cluster, click on the cluster name or icon; status and configuration information will appear in the item view on the right.

You can also use the `cluster_status` command; see Chapter 7, "Monitoring Status", page 195.

CXFS Services Tasks with the GUI

The following tasks tell you how to start and stop CXFS cluster services and how to 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 on **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:
 - **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 a 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.
 - **Force:** If you want to forcibly stop CXFS services even if there are errors (which would normally prevent the stop operation), click on the **Force** checkbox.
2. Click on **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(1M)` or `reboot(1M)` 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 Tie-Breaker Node with the GUI

A *CXFS tie-breaker node* determines whether a CXFS membership quorum is maintained when exactly half of the nodes (of equal CXFS membership weight) are up and can communicate with each other. A CXFS tie-breaker node should always be weighted. There is no default CXFS tie-breaker. For more information, see "Membership Quorums", page 19.



Caution: If the CXFS tie-breaker node in a two-weighted-node cluster 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 reset hardware for all CXFS systems, especially two-weighted-node clusters; reset is for IRIS FailSafe.

The current CXFS tie-breaker node is shown in the detailed view of the cluster.

To set the CXFS tie-breaker node, do the following:

1. **Tie-Breaker Node:** Select the desired node from the list. If there currently is a CXFS tie-breaker, it is selected by default.

To unset the CXFS tie-breaker node, select `None`.

2. Click on **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 `/var/adm/SYSLOG` as well as to individual log files

for each log group. You can customize the logs according to the level of logging you wish to maintain.



Caution: Do not change the names of the log files. If you change the names, errors can occur.

When you define a log configuration, you specify the following information:

- **Log Group:** A set of one or more CXFS processes that use the same log configuration. A log group usually corresponds to one CXFS daemon, such as `crsd`.
- **Log Level:** A number controlling the amount of log messages that CXFS will write into an associated log group's log file.
- **Log File:** The file in which to log messages.

See also "Log Files", page 195.

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 on **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 the 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 186, "/etc/config/cad.options", page 68, and "/etc/config/fs2d.options", page 69.

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 /var/adm/SYSLOG 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(1M)` command uses a set of numbers to indicate these log levels. See "Configure Log Groups with `cmgr`", page 153.

2.

- **Log File:** Do not change this value.

3. Click on **OK**.

Filesystem Tasks with the GUI

The following tasks let you configure 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 Filesystem** guided configuration task set leads you through the steps required to set up a new filesystem. See "Set Up a New Filesystem with the GUI", page 88.

Define a Filesystem with the GUI

This task assumes that you have created volume headers on your disk drives, created the XVM logical volumes, and made the filesystems. For more information, see "Configuring with the GUI", page 84.

To define a filesystem's metadata server, do the following:

1. Enter the following information:
 - **Device Name:** This 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/` and therefore this information is provided for you by default. The last portion of the name (following `cxvm`) is automatically used by the GUI as the logical filesystem name; to use a different logical name, use the `cmgr` command; see "Define a Filesystem with `cmgr`", page 155.
 - **Mount Point:** This value 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(1M)` man page.
 - *(Optional)* **Mount Options:** These options are passed to the `mount(1M)` command and are used to control access to the specified XVM volume. For a list of the available options, see the `fstab(4)` man page.
 - **Metadata Servers:** A list of nodes that will act as metadata servers. To add a node to the list of servers, choose a name from the pull-down node list and click on **Add**. To select all nodes, click on **Add All**.

To remove a node from the list of servers, click on the name in the list to select it and then click on **Remove**.

Note: The order of servers is significant. The first node listed is the preferred metadata server. Click on a logical name to select it and then click on 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 metadata server.

- **Enable Mount on:** A choice of either all nodes in the cluster or a list of selected nodes that you specify below. (The filesystem is always mounted on the current metadata server.)
2. **Selected Nodes:** If you choose **Only Selected Nodes** above, you can select the desired nodes from the **Node** list. You can also use the **Add All** button to select all nodes, which is the same as selecting **All Nodes Currently in Cluster**.
 3. **If Nodes are Added to the Cluster Later:** If you prefer that the filesystem be mounted on all nodes which might be added to the cluster at some later date, select this option. This choice is selected by default.
 4. Click on **OK**.

Mount a Filesystem with the GUI

To mount an existing filesystem on all nodes in the cluster, do the following:

1. **Filesystem to Mount:** Choose a filesystem from the pull-down menu.
2. Click on **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 118.

Unmount a Filesystem with the GUI

To unmount a filesystem from all nodes in the cluster, do the following:

1. Enter the following information:
 - **Filesystem to Unmount:** Choose a filesystem from the pull-down menu.
 - **Force:** Click on the **Force** toggle button to force an unmount using 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. The force option is off by default. For more information, see the `umount(1M)` man page.
2. Click on **OK**.

Modify a Filesystem with the GUI

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(1M)` command and are used to control access to the specified XVM volume. For a list of the available options, see the `fstab(4)` man page.
 - **Metadata Servers:**
 - To delete a node from the list of servers, click on its name and then click on **Delete**.
 - To add a new node to the list of servers, select it from the pull-down list and click on **Add**. To select all nodes, select **Add All**.
 - To rearrange the priority of a server, select it by clicking on its name and then click on 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 below.
 - **Selected Nodes:** If you choose **Only Selected Nodes** above, you can select the desired nodes from the **Node** list. You can also use the **Add All** button to select all nodes, which is the same as selecting **All Nodes Currently in Cluster**.
 - **If Nodes are Added to the Cluster Later:** If you prefer that the filesystem be mounted on all nodes which might be added to the cluster at some later date, select this option. This choice is selected by default.
2. Click on **OK**.

Relocate a Metadata Server for a Filesystem with the GUI

Note: Deferred implementation.

You can relocate a metadata server for a filesystem to any other potential metadata server in the list. The filesystem must be mounted on the system to which the GUI is connected.

- Enter the following information:
 - **CXFS 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 118.

- Click on **OK** to complete the task.

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

Delete a Filesystem with the GUI

You cannot delete a filesystem that is currently mounted. To unmount a filesystem, see "Unmount a Filesystem with the GUI", page 123.

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 on **OK**.

Diagnostic Tasks and Error Recovery with the GUI

This section tells you how to test connectivity using the GUI. You can test the network and serial connections on the nodes in your cluster by entering the requested

inputs. You can test all of the nodes in the cluster at one time, or you can specify an individual node to test.

Test Connectivity with the GUI

The **Test Node Connectivity** screen requires `rsh(1)` 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 on 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 on **Add**.

To delete a node from the list of nodes to be tested, click on the logical name to select it and then click on **Delete**.

3. To start the tests, click on **Start Tests**. To stop the tests, click on **Stop Tests**.
4. To run another test, click on **Clear Output** to clear the status screen and start over with step 3.
5. To exit from the window, click on **Close**.

Revoke Membership of the Local Node with the GUI

You should revoke CXFS 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 180).

To revoke CXFS 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 on **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 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 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 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 180.

You must actively allow CXFS 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 126.
- 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 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 on **OK** to complete the task.

cmgr Reference

You can enter either of the following:

```
# /usr/cluster/bin/cmgr
# /usr/cluster/bin/cluster_mgr
```

You can use the `-p` option for extra prompting. For general information about using `cmgr`, see "cmgr(1M) Overview", page 51.

This chapter tells you how to use the `cmgr(1M)` command to perform the following tasks:

- "Set Configuration Defaults with `cmgr`", page 130
- "Node Tasks with `cmgr`", page 130
- "Cluster Tasks with `cmgr`", page 144
- "CXFS Services Tasks with `cmgr`", page 150
- "Filesystem Tasks with `cmgr`", page 155
- "Diagnostic and Error Recovery Tasks with `cmgr`", page 167
- "Script Example", page 169

This chapter also provides an example of using `cmgr(1M)` commands in a script; see "Script Example", page 169. For an overview of the tasks that must be performed to configure a cluster, see Chapter 4, "GUI Reference", page 101.

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 7, "Monitoring Status", page 195.

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(1M)` command. For more information about licensing, see "Install Software", page 60.

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 each time a change is made to the cluster database; therefore, the larger the 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 partition_id to partitionID
    set reset_type to powerCycle
    set sysctrl_type to msc|mmsc|l2_(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 /dev/ttyd2
    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)
```

Usage notes:

- *logical_hostname* is the same as the hostname (such as `mynode.company.com`), or an abbreviation of the hostname (such as `mynode`), 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 hostname as returned by the `hostname(1)` command on the node being defined. Other nodes in the pool must all be able to resolve this hostname correctly via `/etc/hosts` or a name resolution mechanism. 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. If you do not specify a number, 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. You must not change the node ID number after the node has been defined.

- *partitionID* uniquely defines a partition in a partitioned Origin 3000 system. The `set partition_id` command is optional; if you do not have a partitioned Origin 3000 system, you do not need to use this command.

Note: In an Origin 3000 system, use the `mkpart(1M)` 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
```

If your system is not partitioned, you can skip this command or use a value of 0.

To unset the partition ID, use a value of 0 or none.

- `powerCycle` is the only available reset type
- The system controller type is based on the node hardware, as show in Table 5-1.

Table 5-1 System Controller Types wide

msc	mmsc	
Origin 2000 Deskside	Origin 2000 Rackmount	SGI Origin 3400
Origin 200	SGI 2400	SGI Origin 3800
SGI 2100	SGI 2800	SGI Origin 3200 with optional L2
SGI 2200		

- *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.
- *enabled|disabled* allow you to provide information about the system controller but temporarily disable by setting this value to *disabled* (meaning that CXFS cannot reset the node). To allow CXFS to reset the node, enter *disabled*.
- */dev/ttyd2* is the only legal value for the system controller device.
- *node_sending_reset_command* is the logical name of the node that can reset this node via the system controller port. A node may reset another node when it detects that the node is not responding to heartbeat messages or is not responding correctly to requests. A serial cable must physically connect one of the owner's serial ports to the system controller port of the node being defined. The owner must be a node in the pool. (You can specify the name of a node that is not yet defined. However, the owner must be defined as a node before the node connectivity diagnostic test is run and before the cluster is activated.)
- *tty_device* 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 coexecution cluster, set both values to *true*.
- The node weight should be 0 or 1; 1 is the default. At least one node must have a CXFS membership weight greater than 0.

- *IP_address_or_hostname_(if_DNS)* 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, but only the first priority network is used (there is no failover). SGI requires that this network be private; see "Private Network", page 10.

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: `/etc/sys_id`, `/etc/hosts`, `/etc/nsswitch.conf`", page 64. For information about why a private network is required, see "Private Network", page 10.

For more information, see "Quorum Calculation and Node Weights", page 19, and "Define a Node with the GUI", page 104.

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 104. 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> ? true
Node ID ? [optional] node_ID
Partition ID ? [optional] (0)partition_ID
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>? (msc) model_(based_on_node_hardware)
Sysctrl Password[optional] ? ( )password
Sysctrl Status <enabled|disabled> ? enabled|disabled
Sysctrl Owner ? node_sending_reset_command
Sysctrl Device ? /dev/ttyd2
Sysctrl Owner Type <tty> ? (tty) tty_device
Number of Network Interfaces ? (1) 1|2|3|4|5|6|7|8
NIC 1 - IP Address ? IP_address_or_hostname_(if_DNS)
NIC 1 - Heartbeat HB (use network for heartbeats) <true|false> ? true|false
NIC 1 - (use network for control messages) <true|false> ? true|false
NIC 1 - Priority <1,2,...> ? priority_number
Node Weight ? (1) 0|1
```


For example, in normal mode:

```
cxfs6 # /usr/cluster/bin/cmgr
Welcome to SGI Cluster Manager Command-Line Interface

cmgr> define node cxfs6
Enter commands, when finished enter either "done" or "cancel"

cxfs6 ? set is_failsafe to false
cxfs6 ? set is_cxfs to true
cxfs6 ? set weight to 1
cxfs6 ? add nic cxfs6
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
cxfs6 ? done
```

For example, in prompting mode:

```
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 ID[optional]?
Partition ID ? [optional] (0)
Reset type <powerCycle> ? (powerCycle)
Do you wish to define system controller info[y/n]:n
Number of Network Interfaces ? (1)
NIC 1 - IP Address ? cxfs6
NIC 1 - Heartbeat HB (use network for heartbeats) <true|false> ? true
NIC 1 - (use network for control messages) <true|false> ? true
NIC 1 - Priority <1,2,...> ? 1
Node Weight ? (1)
```

Successfully defined node cxfs6

Modify a Node with cmgr

To modify an existing node, use the following commands:

```
modify node logical_hostname
  set hostname to hostname
  set nodeid to nodeID
  set partition_id to partitionID
  set reset_type to powerCycle
  set sysctrl_type to msc|mmsc|l2_(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 /dev/ttyd2
  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)
```

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 131.



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

Example of Modifying the Node Weight with cmgr

To change to a node weight of 0, you would enter the following, pressing Enter to keep a current setting (the Enter key is not shown in the examples):

```
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)
Is this a CXFS node <true|false> ? (true)
Node ID[optional] ? (13203)
Partition ID[optional] ? (0)
Reset type <powercycle> ? (powerCycle)
Do you wish to modify system controller info[y/n]:n
Number of Network Interfaces ? (1)
NIC 1 - IP Address ? (cxfs6)
NIC 1 - Heartbeat HB (use network for heartbeats) <true|false> ? (true)
NIC 1 - (use network for control messages) <true|false> ? (true)
NIC 1 - Priority <1,2,...> ? (1)
Node Weight ? (1) 0

```

Successfully modified node cxfs6

Following shows the results:

```

cmgr> show node cxfs6
Logical Machine Name: cxfs6
Hostname: cxfs6.americas.sgi.com
Node Is FailSafe: false
Node Is CXFS: true
Nodeid: 13203
Reset type: powerCycle
ControlNet Ipaddr: cxfs6
ControlNet HB: true
ControlNet Control: true
ControlNet Priority: 1
Node Weight: 0

```

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
```

```
Reset type <powerCycle> ? (powerCycle)
```

```
Do you wish to modify system controller info[y/n]:n
```

```
Number of Network Interfaces ? (2)
```

```
NIC 1 - IP Address ? (preston)
```

```
NIC 1 - Heartbeat HB (use network for heartbeats) <true|false> ? (true)
```

```
NIC 1 - (use network for control messages) <true|false> ? (true)
```

```
NIC 1 - Priority <1,2,...> ? (1)
```

```
NIC 2 - IP Address ? (192.168.168.1)
```

```
NIC 2 - Heartbeat HB (use network for heartbeats) <true|false> ? (true)
```

```
NIC 2 - (use network for control messages) <true|false> ? (true)
```

```
NIC 2 - Priority <1,2,...> ? (2)
```

```
Node Weight ? (1)
```

```
Successfully modified node n_preston
```

```
cmgr> show node n_preston
```

```
Logical Machine Name: n_preston
```

```
Hostname: preston.engr.sgi.com
```

```
Node Is FailSafe: true
```

```
Node Is CXFS: true
```

```
Nodeid: 606
```

```
Partition id: 1
```

```
Reset type: powerCycle
```

```
ControlNet Ipaddr: preston
```

```
ControlNet HB: true
```

```
ControlNet Control: true
ControlNet Priority: 1
ControlNet Ipaddr: 192.168.168.1
ControlNet HB: true
ControlNet Control: true
ControlNet Priority: 2
Node Weight: 1
```

To unset the partition ID, use a value of 0 or none.

Reset a Node with `cmgr`

When CXFS is running, you can reset a node with the following command:

```
admin reset node hostname
```

This command uses the CXFS daemons to reset the specified node.

You can reset a node in a cluster (even when the CXFS daemons are not running) by using the `standalone` option of the `admin reset` command:

```
admin reset standalone node hostname
```

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 cxf6
Enter commands, when finished enter either "done" or "cancel"

cxf6 ? set is_FailSafe to true
cxf6 ? done
```

Successfully modified node cxf6

For example, in prompting mode:

```
cmgr> modify node cxf6
Enter commands, you may enter "done" or "cancel" at any time to exit

Hostname[optional] ? (cxf6.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)
Reset type <powerCycle> ? (powerCycle)
Do you wish to modify system controller info[y/n]:n
Number of Network Interfaces ? (1)
NIC 1 - IP Address ? (cxf6)
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)
Node Weight ? (0)

Successfully modified node cxf6
```

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 cxf6-8 with the following configuration:

```
cmgr> show cluster cxf6-8
Cluster Name: cxf6-8
Cluster Is FailSafe: true
Cluster Is CXFS: true
Cluster ID: 20
```

```
Cluster HA mode: normal
Cluster CX mode: normal
FileSystem Device Name: /dev/cxvm/dks2d72s0
FileSystem Mount Point: /mnts/fs1
FileSystem Mount Options:
FileSystem Status: enabled
FileSystem Force flag: false
    FileSystem Server Node: cxfs6
    FileSystem Server Rank: 0
    FileSystem Server Node: cxfs7
    FileSystem Server Rank: 1
FileSystem Device Name: /dev/cxvm/dks2d73s0
FileSystem Mount Point: /mnts/fs2
FileSystem Mount Options: enabled
FileSystem Status: enabled
FileSystem Force flag: false
    FileSystem Server Node: cxfs8
    FileSystem Server Rank: 0
```

```
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)
Number of Cluster FileSystems ? (0)

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
```

IMPORTANT: NODE cannot be deleted if it is a member of a cluster.
The LOCAL node can not be deleted if some other nodes are still defined.

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
Node Is FailSafe: false
Node Is CXFS: true
Nodeid: 13203
Reset type: powerCycle
ControlNet Ipaddr: cxfs6
ControlNet HB: true
ControlNet Control: true
ControlNet Priority: 1
Node Weight: 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
```

Cluster Tasks with `cmgr`

This section tells you how to define, modify, delete, and display a cluster using `cmgr(1M)`. 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 114.

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:

- *clustername* 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 IRIX 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.
- *notify_command* is the command to be run whenever the status changes for a node or cluster.
- *email_address* 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 normally always 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 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
```

The cluster ID is a unique number within your network in the range 1 through 128. The cluster ID is used by the IRIX kernel to ensure 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.

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 144, and "Define a Cluster with the GUI", page 114.

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 cluserid to clusterID
```

The *clusterID* value must be specified when converting an existing FailSafe cluster to CXFS. Specify a unique number within your network in the range 1 through 128. The cluster ID is used by the IRIX 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)
Number of Cluster FileSystems ? (0) 0
```

```
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 cxf6-8
Cluster Name: cxf6-8
Cluster Is FailSafe: false
Cluster Is CXFS: true
Cluster ID: 22
Cluster CX mode: normal
```

```
Cluster cxf6-8 has following 3 machine(s)
    cxf6
    cxf7
    cxf8
```

CXFS 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 cxf6-8
```


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. (If you stop CXFS services using this method, they will not restart when the node is rebooted.)

```
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 Tie-Breaker Node with `cmgr`

A CXFS *tie-breaker node* determines whether a CXFS membership quorum is maintained when exactly half of the nodes (of equal weight) can communicate with each other. A CXFS tie-breaker node should always be weighted. There is no default CXFS tie-breaker.



Caution: If the CXFS tie-breaker node in a two-weighted-node cluster 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 the use of hardware reset for all CXFS clusters, especially two-weighted-node clusters; reset is required for IRIS FailSafe.

To set the CXFS tie-breaker node, use the `modify` command as follows:

```
modify cx_parameters in cluster clustername
set tie_breaker to hostname
```

To unset the CXFS tie-breaker node, use the following command:

```
set tie_breaker to ""
```

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 119.

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.

Configure Log Groups with `cmgr`

You can configure a log group with the following command:

```
define log_group log_group on node hostname [in cluster clustername]
```

The *log_group* variable can be one of the following:

```
clconfd  
cli  
crsd  
diags
```



Caution: Do not change the names of the log files. If you change the names, errors can occur.

The *log_level* variable 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

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
```

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 Filesystem with `cmgr`", page 155.

- The global status of the filesystem is enabled. See "Mount a Filesystem with `cmgr`", page 161.

Define a 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 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
```

Usage notes:

- *logical_filesystem_name* 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/dks2d115s10`, the GUI will automatically supply a logical filesystem name of `dks2d115s10`. 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.

- *devicename* 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/`.
- *mountpoint* 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(1M)` man page.
- *mount_options* are options that are passed to the `mount(1M)` command and are used to control access to the specified XVM volume. For a list of the available options, see the `fstab(4)` man page.
- The `set dflt_local_status` command allows you to say 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 system 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 Filesystem with cmgr", page 161.

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/dks2d76s0
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/dks2d76s1
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/dks2d76s0
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/dks2d77s1**

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 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*. For a filesystem to mount on a given node, both global and local status must be enabled; see "Filesystem Tasks with **cmgr**", page 155.

Nodes must first be enabled by using the `define cxfs_filesystem` and `modify cxfs_filesystem` commands; see "Define a Filesystem with **cmgr**", page 155, and "Modify a Filesystem with **cmgr**", page 162.

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 150.

Unmount a 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 f1 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 f1 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 Filesystem with cmgr

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 Filesystem with `cmgr`", page 155.

For example, in normal mode:

```
cmgr> show cxfs_filesystem fs1 in cluster cxfs6-8
```

```

Name: fs1
Device: /dev/cxvm/dks2d76s0
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/dks2d76s0
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/dks2d76s0
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/dks2d76s0)
```

```
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: Deferred implementation.

To use relocation, the filesystem must be mounted on the system that is running `cmgr`. To relocate a metadata server to another node, use the following command:

```
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
```

Node 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 Filesystem with `cmgr`", page 162.

Delete a 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
```

Diagnostic and Error Recovery Tasks with `cmgr`

This section tells you how to run tests and error tasks.

Test Network 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 "(Optional) `/ .rhosts`", page 71.

Use the following command to test the network connectivity for the nodes in a cluster:

```
test connectivity in cluster clustername [on node hostname1 node hostname2 ...]
```

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,DRVLOCK,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.

Testing the Serial Connections with `cmgr`

See "Serial Reset Connection", page 77.

Revoke Membership of the Local Node with `cmgr`

To revoke CXFS 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 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 membership for the local node permits the node to reapply for CXFS membership. You must actively allow CXFS membership for the local node in the following situations:

- After a manual revocation as in "Revoke Membership of the Local Node with `cmgr`".

- 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 membership for the local node, use the following command:

```
cmgr> admin cxfs_start
```

See also "Shutdown of the Database and CXFS", page 180.

Script Example

The following script defines a three-node cluster of type CXFS and FailSafe. The nodes are of type CXFS.

Note: This example only defines one network interface, as required for CXFS. The hostname is used here for simplicity; however, it requires that you follow the rules in "Hostname Resolution: `/etc/sys_id`, `/etc/hosts`, `/etc/nsswitch.conf`", page 64; you may wish to use the IP address instead to avoid confusion.

This example does not address the system controller, FailSafe resource, and failover policy definitions. For more information on these topics, see *IRIS FailSafe Version 2 Administrator's Guide*.

```
#!/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 weight to 1
    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 weight to 1
    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 weight to 1
    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/dks2d76s0
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/dks2d76s1
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
```


Administration and Maintenance

You can perform offline administration tasks from any node in the cluster. However, when the filesystems are mounted, administration must be done from the server.

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:

- "Executing Scripts Around Mount Operations", page 174
- "Using `fsr(1M)`", page 176
- "Using `find(1)` and `crontab(1)`", page 176
- "Using Hierarchical Storage Management (HSM) Products", page 177
- "Discovering the Metadata Server for a Filesystem", page 177
- "Metadata Server Recovery", page 179
- "Shutdown of the Database and CXFS", page 180
- "Avoiding a Restart of the Database at Reboot", page 186
- "Log File Management", page 186
- "Volume Management", page 187
- "Disk Management", page 188
- "Filesystem Maintenance", page 189
- "Dump and Restore", page 191
- "Cluster Database Backup and Restore", page 192

- "chkconfig Flags", page 193
- "System Tunable Parameters", page 194

If you have upgraded directly from 6.5.12f or earlier, you must manually convert your filesystem definitions to the new format. See "Convert Filesystem Definitions for Upgrades", page 79.

Executing Scripts Around Mount Operations

The `clconfd` daemon checks if the following files exist in the `/var/cluster/clconfd-scripts` directory; if they do exist, it executes them before and after mounting or unmounting a specified CXFS filesystem:

- `cxfs-pre-mount`
- `cxfs-post-mount`
- `cxfs-pre-umount`
- `cxfs-post-umount`

You can create scripts that have these names and use them to apply whatever action is necessary, such as NFS-exporting the filesystem if the CXFS filesystem was successfully mounted. (The executable files can be scripts or binaries.) 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 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` daemon checks the exit code for these scripts. In the case of failure (non-zero), 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.

Example `cxfs-pre-mount`

```
#!/bin/sh
#/var/cluster/clconfd-scripts/cxfs-pre-mount

echo "Preparing to mount CXFS file system \"$1\" >> /dev/console
MNTPT='mount | grep $1 | cut -f 3 -d" "'
if [ -n "${MNTPT}" ] ; then
    /usr/etc/exportfs -u $MNTPT
fi
```

Example `cxfs-post-mount`

```
#!/bin/sh
#/var/cluster/clconfd-scripts/cxfs-post-mount

if [ $2 -eq 0 ] ; then
    echo "Successfully mounted CXFS file system \"$1\" >> /dev/console
    /usr/etc/exportfs -a
else
    echo "FAILED to mount CXFS file system \"$1\" (error=$2) >> /dev/console
fi
exit $2
```

Example `cxfs-pre-umount`

```
#!/bin/sh
#/var/cluster/clconfd-scripts/cxfs-pre-umount

echo "Preparing to unmount CXFS file system \"$1\" >> /dev/console
MNTPOINT=`mount | grep $1 | cut -f 3 -d " "`
if [ -n "${MNTPOINT}" ] ; then
    /usr/etc/exportfs -u $MNTPOINT
fi
```

Example `cxfs-post-umount`

```
#!/bin/sh
#/var/cluster/clconfd-scripts/cxfs-post-umount

if [ $2 -eq 0 ] ; then
    echo "Successfully unmounted CXFS file system \"$1\" >> /dev/console
else
    echo "FAILED to unmount CXFS file system \"$1\" (error=$2) >> /dev/console
    /usr/etc/exportfs -a
fi
exit $2
```

Using `fsr(1M)`

The `fsr(1M)` command can **only** be used on a metadata server for the filesystem it acts upon; the `bulkstat` system call has been disabled for CXFS clients. You should use `fsr` manually, and only on the current metadata server for the filesystem.

Using `find(1)` and `crontab(1)`

Do not include the `find(1)` command in a `crontab(1)` file. 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.

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 known as X/Open Data Storage Management Specification (XSDM). An example of an HSM product is the Data Migration Facility (DMF).

The HSM application must make all of its DMAPI interface calls through the 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.

Note: The current release does not support the relocation of a DMAPI filesystem as used by an HSM such as DMF. This feature will be provided in a future release.

To use HSM with CXFS, do the following:

- Install the `oe.sw.dmi` subsystem on each node in the cluster.
- Use the `dmi` option when mounting a filesystem to be managed. For more information about this step, see "Define a Filesystem with the GUI", page 122, or "Modify a Cluster with `cmgr`", page 147.
- Start the HSM application on the metadata servers for each filesystems to be managed.

Discovering the Metadata Server for a Filesystem

You can discover the metadata server using the GUI or the `cluster_status(1M)` or `clconf_info` commands.

Metadata Server Discovery with the GUI

Do the following:

1. Select **View: Filesystems**
2. In the tree view, click on the name of the filesystem you wish to view. The name of the metadata server is displayed in the item view to the right.

Figure 6-1 shows an example.

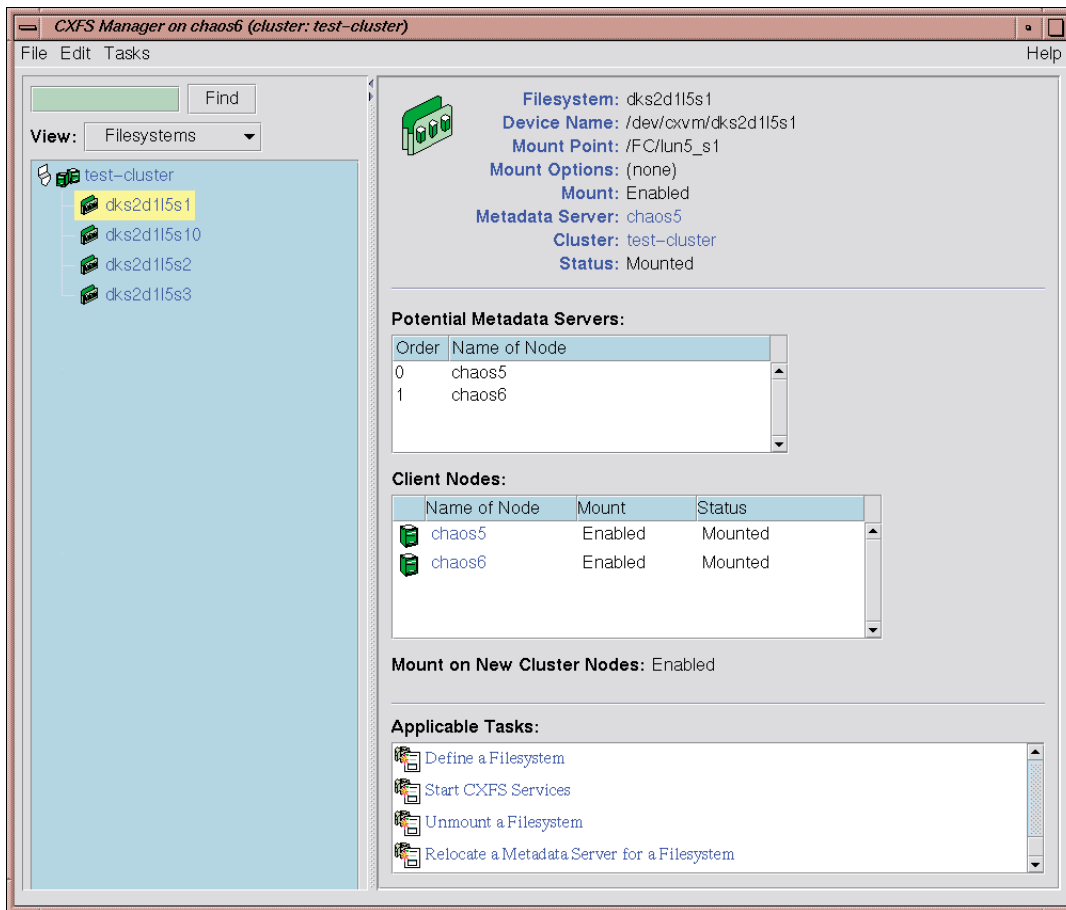


Figure 6-1 Window Showing the Metadata Server

Metadata Server Discovery with `cluster_status`

You can use the `cluster_status` command to discover the 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 198.

Metadata Server Discovery with `clconf_info`

You can use the `clconf_info` command to discover the current 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: Deferred implementation.

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 servers are listed when you define a filesystem.

For more information, see "Define a Filesystem with the GUI", page 122, or "Modify a Cluster with `cmgr`", page 147.

The 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 client dies, the metadata server will clean up after the client. Other 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 membership situation
- Whether any servers have died
- Where the servers are in the recovery order relative to recovering this filesystem

The deceased client is not allowed to rejoin the CXFS 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 Configuration Database Shutdown", page 181
- "Normal CXFS Shutdown", page 182
- "Forced CXFS Shutdown: Revoke Membership of Local Node", page 184

If there are problems, see Chapter 8, "Troubleshooting", page 207. For more information about states, Chapter 7, "Monitoring Status", page 195.

Cluster Configuration Database Shutdown

A *cluster database shutdown* terminates the following user-space daemons that manage the cluster configuration 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:

```
# /etc/chkconfig cluster off
```

Node Status and Cluster Configuration 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.

If the Node is the Metadata Server

If the node is the metadata server for a CXFS filesystem, you should consider moving the metadata server to another node before completing the shutdown if you want the node to remain down.

To move the filesystem to a new metadata server, do the following:

1. Run the `fuser(1M)` command on the metadata server to verify that there are no local users.
2. Unmount the CXFS filesystem from the metadataserver only by using the `umount(1M)` command (this is the only time you should do a manual unmount of a CXFS filesystem):

```
# umount filesystem
```

For information, see "Unmount a Filesystem with the GUI", page 123, "Unmount a Filesystem with `cmgr`", page 162, and "Unmounting Filesystems", page 190.

The relocation process will start automatically, assuming that there are other potential metadata servers defined for the filesystem. For more information, see "Define a Filesystem with the GUI", page 122, or "Modify a Cluster with `cmgr`", page 147. The server that is chosen must be a filesystem client; other filesystem clients will experience a delay during the relocation process.

Restart the Cluster Configuration 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 membership quorum. A normal CXFS shutdown does the following:

- Unmounts all the filesystems
- Terminates the CXFS 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.

To perform a normal CXFS shutdown, enter the following `cmgr(1M)` 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 118.

Note: This action deactivates CXFS services on **one** node, forming a new CXFS 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 membership. See "Node Status", page 200.

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 membership quorum calculation. See "Normal CXFS Shutdown", page 182.

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 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 membership quorum

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 184.

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 118) 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 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 cluster membership
- Marks node as DOWN



Caution: A forced CXFS shutdown may cause the cluster to fail if the cluster drops below CXFS membership quorum.

If you do a forced shutdown on a metadata server, it loses membership immediately. At this point a secondary 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(1M)` command to revoke the 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 126. This action can also be triggered automatically by the kernel after a loss of CXFS 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 `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/Remove Nodes in the Cluster with the GUI", page 109, or "Modify a Cluster with `cmgr`", page 147
- "Delete a Node with the GUI", page 114, or "Delete a Node with `cmgr`", page 140

Rejoining the Cluster after a Forced CXFS Shutdown

After a forced CXFS shutdown, the local node will not resume CXFS membership until the node is rebooted or until you explicitly allow CXFS membership for the local node by entering the following `cmgr(1M)` 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 127.

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 cluster 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:

```
cmgr> admin cxfs_start
```

Reset Capability and Forced CXFS Shutdown



Caution: If you perform forced shutdown on a node with 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 "Hardware Resets", page 24.

Avoiding a Restart of the Database at Reboot

The `cxfs_cluster` flag to `chkconfig(1M)` controls the `clconfd` daemon. If it is turned off, `clconfd` will not be started at the next reboot and the kernel will not be configured to join the cluster. It is useful to turn it off before rebooting the node if you want to temporarily remove the node from the cluster for system or hardware upgrades or for other maintenance work.

Do the following:

```
# /etc/chkconfig cxfs_cluster off
# 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 120.

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 day and month name as suffixes. You may want to place an entry in the `root` crontab to run this script periodically.



Caution: If you run the script twice in one day, it will overwrite the previous saved copy.

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 `/etc/config/cad.options`, page 68, and `/etc/config/fs2d.options`, page 69

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: 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 *XVM Volume Manager Administrator's Guide*.

Disk Management

This section describes the CXFS differences for backups, NFS, Quotas, and SAMBA.

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 a client or server 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 anywhere in the cluster, just as if it was an XFS filesystem.

SAMBA

You can run SAMBA on top of CXFS, allowing NT machines to be clients 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 configuration 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 121; for information about `cmgr(1M)`, see "Cluster Tasks with `cmgr`", page 144.

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 taskset) or with the `modify` subcommand to `cmgr(1M)`. See the following:

- For information about mounting using the GUI, see "Set Up a New Filesystem with the GUI", page 88, and "Define a Filesystem with the GUI", page 122.
- For information about defining and mounting a new filesystem with `cmgr`, see "Modify a Cluster with `cmgr`", page 147.
- For information about mounting a filesystem that has already been defined but is currently unmounted, see "Define a Filesystem with `cmgr`", page 155.

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(1M)` 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.

CXFS filesystems must be mounted on all nodes in the cluster or none. (Otherwise, the filesystem may be mounted on different nodes in an inconsistent way that may result in data loss and/or corruption.) The GUI and `cmgr` will not let you mount a filesystem on a subset of nodes in the cluster.

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 `modify` subcommand to `cmgr`. For information, see "Unmount a Filesystem with the GUI", page 123, or "Unmount a Filesystem with `cmgr`", page 162.

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 metadata servers, the unmount cannot succeed before all the 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 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 a Filesystem with the GUI", page 123, or "Unmount a Filesystem with `cmgr`", page 162.
2. Mount the filesystem as an XFS filesystem. See *IRIX Admin: Disks and Filesystems*.
3. Use the `xfsgrowfs(1M)` command to grow it.
4. Unmount the XFS filesystem with the `umount(1M)` command.
5. Mount the filesystem as a CXFS filesystem. See "Mount a Filesystem with the GUI", page 123, or "Define a Filesystem with `cmgr`", page 155.

Dump and Restore

You must perform dump and restore procedures from the metadata server. The `xfsdump(1M)` and `xfrestore(1M)` commands make use of special system calls that will only function on the CXFS metadata server node.

The filesystem can have active clients during a dump process.

In a clustered environment, a CXFS filesystem may be directly accessed simultaneously by many clients and a 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. It is recommended that the inventory be made accessible by all nodes in the cluster using one of the following methods:

- Relocate the inventory to a shared filesystem.

For example:

- On the node currently containing the inventory, enter the following:

```
# cp -r /var/xfsdump /shared_filesystem
# mv /var/xfsdump /var/xfsdump.bak
# ln -s ../shared_filesystem /var/xfsdump
```

- On all other nodes in the cluster, enter the following:

```
# mv /var/xfsdump /var/xfsdump.bak
# ln -s ../shared_filesystem /var/xfsdump
```
- Export the directory using an NFS shared filesystem.
For example:
 - On the node currently containing the inventory, add `/var/xfsdump` to `/etc/exports` and then enter the following:

```
# exportfs -a
```
 - On all other nodes in the cluster, enter the following:

```
# mv /var/xfsdump /var/xfsdump.bak
# ln -s /hosts/hostname/var/xfsdump /var/xfsdump
```

Note: It is the `/var/xfsdump` directory that should be shared, rather than the `/var/xfsdump/inventory` directory. If there are inventories stored on various nodes, you can use `xfsinvutil(1M)` 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 this state at a later point, such as after a disk crash or after making bad changes.

To perform a backup of the cluster database, use the `cdbBackup(1M)` command.

To perform a restore, do the following:

1. Stop cluster services.
2. Remove the old database.
3. Use the `cdbRestore` command.

For more information, see the `cdbRestore(1M)man` page.

chkconfig Flags

Note: These flags are not normally manipulated with the `chkconfig` command by the administrator; they are set or unset by the GUI or `cmgr`. These flags 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(1M)` command:


- `cluster`, which 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.
- `cxfs_cluster`, which 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.

System Tunable Parameters

Table 6-1 shows the system tunable parameters available with CXFS.

Table 6-1 System Tunable Parameters

Parameter	Description
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.
	<div style="display: flex; align-items: center;">  <p>Caution: Before setting the time-out, you should understand the ramifications of doing so on your system. Modification of this parameter is not generally recommended.</p> </div>
cms_reset_timeout_action	<p>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:</p> <ul style="list-style-type: none"> • 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 continue (that is, fakes acknowledgment) • 2 - Panics the local node

Monitoring Status

You can view the system status in the following ways:

- Monitor log files in `/var/cluster/ha/logs`.
- Use the GUI or the `tail(1)` command to view the end of the `SYSLOG` file.
- Keep continuous watch on the state of a cluster using the tree view or the `cluster_status(1M)` command.
- Query the status of an individual node or cluster using either the GUI or the `cmgr(1M)` command.
- Manually test the filesystems with the `ls(1)` command.
- Monitor the system with Performance Co-Pilot (PCP). You can use PCP 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 *Performance Co-Pilot User's and Administrator's Guide*, *Performance Co-Pilot Programmer's Guide*, and the `dkvis(1)`, `pmie(1)`, `pmieconf(1)`, and `pmlogger(1)` man pages.

Note: You must manually install the XVM statistics for PCP package; it is not installed by default. See "Install Software", page 60.

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

Log Files

You should monitor the following log files listed for problems:

- `/var/adm/SYSLOG` (system log; look for a `Membership delivered` message to indicate that a cluster was formed)
- `/var/cluster/ha/log/cad_log` (events from the GUI and `clconfd`)
- `/var/cluster/ha/log/clconfd_hostname` (kernel status)
- `/var/cluster/ha/log/cli_hostname` (command line interface log)

- `/var/cluster/ha/log/cmond_log` (monitoring of other daemons)
- `/var/cluster/ha/log/crsd_hostname` (reset daemon log)
- `/var/cluster/ha/log/diags_hostname` (output of the diagnostic tools such as the serial and network connectivity tests)
- `/var/cluster/ha/log/fs2d_log` (fs2d membership status)
- `/var/sysadm/salog` (system administration log, which contains a list of the commands run by the GUI)

If the disk is filling with log messages, see "Log File Management", page 186.



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`, or `cmgr` commands.

Key to Icons and Colors

The following tables show keys to the icons and colors used in the CXFS Manager GUI.

Table 7-1 Key to Icons










Icon	Entity
	Node
	Cluster
	Filesystem
	Expanded tree
	Collapsed tree

Table 7-2 Key to Colors

Icon	Color	State
	Gray	Inactive or unknown (CXFS services may not be active)
	Green and white	Enabled for mount (CXFS services may not be active)

Icon	Color	State
	Green	Up or mounted without error
	Red	Down or mounted with error

Check Cluster Status with the GUI

The easiest way to keep a continuous watch on the state of a cluster is to use the tree view 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 200.

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)
```

Check Cluster Status with `cmgr`

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

```
cmgr> show status of cluster cluster_name
```

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 membership. For more information, see "Membership Quorums", page 19.
- **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 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 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

Monitoring Node Status with the GUI

You can use the tree view to monitor the status of the nodes. Select **View: Nodes in Cluster**.

To determine whether a node applies to CXFS, to FailSafe, or both, double-click on the node name in the display. Figure 7-1, page 202 shows an example of a node that is of type CXFS only.

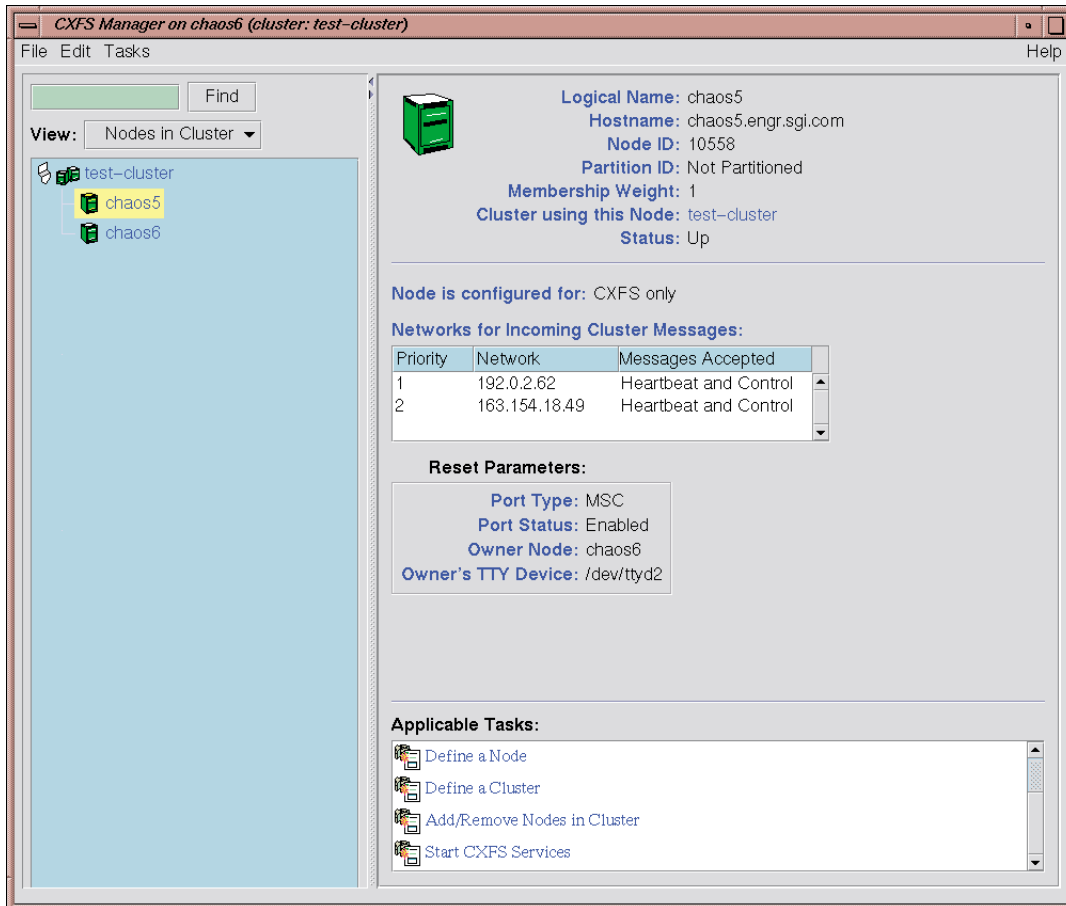


Figure 7-1 Node Status

Querying Node Status with `cmgr`

To query node status, use the following `cmgr(1M)` 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 command:

```
cmgr> admin ping node node_name
```

This command uses the CXFS daemons to test whether the system controller is responding.

You can verify reset connectivity on a node in a cluster even when the CXFS daemons are not running by using the `standalone` option of the `admin ping` command:

```
cmgr> admin ping standalone node node_name
```

This command calls the `ping` command directly to test whether the system controller on the indicated node is responding.

Monitoring Reset Serial Line with `cmgr`

You can use the `cmgr(1M)` 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 "Install Software", page 60

You can use Performance Co-Pilot (PCP) 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(1)` command from the `pcp_eoe.sw.monitor` package. This command is provided with the IRIX release and can be used to produce an ASCII report of selected metrics from the `xvm` group in the PCP namespace of available metrics.
- By using the optional `pmgxvm(1)` command provided with the PCP `pcp.sw.monitor` package (an optional product available for purchase).

If you have the `pcp.sw.monitor` package, you can also use the `pmchart(1)` command to view time-series data in the form of a moving graph. Figure 7-2 shows an example.

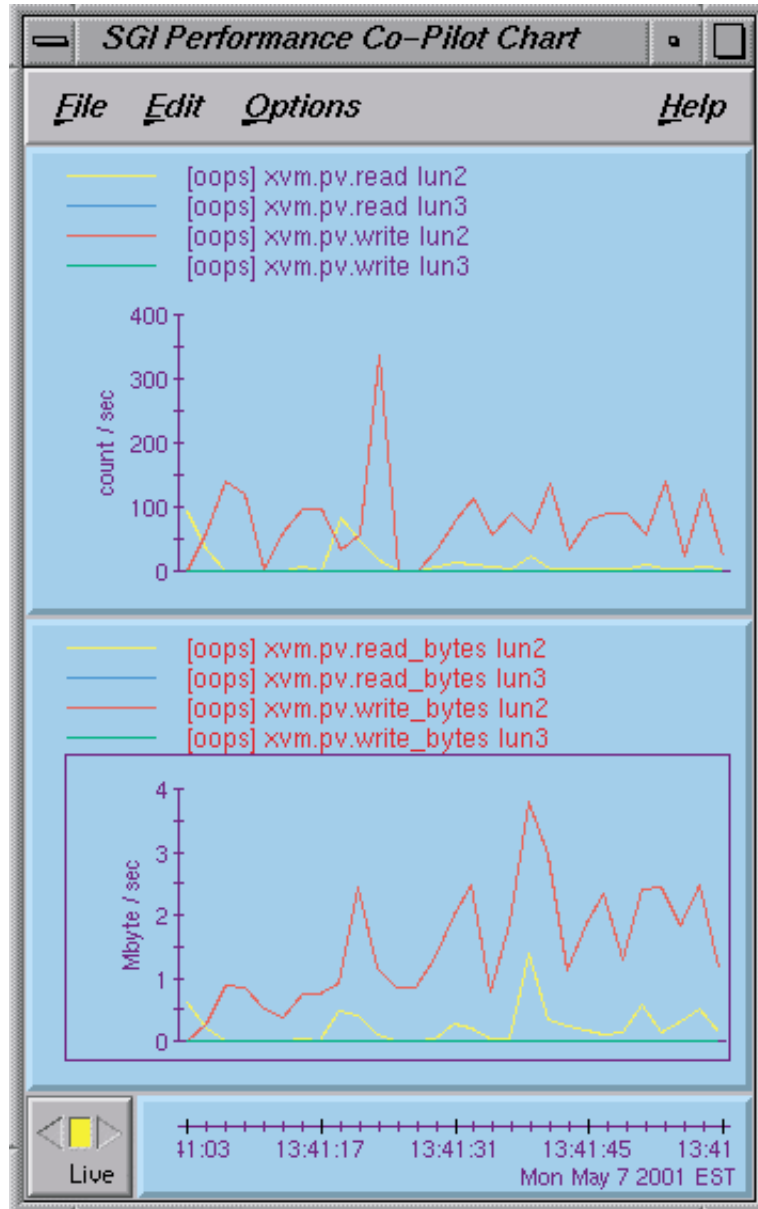


Figure 7-2 pmgxvm chart

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"
- "Avoiding Problems", page 217
- "Common Problems", page 225
- "Understanding Error Messages", page 228
- "Corrective Actions", page 247

Troubleshooting Strategy

To troubleshoot CXFS problems, do the following:

- "Know the Tools"
- "Identify the Cluster Status", page 214
- "Locate the Problem", page 215

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", page 208
- "Cluster Configuration Tools", page 208
- "Cluster Control Tools", page 209

- "Networking Tools", page 210
- "Cluster/Node Status Tools", page 211
- "Performance Monitoring Tools", page 212
- "Kernel Status Tools", page 212
- "Log Files", page 213

Physical Storage Tools

Understand the following physical storage tools:

- To display the hardware inventory, use the `hinv(1M)` command:

```
# /sbin/hinv
```

If the output is not what you expected, do a probe for devices and perform a SCSI bus reset, using the `scsiha(1M)` command:

```
# /usr/sbin/scsiha -pr bus_number
```

- To configure I/O devices, use the `ioconfig(1M)` command:

```
# /sbin/ioconfig -f /hw
```

- To show the physical volumes, use the `xvm(1M)` command:

```
# /sbin/xvm show -v phys/
```

See *XVM Volume Manager Administrator's Guide*.

Cluster Configuration Tools

Understand the following cluster configuration tools:

- To configure XVM volumes, use the `xvm(1M)` command:

```
# /sbin/xvm
```

See *XVM Volume Manager Administrator's Guide*.

- To configure CXFS nodes and cluster, use either the GUI or the `cmgr(1M)` command:
 - The GUI:

```
# /usr/sbin/cxtask
```

See "GUI Overview", page 47 and Chapter 4, "GUI Reference", page 101.
 - The `cmgr(1M)` command line with prompting:

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

See "cmgr(1M) Overview", page 51, and Chapter 5, "cmgr Reference", page 129.
- To reinitialize the database, use the `cdbreinit` command:

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

See "Recreating the Cluster Configuration Database", page 252.

Cluster Control Tools

Understand the following cluster control tools:

- To start and stop the cluster services daemons, use the following commands:

```
# /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:

- "fs2d Membership Quorum Stability", page 219
- "Restarting CXFS Services", page 248
- "Clearing the Cluster Configuration Database", page 248
- "Stopping and Restarting Cluster Infrastructure Daemons", page 252

- To start and stop CXFS services, use the GUI or the following `cmgr(1M)` 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 relocation of the filesystem; avoid this. See "CXFS Services Tasks with `cmgr`", page 150, and "CXFS Services Tasks with the GUI", page 117.

Note: Relocation is deferred in this release.

- To allow and revoke membership on the local node, forcing recovery of the metadata server for the local node, use the GUI or the following `cmgr(1M)` 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 membership until its recovery is complete.

See the following:

- "Revoke Membership of the Local Node with `cmgr`", page 168
- "Allow Membership of the Local Node with `cmgr`", page 168
- "Revoke Membership of the Local Node with the GUI", page 126
- "Allow Membership of the Local Node with the GUI", page 127

Networking Tools

Understand the following networking tools:

- To send packets to network hosts, use the `ping(1)` command:

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

- To show network status, use the `netstat(1)` command:

```
# /usr/etc/netstat
```

Cluster/Node Status Tools

Understand the following cluster/node status tools:

- To show which cluster daemons are running, use the `ps(1)` command:

```
# /sbin/ps -ef | grep cluster
```

See "Verify that the Cluster Daemons are Running", page 82.

- To see cluster and filesystem status, use one of the following:

- GUI:

```
# /usr/sbin/cxdetail
```

See "Display a Cluster with the GUI", page 117.

- `cluster_status(1M)` command:

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

See "Check Cluster Status with `cluster_status`", page 198.

- `clconf_info` command:

```
# /var/cluster/bin/clconf_info
```

- To see the mounted filesystems, use the `mount(1M)` or `df(1)` commands:

```
# /sbin/mount
```

```
# /usr/sbin/df
```

- Use the `df(1)` command to report number of free disk blocks:

```
# /usr/sbin/df
```

- Use the `xvm(1M)` command to show volumes:

```
# /sbin/xvm show vol/
```

See *XVM Volume Manager Administrator's Guide*.

Performance Monitoring Tools

Understand the following performance monitoring tools:

- To monitor system activity, use the `sar(1)` command:

```
# /usr/bin/sar
```
- To monitor file system buffer cache activity, use the `bufview(1)` command:

```
# /usr/sbin/bufview
```

Note: Do not use `bufview` interactively on a busy node; run it in batch mode.

- To monitor operating system activity data, use the `osview(1)` command:

```
# /usr/sbin/osview
```
- To monitor the statistics for an XVM volume, use the `xvm(1M)` command:

```
# /sbin/xvm change stat on {concatname|stripename|physname}
```

See *XVM Volume Manager Administrator's Guide*.
- To monitor system performance, use Performance Co-Pilot. See *Performance Co-Pilot User's and Administrator's Guide*, *Performance Co-Pilot Programmer's Guide*, and the `pmie(1)` and `pmieconf(1)` man pages.

Kernel Status Tools

Understand the following kernel status tools (this may require help from SGI service personnel):

- To determine kernel status, use the `icrash(1M)` 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

- `stthread | 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 membership (the leader)
 - `cms_leader()` indicates that the node is leading the CXFS membership creation
 - `cms_declare_membership()` indicates that the node is ready to declare the CXFS 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 membership
 - `cms_stable()` indicates that the CXFS 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 invoke internal kernel routines that provide useful debugging information, use the `idbg(1M)` command:

```
# /usr/sbin/idbg
```

Log Files

Understand the following log files:

- `/var/adm/SYSLOG` (look for Membership delivered)
- `/var/cluster/ha/log/cad_log`
- `/var/cluster/ha/log/clconfd_hostname`
- `/var/cluster/ha/log/cli_hostname`
- `/var/cluster/ha/log/cmond_log`
- `/var/cluster/ha/log/crsd_hostname`

- `/var/cluster/ha/log/diags_hostname`
- `/var/cluster/ha/log/fs2d_log`
- `/var/sysadm/salog`

See "Log Files", page 195.

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 82.
- Is the cluster state consistent on each node? Run the `clconf_info` command on each node and compare.
- Which nodes are in the CXFS membership? See "Check Cluster Status with `cluster_status`", page 198, "Check Cluster Status with `cmgr`", page 200, and the `/var/adm/SYSLOG` file.
- Which nodes are in the `fs2d` membership? See the `/var/cluster/ha/log/fs2d_log` files on each node.
- Is the database consistent on all nodes? Determine this logging in to each node and examining the `/var/cluster/ha/log/fs2d_log` file and database checksum.
- Log onto the various client nodes or use the GUI tree view display with details showing to answer the following:
 - Are the devices available on all nodes? Use the following:
 - The `xvm(1M)` command to show the physical volumes:

```
xvm:cluster> show -v phys/
```
 - List the contents of the `/dev/cxvm` directory with the `ls(1)` command:

```
# ls /dev/cxvm
```
 - Use the `hinv(1M)` command to display the hardware inventory:

```
# hinv
```


- Are the filesystems mounted on all nodes? Use `mount(1M)` and `clconf_info` commands.
- Which node is the metadata server for each filesystem? Use the `cluster_status(1M)` or `clconf_info` commands.

On the metadata server, use the `clconf_info` command.

- Is the metadata server in the process of recovery? Use the `icrash(1M)` command to search for messages and look at `/var/adm/SYSLOG`. See "Kernel Status Tools", page 212. 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(1M)` command:

```
xvm:cluster> show vol/*
```

Locate the Problem

To locate the problem, do the following:

- Examine the log files (see "Log Files", page 213):
 - Search for errors in all log files. See "Log Files", page 195. Examine all messages within the timeframe in question.
 - Trace errors to the source. Try to find an event that triggered the error.

- Use the `icrash` commands. See "Kernel Status Tools", page 212.
- Use detailed information from the tree view in the GUI to drill down to specific configuration information.
- Run the **Test Connectivity** task in the GUI. See "Test Connectivity with the GUI", page 126.
- Determine how the nodes of the cluster see the current CXFS membership by entering the following command on each 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 `SYSLLOG` on each node to make sure the CXFS filesystems have been successfully mounted or unmounted. If a mount/unmount fails, the error will be logged and the operation will be retried after a short delay.

- Use the `sar(1)` system activity reporter to show the disks that are active. For example, the following example will show the disks that are active, put the disk name at the end of the line, and poll every second for 10 seconds:

```
# sar -DF 1 10
```

For more information, see the `sar(1)` man page.

- Use the `bufview(1)` file system 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(1)` man page.

- Use the `icrash(1M)` IRIX system crash analysis utility. For more information, see the `icrash(1M)` 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
```

Avoiding Problems

This section covers the following:

- "Proper Start Up", page 218
- "Eliminating a Residual Cluster", page 218
- "fs2d Membership Quorum Stability", page 219
- "Consistency in Configuration", page 219
- "Weight Nodes Appropriately", page 219
- "GUI Use", page 220
- "Log File Names and Sizes", page 220
- "Netscape and the Brocade Switch GUI", page 220

- "Performance Problems with Unwritten Extent Tracking and Exclusive Write Tokens", page 220
- "Avoiding Excessive Filesystem Activity Caused by the `crontab` File", page 222
- "Using System Capacity Wisely", page 222
- "Restarting CXFS after a Forced Shutdown", page 223
- "Remove Unused Nodes", page 223
- "Reboot Before Changing Node ID or Cluster ID", page 223
- "Removing Reset Lines", page 224
- "Appropriate Use of `xfstools`", page 224

Proper Start Up

Ensure that you follow the instructions in "Preliminary Steps", page 81, before configuring the cluster.

Eliminating 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:

```
# icrash -e 'stthread | 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:

```
# icrash -e 'stthread | grep cmsd'
```

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.

fs2d Membership Quorum Stability

The fs2d membership quorum must remain stable during the configuration process. If possible, use multiple windows to display the fs2d_log file for each 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 fs2d 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 fs2d 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 fs2d 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 "Configure System Files", page 64.

Weight Nodes Appropriately

Use appropriate weights for nodes:

- Only use a weight of 0 or 1
- Use an odd number of weighted nodes for stability
- Do not give weight to unstable clients

GUI Use

The GUI provides a convenient display of a cluster and its components through the tree view. 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 tree view 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 45.

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 186. 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 120, or "Configure Log Groups with cmgr", page 153.

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(1) or ksh(1) shells:

```
$ NOJIT=1; export NOJIT
```
- For csh(1) 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 *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(1M)` man page.

Avoiding 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(1)` 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(1M)` 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 these `crontab` entries on all the 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` and `fsr` lines.

The `fsr` command can only be run on the metadata server, so it is not harmful to leave it in the `crontab` file for clients, but it will not be executed.

Using System Capacity Wisely

To avoid a loss of connectivity between the metadata server and the 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(1)` tool.

See also "Out of Logical Swap Space", page 231.

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 Configuration Database", page 252.

Remove Unused Nodes

If a node is going to be down for a while, remove it from the cluster and the pool to avoid `fs2d` membership and CXFS membership quorum problems. See the following sections:

- "Modify a Cluster Definition with the GUI", page 116
- "Modify a Cluster with `cmgr`", page 147
- "Delete a Node with `cmgr`", page 140

Restarting 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 184.

Removing Reset Lines

When reset is enabled, CXFS requires a `reset successful` message before it moves the metadata server. Therefore, if you have the reset capability enabled and you must remove the reset lines for some reason, you must also disable the reset capability. See "Modify a Node with the GUI", page 111, or "Modify a Node with `cmgr`", page 136.

Note: The reset capability **highly** recommended to ensure data integrity, especially for two-weighted-node clusters; reset is required for IRIS FailSafe. See "Cluster Environment", page 7.

Appropriate Use of `xfstool` `repair`

CXFS file systems are really clustered XFS filesystem; therefore, in case of a file system corruption, you can use the `xfstool check(1M)` and `xfstool repair(1M)` 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 `xfstool repair`.



Caution: If you run `xfstool repair` without first replaying the XFS logs, you may introduce data corruption.

You should only run `xfstool repair` in case of an actual filesystem corruption; forced filesystem shutdown messages **do not** necessarily imply that `xfstool repair` should be run. Following is an example of a message that does indicate an XFS file corruption:

```
XFS read error in file system meta-data 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:

```
# xfstool logprint -t device
```

If you run `xfstool` before mounting the filesystem, `xfstool` will delete all of this valuable metadata.

If you think you have a filesystem with real corruption, do this:

1. Mount the device in order to replay the log:

```
# mount device any_mount_point
```

2. Unmount the filesystem:

```
# umount device
```

3. Check the filesystem:

```
# xfs_check device
```

4. View the repairs that could be made, using `xfstool` in no-modify mode:

```
# xfs_repair -n device
```

5. If you are certain that the repairs are appropriate, complete them:

```
# xfs_repair device
```

For more information, see *IRIX Admin: Disks and Filesystems*.

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 82
 - "License Error", page 232
 - "Install Software", page 60
- Are the cluster daemons running? See "Verify that the Cluster Daemons are Running", page 82.
- Are the `tcpmux` and `tcpmux/sgi_sysadm` services enabled in the `/etc/inetd.conf` file?
- Are the `inetd` or `tcp` wrappers interfering? This may be indicated by `connection refused` or `login failed` messages.

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 186. You may also want to consider choosing a less-verbose log level; see the following:

- `/etc/config/cad.options`, page 68
- `/etc/config/fs2d.options`, page 69
- "Configure Log Groups with the GUI", page 120

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: `/etc/sys_id`, `/etc/hosts`, `/etc/nsswitch.conf`", page 64.

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 `vmcore.#.comp`, `unix.#`, `/var/adm/SYSLOG`, and cluster log files available.

Node is Detected but Never Joins Membership

If a node is detected in `/var/adm/SYSLOG` but it never receives a `Membership delivered` message, it is likely that there is a network problem. See "Configure System Files", page 64.

Cell ID Count and "Membership Delivered" Messages

The `Membership delivered` messages in the `/var/adm/SYSLOG` 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 `/var/adm/SYSLOG`, this 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 "Configure System Files", page 64.

You Cannot Log In

If you cannot log in to a 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
```

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"
- "clconfd Daemon Death", page 230
- "Out of Logical Swap Space", page 231
- "No Cluster Name ID Error", page 231
- "Lost CXFS Membership", page 232
- "License Error", page 232
- "IP Address Error", page 232
- "SYSLOG Errors", page 233
- "Log File Errors", page 242

Normal Messages

You can expect to see the following messages. They are normal and do not indicate a problem.

```
NOTICE: Error reading msg 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. However, reset lines are highly recommended for CXFS clusters, especially two-weighted-node clusters; reset lines are required for IRIS FailSafe.

NOTICE: Error reading msg header 4 channel 1 cell 2

The msg 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 `clconf_info` command. The `clconf_info` command first registers as a 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 232.

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 "Install Software", page 60.

Out of Logical Swap Space

The following example `/var/adm/SYSLOG` 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 "Using System Capacity Wisely", page 222.

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 *XVM Volume Manager Administrator's Guide*.

Lost CXFS Membership

The following message in `/var/adm/SYSLOG` 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 127, or "Allow Membership of the Local Node with `cmgr`", page 168.

License Error

If you see the following message in the `/var/cluster/ha/log/clconfd_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 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 "Install Software", page 60.

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 "Eliminating a Residual Cluster", page 218. 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.

SYSLOG Errors

CXFS logs both normal operations and critical errors to `/var/adm/SYSLOG`, as well as to individual log files for each log group.

In general, errors in the `/var/adm/SYSLOG` 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_NOSEVER, clconf
ipc: ipccInt_connect() failed, file /var/cluster/ha/comm/clconfd-ipc_cxfs0
```

Table 8-1 shows the parts of the preceding SYSLOG message.

Table 8-1 SYSLOG Error Message Format

Content	Part	Meaning
Sep 7 11:12:59	Timestamp	September 7 at 11:12 AM.
6X	Facility and level	6X indicates an informational message. See <code>syslogd(1M)</code> and the file <code>/usr/include/sys/syslog.h</code> .
cxfs0	Node name	The node whose logical name is <code>cxfs0</code> is the node on which the process is running.

Content	Part	Meaning
<code>cli[5830]</code>	Process[ID]	The process sending the message is <code>cli</code> and its process ID number is 5830.
<code><CI>E clconf 0</code>	Internal information: message source, logging subsystem, and thread ID	The message is from the cluster infrastructure (CI). <code>E</code> indicates that it is an error. The <code>clconf</code> command is the logging subsystem. <code>0</code> indicates that it is not multithreaded.
<code>CI_IPCERR_NOSEVER, clconf ipc</code>	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.
<code>ipccInt_connect() failed, file /var/cluster/ha/comm/clconfd-ipc_cxfs0</code>	Message text	A connection failed for the <code>clconfd-ipc_cxfs0</code> file.

The following sections present only the message identifiers and text.

cli Errors

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_INVALID, 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_INVALID, CLI private command: failed (IP address (128.162.89.33) specified for control network is cxf0 is assigned to control network of machine (cxf0).)

You specified the same IP address for two different control networks of node cxf0. Use a different IP address.

CI_FAILURE, CLI private command: failed (Unable to validate hostname of machine (cxf0) being modified.)

The DNS resolution of the cxf0 name failed. To solve this problem, add an entry for cxf0 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 252.

clconfd Errors

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 Configuration Database", page 248.

If the database exists, restart the cluster daemons; see "Stopping and Restarting Cluster Infrastructure Daemons", page 252.

CI_ERR_NOTFOUND, Could not get Cellular status for local machine (cxf1)

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 252.

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(1M)` command to modify the node's definition and add reset information. See "Define a Node with the GUI", page 104, or "Modify a Node with `cmgr`", page 136.

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(1M)` 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(1M)` 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(1M)`. You must use `mkfs(1M)` to construct the filesystem before mounting it. For more information, see the `mkfs(1M)` man page.

CI_FAILURE, Could not write newincarnation number to CDB, error = 9.

There was a problem accessing the cluster configuration database. Retry the operation. If the error persists, stop and restart the cluster

daemons; see "Stopping and Restarting Cluster Infrastructure Daemons", page 252.

If the problem persists, clear the database, reboot, and re-create the database. See "Clearing the Cluster Configuration Database", page 248.

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 client node that is no longer in the cluster database. If you can identify the 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/strip4 /xvm/strip4): /dev/xvm/strip4: Invalid argument
```

You have tried to mount a filesystem without first running mkfs(1M). You must use mkfs(1M) to construct the filesystem before mounting it. For more information, see the mkfs(1M) 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 Errors

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 Configuration Database", page 252.

```
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"`, page 67.

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"`, page 67.

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 Errors

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 node while some other 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 Configuration Database", page 252.

```
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 (non-zero) status. Look at the corresponding daemon logs for error messages.

fs2d Errors

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 252.

If the problem persists, clear the database, reboot, and re-create the database. See "Clearing the Cluster Configuration Database", page 248.

```
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 252.

If the problem persists, clear the database, reboot, and re-create the database. See "Clearing the Cluster Configuration Database", page 248.

```
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 252.

If the problem persists, clear the database, reboot, and re-create the database. See "Clearing the Cluster Configuration Database", page 248.

```
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 252.

If the problem persists, clear the database, reboot, and re-create the database. See "Clearing the Cluster Configuration Database", page 248.

```
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 Configuration Database", page 252.

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 Errors

CXFS maintains logs for each of the CXFS daemons. For information about customizing these logs, see "Set Log Configuration with the GUI", page 119.

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 8-2, page 243, shows the parts in the preceding message.

Table 8-2 Log Error Message Format

Content	Part	Meaning
cad_log	Timestamp	The message pertains to the cad daemon
Sep 2 17:25:06.092	Timestamp 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 252.

```
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 252.

```
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 252.

```
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_INVALID, Cluster (bob) not defined

The cluster called bob is not present in cluster database.

CI_ERR_INVALID, CLI private command: failed (Cluster (bob) not defined)

The cluster called bob is not present in cluster database.

CI_IPCERR_AGAIN, ipccnt_connect(): file
`/var/cluster/ha/comm/clconfd-ipc_cxfs0` lock failed - Permission denied

The 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, ipccnt_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: ipccnt_connect() failed, file
`/var/cluster/ha/comm/clconfd-ipc_cxfs0`

The local node (`cxfs0`) is not defined in the cluster database.

```
CI_IPCERR_NOSEVER, 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_INVALID, Nodeid -1 is invalid.
I_CONFERR_INVALID, 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_INVALID, 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_CRSEERR_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 cxfsl (1), error 4
```

There are communication problems between the local node and node `cxfsl`. Check the control networks of the two nodes.

```
Communication failure send new quorum to machine cxfsl (102)
(error 6003)
```

There are communication problems between the local node and node `cxfsl`. Check the control networks of the two nodes.

```
Failed to copy CDB transaction to node cxfsl (1)
```

There are communication problems between the local node and node `cxfsl`. Check the control networks of the two nodes.

```
Outgoing RPC to hostname : NULL
```

If you see this message, check your RPC setup. For more information, see the `rpcinfo(1M)`, `rpcinfo(1M)`, and `portmap(1M)` man pages.

Corrective Actions

This section covers the following corrective actions:

- "Restarting CXFS Services", page 248
- "Clearing the Cluster Configuration Database", page 248
- "Recovering a Two-Node Cluster", page 249
- "Rebooting", page 249

- "Rebooting without Rejoining the Cluster", page 251
- "Stopping and Restarting Cluster Infrastructure Daemons", page 252
- "Recreating the Cluster Configuration Database", page 252

Restarting CXFS Services

If CXFS services 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(1M)` 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 118, or "Start CXFS Services with `cmgr`", page 150.

Clearing the Cluster Configuration 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 223.

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 "Eliminating a Residual Cluster", page 218, 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
```

If you want CXFS services to restart whenever the node is rebooted, use the GUI or `cmgr(1M)` to start CXFS services. For information, see "Start CXFS Services with the GUI", page 118, and "Start CXFS Services with `cmgr`", page 150.

The following are situations that may require a rebooting:

- If some 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 nodes and there is no CXFS tie-breaker:
 - `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 weighted nodes **after** the initial CXFS membership is formed. For more information, see "Membership Quorums", page 19.

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 weighted nodes
 - 50% of the weighted nodes, **one of which is the CXFS tie-breaker**

To allow `node2` to form a cluster by itself, you must do the following:

1. Set `node2` to be the CXFS tie-breaker node, using either the GUI or `cmgr`:
 - See "Set Tie-Breaker Node with the GUI", page 119.
 - See "Set the Tie-Breaker Node with `cmgr`", page 151.
2. Revoke the CXFS membership of `node2`:
 - See "Revoke Membership of the Local Node with the GUI", page 126.
 - In `cmgr`, enter:

```
cmgr> admin cxfs_stop
```

See "Revoke Membership of the Local Node with `cmgr`", page 168.
3. Allow CXFS membership of `node2`:
 - See "Allow Membership of the Local Node with the GUI", page 127.
 - In `cmgr`, enter:

```
cmgr> admin cxfs_start
```

See "Allow Membership of the Local Node with `cmgr`", page 168.
4. Unset the CXFS tie-breaker node capability.



Caution: If the CXFS tie-breaker node in a two-weighted-node cluster fails or if the administrator stops CXFS services, the other node will do a forced shutdown, which unmounts all CXFS filesystems. It is highly recommended that you use hardware reset for CXFS to ensure protection of data, especially for two-weighted-node clusters; reset is required for IRIS FailSafe.

Use either the GUI or `cmgr`:

- "Set Tie-Breaker Node with the GUI", page 119
- "Set the Tie-Breaker Node with `cmgr`", page 151

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 membership to form and for filesystems to mount.

Rebooting without Rejoining the Cluster

The `cluster` flag to `chkconfig(1M)` 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:

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

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 248. For general information about the daemons, see "Daemons", page 27.

Recreating the Cluster Configuration Database

To recreate the initial cluster database, do the following:

1. Ensure that the `fs2d` 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 223.

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 8, "Troubleshooting", page 207, 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 1.)
- [] 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 Upgrades 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 switch rather than a hub for performance and control.
- [] Are there reset lines? The reset capability is optional for CXFS but **highly** recommended to ensure data integrity, especially for two-weighted-node clusters.. Reset is **required** for IRIS FailSafe. See "Hardware Components", page 16, and "Hardware Resets", page 24.
- [] Read this book.
- [] Verify that the network is usable. See Chapter 2, "Installation of CXFS Software and System Preparation", page 59.
- [] Install the CXFS software. See "Install Software", page 60.

- [] 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 "Configure System Files", page 64.
- [] Perform other system configuration steps and reboot. See:
 - [] "Configure System Files", page 64
 - [] "(Optional) Configure for Automatic Restart", page 72
 - [] "Configure Network Interfaces", page 73
 - [] "Configure the Serial Ports", page 75
 - [] "Reboot the System", page 76
- [] Completely configure a **small** cluster (3 nodes). See Chapter 3, "Initial Configuration of the Cluster", page 81.
- [] 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 8, "Troubleshooting", page 207.
- [] Set up the filesystems. See Chapter 3, "Initial Configuration of the Cluster", page 81.

Glossary

cell ID

A number associated with a node that is used by the CXFS software and appears in messages.

CLI

Command line interface commands used by the cluster manager tools: the CXFS Manager graphical user interface (GUI) and the `cmgr(1M)` command.

cluster

The set of nodes in the pool that have been defined as a cluster. The cluster is identified by a simple name; this name must be unique within the pool. (For example, you cannot use the same name for the cluster and for a node.)

All nodes in the cluster are also in the pool. However, all nodes in the pool are not necessarily in the cluster; that is, the cluster may consist of a subset of the nodes in the pool. There is only one cluster per pool.

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 `fs2d` daemon.

cluster domain

XVM concept in which a filesystem applies to the entire cluster, not just to the local node. See also *local domain*.

cluster membership

See *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

See *node*

coexecution

The ability to run CXFS and IRIS FailSafe together. For more information, see "Coexecution of CXFS and IRIS FailSafe", page 37.

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.

CXFS database

See *cluster database*.

CXFS membership

The group of CXFS nodes that are 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 membership. Once accepted, the node can share the filesystems of the cluster. (Also known as *kernel-space membership*.) CXFS membership differs from *fs2d database membership* and FailSafe membership. For more information about FailSafe, see *IRIS FailSafe Version 2 Administrator's Guide*.

tie-breaker node

A node identified as a tie-breaker for CXFS to use in the process of computing CXFS 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 tie-breaker. The CXFS tie-breaker differs from the FailSafe tie-breaker; see *IRIS FailSafe Version 2 Administrator's Guide*.

database

See *cluster database*.

database membership

See *fs2d database membership*.

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 membership and *fs2d database membership*. For more information about FailSafe, see *IRIS FailSafe Version 2 Administrator's Guide*.

fs2d database membership

The group of nodes in the **pool** that are accessible to `fs2d` and therefore are able to receive cluster database updates; this may be a subset of the nodes defined in the pool. The `fs2d` daemon manages the distribution of the cluster database (CDB) across the nodes in the pool. (Also known as *user-space membership*.)

heartbeat messages

Messages that cluster software sends between the nodes that indicate a node is up and running. Heartbeat messages and *control messages* are sent through a node's network interfaces that have been attached to a control network.

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.

kernel-space membership

See *CXFS 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 *fs2d database membership* and *CXFS membership*.

membership weight

A number (usually 0 or 1) that is assigned to a node for purposes of calculating CXFS membership quorum.

membership version

A number associated with a node's cell ID that indicates the number of times the CXFS 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 node that coordinates updating of metadata on behalf of all nodes in a cluster. See also *metadata*.

node

An operating system image, usually an individual computer. All nodes must be running adjacent levels of the IRIX operating system; for example, 6.5.14f and 6.5.14f. The nodes are connected to a storage area network (SAN) that connects the storage systems to the nodes in the cluster. A node can belong to only one cluster.

The term *node* does not have the same meaning as a node in an Origin system.

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. The default ID is a 5-digit number based on the machine's serial number and other machine-specific

information; it is not sequential. 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 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 entire set of nodes that are coupled to each other by networks and are defined as nodes in the cluster database. The nodes are usually close together and should always serve a common purpose. A replicated cluster database is stored on each node in the pool.

All nodes that can be added to a cluster are part of the pool, but not all nodes in the pool must be part of the cluster. There is only one pool. (Other pools may exist, but each is disjoint from the other. They share no node or cluster definitions.)

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.)

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.

split-brain syndrome

A situation in which multiple clusters are formed due to a network partition and the lack of reset and/or CXFS tie-breaker capability.

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 (CDB) 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.

tie-breaker node

See *CXFS tie-breaker node*.

user-space membership

See *fs2d database membership*.

quorum

The number of nodes required to form a cluster. For CXFS membership, forming an **initial** membership quorum requires a majority (>50%) of the weighted nodes in the cluster; **maintaining** the existing cluster quorum requires half (50%) of the weighted nodes in the cluster. For *fs2d* database membership, forming and maintaining a membership quorum requires 50% of the nodes in the pool.

weight

See *membership weight*.

Index

6.5.12f and earlier filesystem conversion, 79

A

- activate CXFS services
 - cmgr, 150
- activate/deactivate CXFS services
 - GUI, 118
- ACTIVE cluster status, 198
- add a node to the cluster
 - GUI, 109
- add nic, 131
- adjacent OS allowed, 35
- administration, 173
- administrative shutdown, 182
- age, 216
- allow CXFS membership
 - cmgr, 168
 - GUI, 127
- atime, 33
- AutoLoad boot parameter, 72
- automatic restart of nodes, 72

B

- backups, 188
- bulkstat, 176

C

- CAD options file, 68
- cad process, 27, 82
- cdbreinit, 223, 249, 252
- cdbutil, 217

- cell ID, 216
- change a filesystem
 - cmgr, 162
 - GUI, 124
- change a node
 - cmgr, 136
 - GUI, 111
- Channel traffic, 228
- chkconfig flags, 193
- clconfd process, 27, 82
- clconfd-scripts directory, 174
- clconf_info, 179, 216
- clearing the database, 248
- CLI (underlying command line interface), 45
- cluster chkconfig flag, 193
- cluster conversion
 - cmgr, 147
 - GUI, 116
- cluster daemon restart, 252
- cluster database
 - backup/restore, 192
 - quorum, 219
 - shutdown, 181
- cluster definition
 - cmgr, 144
 - GUI, 114
- cluster deletion
 - cmgr, 148
 - GUI, 117
- cluster display
 - cmgr, 150
 - GUI, 117
- cluster domain, 187
- cluster ID, 117
- Cluster manager tools, 44
- cluster membership, 9
- cluster mode, 116

- cluster modification
 - cmgr, 147
 - GUI, 116
- cluster status, 198
- cluster tasks
 - cmgr, 144
- cluster (term), 8
- cluster_admin subsystem, 27
- cluster_control subsystem, 27
- cluster_mgr
 - See "cmgr", 51
- cluster_services subsystem, 27
- cluster_status, 179, 198
- cmgr
 - command line execution, 55
 - c option, 55
 - exiting, 53
 - help, 52
 - overview, 44, 51
 - script mode, 169
 - scripts and, 53
 - See "configuration tasks", 129
 - shell and, 55
 - template files, 56
- cmgr-templates directory, 56
- cmond process, 27, 82
- cmsd, 28
- cmsd kernel thread, 218
- cms_dead(), 213
- cms_declare_membership(), 213
- cms_follower(), 213
- cms_leader(), 213
- cms_nascent(), 213
- cms_reset_timeout, 194
- cms_reset_timeout_action, 194
- cms_shutdown(), 213
- cms_stable(), 213
- coexecution, 35
- coexecution with IRIS FailSafe, 37
- colors and icons, 197
- command line interface, 45
 - See also "cmgr", 51
- communication paths, 29
- concatenation, 187
- configuration tasks
 - cmgr
 - cluster definition, 144
 - cluster deletion, 148
 - cluster display, 150
 - cluster tasks, 144
 - cmgr, 129
 - connectivity test, 167
 - convert a FailSafe cluster, 147
 - CXFS services, 150
 - cxfs tie-breaker, 151
 - defaults, 130
 - filesystem deletion, 167
 - filesystem modification, 162
 - filesystem mount/unmount, 161
 - log configuration, 153
 - membership allow/revoke, 168
 - metadata server relocation, 166
 - metadataserver definition, 155
 - node definition, 131
 - node deletion, 140
 - node display, 142
 - node modification, 136
 - node reset, 139
 - node tasks, 130
 - notify administrator of cluster changes, 144
- GUI
 - cluster definition, 114
 - cluster deletion, 117
 - cluster display, 117
 - cluster modification, 116
 - connectivity test, 126
 - convert a FailSafe cluster, 116
 - convert a FailSafe node, 113
 - CXFS services start/stop, 118
 - cxfs tie-breaker, 119
 - display a node, 114
 - filesystem deletion, 125
 - filesystem modification, 124

- filesystem mount/unmount, 123
 - log configuration, 119
 - membership allow/revoke, 126
 - metadata server relocation, 125
 - metadataserver definition, 122
 - node addition/removal , 110
 - node definition, 104
 - node deletion, 114
 - node modification, 111
 - node resets, 110
 - set up a new filesystem, 88
 - setting up a new cluster, 87
 - connectivity test
 - cmgr, 167
 - GUI, 126
 - control network, 10, 104
 - convert a FailSafe cluster
 - cmgr, 147
 - GUI, 102, 116
 - convert a FailSafe node
 - cmgr, 139
 - GUI, 102, 113
 - convert filesystem definitions for 6.5.12f and earlier, 79
 - convert from FailSafe task set, 102
 - corpseleader process, 28
 - create a cluster
 - cmgr, 144
 - GUI, 114
 - create a filesystem metadata server
 - cmgr, 155
 - GUI, 122
 - create a node
 - GUI, 104
 - crontab restrictions, 177
 - crsd process, 27, 82
 - ctime, 33
 - curses interface, 198
 - CXFS Manager
 - overview, 47
 - See "configuration tasks, cmgr", 87
 - CXFS membership allow/revoke
 - cmgr, 168
 - GUI, 126
 - CXFS membership
 - current information, 216
 - CXFS membership, 9
 - split, 24
 - CXFS membership problems, 219
 - CXFS services start/stop
 - GUI, 118
 - CXFS services stop/start
 - cmgr, 150
 - CXFS shutdown forced
 - cmgr, 184
 - CXFS shutdown forced, 168
 - GUI, 126
 - CXFS shutdown normal
 - cmgr, 182
 - GUI, 118
 - CXFS tie-breaker node
 - cmgr, 151
 - GUI, 119
 - cxfsd process, 28
 - cxfsfilesystemUpgrade, 79
 - cxfs*mount.sh scripts, 174
 - cxfs*umount.sh scripts, 174
 - cxfs_cluster chkconfig flag, 193
- ## D
- daemon restart, 252
 - daemons, 27, 82
 - data flow, 33
 - data management API, 177
 - Data Migration Facility, 36, 177
 - database
 - clearing, 248
 - dump, 217
 - membership, 9
 - shutdown, 181
 - dcshake process, 28

- deactivate CXFS services
 - cmgr, 151
 - GUI, 118
- define a cluster
 - cmgr, 144
 - GUI, 114
- defragmenter, 3
- delete a cluster
 - cmgr, 148
 - GUI, 117
- delete a filesystem
 - cmgr, 167
 - GUI, 125
- delete a node from the cluster
 - cmgr, 147
 - GUI, 109
- delete a node from the pool
 - cmgr, 140
 - GUI, 114
- diagnostic tasks
 - cmgr, 167
 - GUI, 126
- disk buffering, 6
- disk management, 188
- disk striping, 187
- display a cluster
 - cmgr, 150
 - GUI, 117
- display nodes
 - cmgr, 142
 - GUI, 114
- DMAPI, 177
- DMF, 36, 177
- DNS, 66, 73
- domain, 187
- domain name service , 66
- DOWN node state, 200
- dump, 191
- dump of the database, 217

E

- ERROR cluster status, 198
 - /etc/config/cad.options file, 68
 - /etc/config/fs2d.options file, 69
 - /etc/config/netif.options, 66
 - /etc/fstab not used in CXFS, 2
 - /etc/hosts, 65
 - /etc/init.d, 252
 - /etc/init.d/cxfs stop, 118
 - /etc/mtab, 2
 - /etc/nsswitch.conf, 66
 - /etc/services file, 67
 - /etc/sys_id, 65
- extent tracking, 221

F

- FailSafe coexecution, 37
- FailSafe conversion for a cluster
 - cmgr, 147
 - GUI, 102, 116
- FailSafe conversion for a node
 - cmgr, 139
 - GUI, 113
- FailSafe membership, 10
- filesystem deletion
 - cmgr, 167
 - GUI, 125
- filesystem format changes between 6.5.12f and 6.5.13f, 79
- filesystem growth, 191
- filesystem maintenance, 189
- filesystem metadata server definition
 - cmgr, 155
 - GUI, 122
- filesystem modification
 - cmgr, 162
 - GUI, 124
- filesystem mounting

- cmgr, 147, 161
- filesystem mounting/unmounting, 189
 - GUI, 123
- filesystem reorganizer, 3
- filesystem task set, 88
- filesystem tasks
 - cmgr, 155
 - GUI, 121
- filesystem unmounting
 - cmgr, 162
- filesystems and IRIS FailSafe, 39
- find command and crontab, 177
- FLEXlm license key, 35
- forced CXFS shutdown
 - cmgr, 184
- forced shutdown and restart, 185, 223
- forced shutdown, 184
- forced unmount, 123
- fs2d
 - database membership, 9
 - process, 27
- fs2d options file, 69
- fs2d process, 82
- fs2d_log, 219
- fsr, 176
- fsr_xfs, 3
- fstab not used in CXFS, 2
- fx command, 88

G

- growing filesystems, 191
- guaranteed-rate I/O, 6
- GUI
 - overview, 44
 - tasks
 - See "configuration tasks", 101

H

- heartbeat network, 10, 104
- help
 - for cmgr, 52
 - for GUI, 87
- hierarchical storage management, 177
- hostname control network, 104
- hosts file, 65
- hsm, 177

I

- icons and colors, 196
- icrash, 218
- INACTIVE
 - cluster status, 198
 - node state, 200
- incarnation, 216
- initial cluster configuration
 - cmgr, 89
 - GUI, 84
- installation, 59
- inventory file, 191
- IP address and control network, 104
- IRIS FailSafe coexecution, 37
- IRISconsole, 36
- is_* commands, 131

J

- java files and release level, 63
- java_plugin, 63

K

- kernel-space membership, 9
- kernel-tunable parameters, 194

L

- L2, 132
- LAN, 4
- Legato NetWorker, 188
- license error, 232
- local area networks (LANs), 4
- local domain, 187
- local node, 183
- log configuration
 - cmgr, 153
 - GUI, 119
- log files
 - list of, 195
 - management, 186
 - monitoring, 195

M

- maintenance, 173
- membership
 - See "cluster membership or CXFS membership", 9
- membership weight, 104
- mesgtcpaccept process, 28
- mesgtcpdiscovery, 28
- mesgtcpmulticast, 28
- mesgtcprcv process, 28
- metadata clients, 25
- metadata examples, 4
- metadata server definition
 - cmgr, 155
 - GUI, 122
- metadata server
 - discovery, 177, 179
 - recovery, 180
 - term defined, 11
- metadata server relocation
 - cmgr, 166
 - GUI, 125
- mirroring, 187

- mkfs command, 88
- mkpart, 107, 132
- MMSC, 132
- mode of cluster, 116
- modify a filesystem
 - cmgr, 136, 162
 - GUI, 124
- modify a node
 - cmgr, 136
 - GUI, 111
- mount a filesystem
 - cmgr, 161
 - GUI, 123
- mount options, 122
- mount points, 122
- mount scripts, 174
- mounting, 189
- mount-point nesting, 7
- move the metadata server, 182
- MSC, 132
- mtime, 33

N

- name restrictions, 64
- name service daemon, 66
- named pipes, 6
- nesting of mount points, 6
- NetBackup, 188
- netif.options, 66
- network connectivity
 - cmgr, 167
 - GUI, 126
- network information service, 66
- network interface configuration, 73
- network partition, 25
- network routing, 74
- network segment, 10
- networked filesystem comparison, 4
- NetWorker, 188

networks, 10
 NFS, 188
 NFS-exporting, 174
 NIS, 66, 73
 node addition/removal
 GUI, 109
 node conversion
 cmgr, 139
 GUI, 113
 node definition
 cmgr, 131
 GUI, 104
 node deletion
 cmgr, 140
 GUI, 114
 node display
 cmgr, 142
 GUI, 114
 node modification
 cmgr, 136
 GUI, 111
 node
 definition, 7
 state, 200
 status, 200
 tasks
 See "configuration tasks", 104
 weighting, 19
 node reset
 cmgr, 139
 GUI, 110
 node status
 forced CXFS shutdown, 185
 normal CXFS shutdown, 183
 status
 database shutdown, 181
 node tasks
 cmgr, 130
 GUI, 103
 node weight, 104
 normal CXFS shutdown, 118
 notify administrator of cluster changes

 cmgr, 144
 GUI, 116
 nsadmin, 66
 nsd, 66
 nsswitch.conf, 66
 NT clients, 189
 number of nodes supported, 35
 NVRAM variables, 72

O

Origin 200, 36
 Origin 2000, 36
 OS level, 35

P

parameters (system tunable), 194
 partition of network and reset of hardware, 25
 partition, 132
 pcp_eoe, 204
 pcp_eoe.sw.xvm, 204
 peer-to-disk model, 5
 Performance Co-Pilot (PCP)
 XVM statistics, 64, 204
 physvol, 231
 pipes (named), 6
 platforms, 36
 pmgsvm, 205
 pool, 8
 private network, 10
 private network required, 35
 prompt mode for cmgr, 52

Q

quorum, 19, 219
 quotas, 2, 188

R

- race to reset, 25
- read and metadata flow, 33
- reboot, 76, 249
- reconfigure a cluster, 218
- recovery features, 187
- Recovery process, 28
- recovery
 - of the metadata server, 180
 - terminology, 15
 - two-weighted-node cluster, 44
- rejoin the cluster
 - forced CXFS shutdown, 185
 - normal CXFS shutdown, 184
- relocate a metadata server
 - cmgr, 166
 - GUI, 125
- relocation, 14
 - two-weighted-node cluster, 44
- remote reset connections, 25
- remove a cluster
 - cmgr, 148
 - GUI, 117
- remove a filesystem from a metadata server, 182
- remove a filesystem
 - cmgr, 167
 - GUI, 125
- remove a node from the cluster
 - cmgr, 147
 - GUI, 109
- remove a node from the pool
 - cmgr, 140
 - GUI, 114
- remove nic, 131
- reorganizer, 3
- reset a database, 248
- reset a node
 - cmgr, 139
 - GUI, 110
- reset of hardware, 24
- reset race, 25

- restart cluster daemons, 252
- restart (avoiding), 186
- restarting CXFS services, 248
- restarting nodes automatically, 72
- restore, 191
- revoke CXFS membership
 - cmgr, 168
 - GUI, 126
- /.rhosts file, 71
- rotatelog, 186
- rotating log files, 186
- routed, 74
- routing, 74

S

- SAMBA, 188
- SAN hardware, 36
- SAN, 1
- script mode and cmgr, 169
- serial connections
 - cmgr, 167
 - GUI, 126
- serial line, 11
- serial port configuration, 75
- set commands, 131
- show a cluster
 - cmgr, 150
 - GUI, 117
- show a node
 - cmgr, 142
 - GUI, 114
- shutdown and restart, 185, 223
- shutdown
 - cluster database, 181
 - forced CXFS cluster database, 184
 - normal CXFS database, 182
 - normal CXFS
 - GUI, 118
- shutdown, 182

- size of the cluster, 35
- start CXFS services
 - cmgr, 150
- start CXFS services, 184
- start/stop CXFS services
 - GUI, 118
- status
 - cluster, 198
 - node, 200
 - system controller, 203
 - system, overview, 195
- sthreads, 28
- stop CXFS services
 - cmgr, 151
- storage area network, 1
- striping, 187
- subsystems installed, 61
- swap to a file, 6
- sysctrl* commands, 131
- SYSLOG, 190
- system controller status, 203
- system controller types, 132
- system
 - files, 64
 - software communication paths, 29
 - status, 195
 - view, 27
- system-tunable parameters, 194
- systunes, 194
- sys_id, 65

T

- Tape Management Facility (TMF), 36
- tasks, 48
- template files, 56
- test connectivity
 - cmgr, 167
 - GUI, 126
- threads, 27
- tie-breaker node

- cmgr, 151
 - GUI, 119
- TMP, 36
- tokens, 33
- tools, 44
- TRIX, 36
- troubleshooting, 207
- trusted IRIX, 36
- two-weighted-node cluster, 44

U

- Unified Name Service, 66
- UNKNOWN cluster status, 198
- UNKNOWN node state, 200
- unmount a filesystem
 - cmgr, 162
 - GUI, 123
- unmount scripts, 174
- unmounting, 190
- UNS, 66
- unwritten extent tracking, 221
- UP node state, 200
- upgrading from 6.5.12f or earlier, 79
- user-space membership, 9

V

- /var/cluster/clconfd-scripts directory, 174
- /var/cluster/cmgr-scripts/rotatelogs, 186
- /var/xfsdump/inventory, 191
- VERITAS NetBackup, 188
- volume management, 187

W

- weight, 104
- weighting, 19

write and metadata flow, 33

X

xf quotas, 188

xfsdump and xfsrestore, 191

xf_fsr (fsr_xfs), 3

XSDM, 177

xthreads, 28

xvm command, 88

XVM statistics, 204

XVM, 36, 187

X/Open Data Storage Management
Specification, 177