

# TruCluster Server

---

## Technical Overview

Part Number: AA-RHGVA-TE

**July 1999**

**Product Version:** TruCluster Server Version 5.0

**Operating System and Version:** Tru64 UNIX Version 5.0

This document describes the components and features of TruCluster Server Version 5.0.

---

© 1999 Compaq Computer Corporation

COMPAQ, the Compaq logo, and the Digital logo are registered in the U.S. Patent and Trademark Office.

Microsoft and Windows NT are registered trademarks of Microsoft Corporation. Intel, Pentium, and Intel Inside are registered trademarks of Intel Corporation. UNIX is a registered trademark and The Open Group is a trademark of The Open Group in the US and other countries. Other product names mentioned herein may be the trademarks of their respective companies.

Possession, use, or copying of the software described in this publication is authorized only pursuant to a valid written license from Compaq Computer Corporation or an authorized sublicensor.

Compaq Computer Corporation shall not be liable for technical or editorial errors or omissions contained herein. The information in this document is subject to change without notice.

---

# Contents

## About This Manual

### 1 Introduction to TruCluster Server

### 2 Clusterwide File Systems, Storage, and Device Names

2.1	Supported File Systems .....	2-2
2.2	Cluster File System .....	2-4
2.3	Device Request Dispatcher .....	2-5
2.4	Context-Dependent Symbolic Link .....	2-7
2.5	Device Names .....	2-9
2.6	Worldwide ID .....	2-11
2.7	Clusters and the Logical Storage Manager .....	2-12

### 3 Connection Manager

3.1	Connection Manager .....	3-1
3.2	Votes .....	3-2
3.2.1	Member Votes .....	3-2
3.2.2	Quorum Disk Votes .....	3-2
3.2.3	Member-Specific Cluster Expected Votes .....	3-2
3.3	Calculating Cluster Quorum .....	3-3
3.4	Using a Quorum Disk .....	3-5
3.5	Cluster Partitions .....	3-8
3.6	Monitoring the Connection Manager .....	3-9

### 4 Highly Available Applications

### 5 Cluster Application Availability

5.1	Overview .....	5-1
5.2	CAA Architecture .....	5-2
5.3	Introduction to Resources .....	5-5
5.4	Resource Profiles .....	5-5

5.5	Action Scripts .....	5-7
<b>6</b>	<b>Cluster Alias</b>	
6.1	Overview .....	6-4
6.2	The Default Cluster Alias .....	6-4
6.3	Number of Aliases .....	6-6
6.4	Location of Alias IP Addresses .....	6-6
6.5	Routing for Alias Addresses .....	6-7
6.5.1	Common Subnet Routing .....	6-7
6.5.2	Virtual Subnet Routing .....	6-8
6.5.3	Routing Example .....	6-8
6.6	in_single and in_multi Services .....	6-9
6.7	Alias Attributes .....	6-12
6.8	Service Attributes .....	6-14
6.9	RPC Services and Cluster Alias .....	6-17
6.10	Redirecting Packets Within a Cluster .....	6-18
<b>7</b>	<b>Memory Channel</b>	
<b>8</b>	<b>Distributed Lock Manager</b>	
<b>9</b>	<b>Cluster Installation and Administration</b>	
9.1	Installation .....	9-1
9.2	Administration .....	9-2
<b>Glossary</b>		
<b>Index</b>		
<b>Figures</b>		
1-1	A Cluster's View of Hardware .....	1-2
2-1	Storage Software Layering in a Cluster .....	2-2
2-2	CFS Makes File Systems Available to All Cluster Members ..	2-5
2-3	CDSL Pathname Resolution .....	2-8
3-1	Two-Member deli Cluster Without a Quorum Disk .....	3-6
3-2	Two-Member deli Cluster with Quorum Disk Survives Member Loss .....	3-7
5-1	Application Failover with CAA .....	5-2

5-2	CAA Architecture .....	5-4
6-1	Client's View of a Cluster With and Without Cluster Alias ...	6-1
6-2	Cluster Alias Functional Overview .....	6-3
6-3	Cluster Using Two Aliases .....	6-5
6-4	Alias Routing Example .....	6-9
6-5	in_single Service Accessed Through Default Cluster Alias ....	6-11
6-6	in_multi Service Accessed Through Default Cluster Alias ....	6-12
7-1	Memory Channel Logical Diagram .....	7-2

## Tables

1-1	Features in the TruCluster Server Version 5.0 Product .....	1-3
2-1	UNIX File Systems Supported in a Cluster .....	2-3
2-2	Examples of New Device Names .....	2-10



---

## About This Manual

This manual provides a concise summary of the features available in the TruCluster™ Server Version 5.0 product.

---

### Note

---

The information in this manual does not supersede that in the TruCluster Server *Software Product Description*, which is the definitive legal description of this product.

---

## Audience

This manual is for anyone who is interested in a descriptive overview of the TruCluster Server features and functions.

## Organization

This manual contains the following chapters and glossary:

- Chapter 1 Provides an introduction to TruCluster Server.
- Chapter 2 Describes the clusterwide file systems, context-dependent symbolic links (CDSLs), storage, and device-naming conventions.
- Chapter 3 Introduces the connection manager and its role in forming and maintaining a cluster.
- Chapter 4 Defines the three basic types of highly available applications in a cluster: single-instance, multi-instance, and distributed.
- Chapter 5 Provides an overview of the cluster application availability (CAA) subsystem, which provides clusterwide management for single-instance applications.
- Chapter 6 Provides an overview of the cluster alias subsystem, which makes the cluster look like a single system to the outside world.
- Chapter 7 Describes the Memory Channel interconnect, which provides a high-speed connection among all cluster members.

- Chapter 8 Describes the distributed lock manager (DLM), which provides functions that allow cooperating processes in a cluster to synchronize access to a shared resource.
- Chapter 9 Provides an overview of cluster installation and administration.
- Glossary Defines common terms used throughout the TruCluster Server documentation.

## Related Documents

The following documents and manuals provide detailed information about the TruCluster Server product:

- TruCluster Server *Software Product Description (SPD)* — The legal description of the TruCluster Server Version 5.0 product.
- TruCluster Server *Release Notes* — Provides a brief introduction to TruCluster Server, describes known problems and workarounds, and lists supported hardware (the SPD contains detailed tables of supported hardware).
- TruCluster Server *Hardware Configuration* — Describes how to set up the systems that will become cluster members, and how to configure cluster shared storage.
- TruCluster Server *Software Installation* — Describes how to install the TruCluster Server software.
- TruCluster Server *Highly Available Applications* — Describes how to deploy existing applications in a TruCluster Server cluster and how to write cluster-aware applications.
- TruCluster Server *Cluster Administration* — Describes cluster-specific administration tasks.

## Online Documentation

Most of the documentation for TruCluster Server, including the reference pages, is available on the TruCluster Server Software Version 5.0 CD-ROM in a format that is readable with a Web browser. Much of this documentation is also presented in PDF format.

To read the HTML documentation using the Netscape browser included with the Tru64 UNIX operating system:

1. Log in as the superuser.
2. Create a mount point for the cluster documentation as follows:

```
# mkdir /usr/share/doclib/online/clusters
```

3. Insert the CD-ROM into the CD-ROM drive and mount it on a Tru64 UNIX Version 5.0 system as follows:

```
# mount -r -t cdfs -o rrip /dev/disk/cdromnc \  
/usr/share/doclib/online/clusters
```

Mount it on a pre-Tru64 UNIX Version 5.0 system as follows:

```
# mount -r -t cdfs -o rrip /dev/rznc \  
/usr/share/doclib/online/clusters
```

In these commands, *n* is the unit number of your CD-ROM device.

4. Once the CD-ROM is mounted, users can access the documentation with Netscape as follows:

- To start Netscape from a terminal emulator window, run the following command:

```
$ /usr/bin/X11/netscape &
```

- To start Netscape from the Common Desktop Environment (CDE) front panel, click Application Manager -> Desktop\_apps -> Netscape.

To view the PDF files you need to install Version 3.0 or higher of the Adobe Acrobat Reader. Versions of the Acrobat Reader for Tru64 UNIX, Windows PCs, Macintosh, and other platforms are included on the TruCluster Server Software Version 5.0 CD-ROM. You can also obtain the latest version directly from the Adobe Systems Inc. web site, <http://www.adobe.com>. With the Acrobat Reader you can scroll through books, print selected sections or the entire book, and copy sections to the clipboard.

With this release of the CD-ROM, most cross-references are hot links; you can now follow those references from book to book, from book to reference page, from reference pages to book, and from reference page to reference page. The book or reference page you are referring to opens in a separate window so you can gather the information you need, and then easily return to the book or reference you were originally reading.

## Reader's Comments

Compaq welcomes any comments and suggestions you have on this and other Tru64 UNIX manuals.

You can send your comments in the following ways:

- Fax: 603-884-0120 Attn: UBPG Publications, ZKO3-3/Y32
- Internet electronic mail: `readers_comment@zk3.dec.com`

A Reader's Comment form is located on your system in the following location:

```
/usr/doc/readers_comment.txt
```

- Mail:

Compaq Computer Corporation  
UBPG Publications Manager  
ZKO3-3/Y32  
110 Spit Brook Road  
Nashua, NH 03062-9987

A Reader's Comment form is located in the back of each printed manual. The form is postage paid if you mail it in the United States.

Please include the following information along with your comments:

- The full title of the book and the order number. (The order number is printed on the title page of this book and on its back cover.)
- The section numbers and page numbers of the information on which you are commenting.
- The version of Tru64 UNIX that you are using.
- If known, the type of processor that is running the Tru64 UNIX software.

The Tru64 UNIX Publications group cannot respond to system problems or technical support inquiries. Please address technical questions to your local system vendor or to the appropriate Compaq technical support office. Information provided with the software media explains how to send problem reports to Compaq.

## Conventions

This manual uses the following conventions:

#	A number sign represents the superuser prompt.
% <b>cat</b>	Boldface type in interactive examples indicates typed user input.
<i>file</i>	Italic (slanted) type indicates variable values, placeholders, and function argument names.
cat(1)	A cross-reference to a reference page includes the appropriate section number in parentheses. For example, <code>cat(1)</code> indicates that you can find information on the <code>cat</code> command in Section 1 of the reference pages.
Mb/s	This symbol indicates megabits per second.
MB/s	This symbol indicates megabytes per second.



---

## Introduction to TruCluster Server

TruCluster Server Version 5.0 is a highly integrated synthesis of Tru64 UNIX software, AlphaServer systems, and storage devices that operate as a single system. A TruCluster Server cluster acts as a single virtual system, even though it is made up of multiple systems. Members of the cluster can share resources, data storage, and clusterwide file systems under a single security and management domain, yet they can boot or shut down independently without disrupting the cluster.

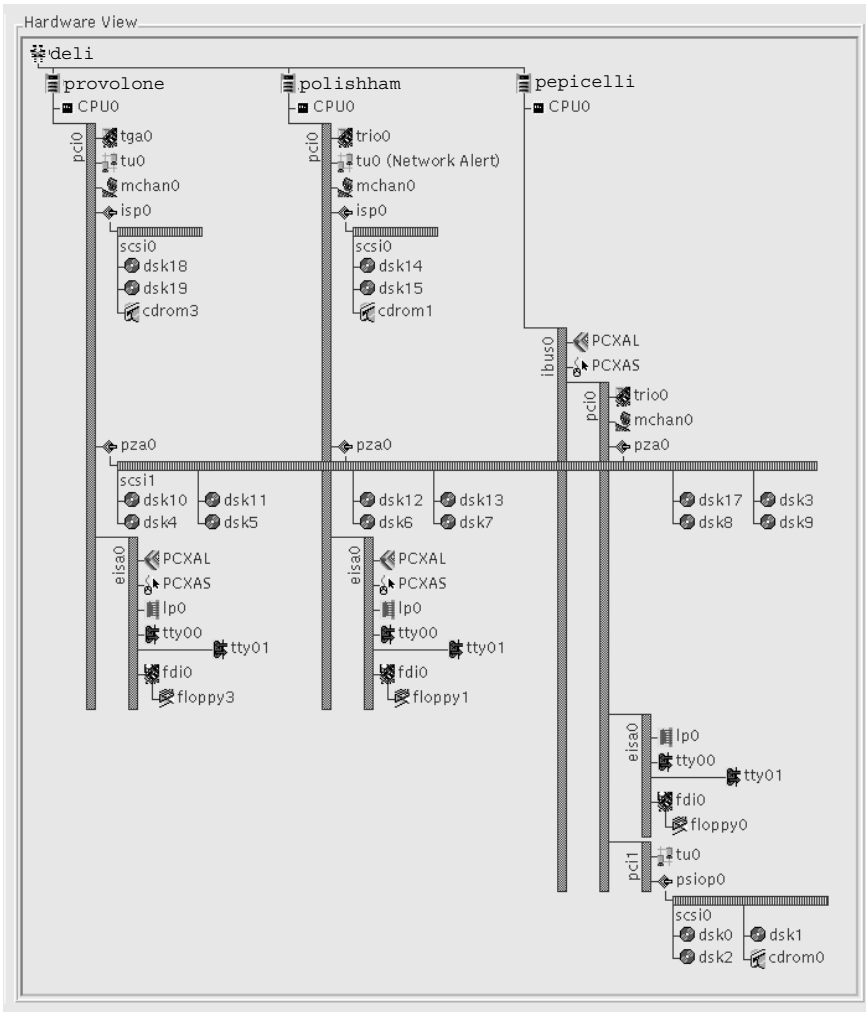
A TruCluster Server environment can be as simple or feature-rich as you require. You configure a cluster that fits your needs, from a two-node cluster up to an eight-node cluster running high availability applications such as transaction processing systems, servers for network client/server applications, data-sharing applications that require maximum uptime, and distributed parallel processing applications that take full advantage of the TruCluster Server application programming interfaces (APIs).

TruCluster Server includes a cluster alias for the Internet protocol suite (TCP/IP) so that a cluster appears as a single system to its network clients and peers.

If you know how to manage a Tru64 UNIX system, you already know how to manage a TruCluster Server cluster because TruCluster Server extends single-system management capabilities to clusters. It provides a clusterwide namespace for files and directories, including a single root (/) file system that all cluster members share. In like manner, it provides a clusterwide namespace for storage devices; each storage device has the same unique device name throughout the cluster.

The SysMan suite of graphical management utilities provides an integrated view of the cluster environment, letting you manage a single member or the entire cluster. Figure 1-1 shows the SysMan Station hardware view for a cluster named `deli`, with three members: `provolone`, `polishham`, and `pepicelli`.

**Figure 1–1: A Cluster's View of Hardware**



TruCluster Server preserves the following availability and performance features found in the TruCluster products provided for the Tru64 UNIX Version 4.0 series operating system:

- Like the TruCluster Available Server Software and TruCluster Production Server products, TruCluster Server lets you deploy highly available services that can access their disk data from any member in the cluster.

Any application that can run on Tru64 UNIX can run as a highly available single-instance application in a cluster. The application is automatically relocated (failed over) to another cluster member in the

event that a required resource, or the the current member itself, becomes unavailable.

- Like the TruCluster Production Server Software product, TruCluster Server lets you run components of distributed applications in parallel, providing high availability while taking advantage of cluster-specific synchronization mechanisms and performance optimizations.

TruCluster Server Version 5.0 provides the features listed in Table 1–1.

**Table 1–1: Features in the TruCluster Server Version 5.0 Product**

Feature	Description
Clusterwide namespace	The Cluster File System (CFS) supports a single clusterwide namespace and uniform coherent access to all file systems in a cluster. Context-dependent symbolic links (CDSLs) are used to maintain per-system configuration and data files within the shared CFS root (/), /usr, and /var file systems. See Section 2.2 for more information on CFS. See Section 2.4 for more information on CDSLs.
Clusterwide access to disk and tape storage	The device request dispatcher facility provides highly available clusterwide access to both character and block disk devices, as well as tape devices. All cluster disk and tape I/O passes through the device request dispatcher. See Section 2.3 for more information on the device request dispatcher.
Clusterwide Logical Storage Manager (LSM)	The semantics of LSM have been extended to a cluster environment. See Section 2.7 for more information on LSM in a cluster environment.
Connection manager	The connection manager ensures that all cluster members communicate with each other in order to control the formation of a cluster. The connection manager calculates the votes required for quorum and decides when members are added to and removed from the cluster. See Chapter 3 for more information on the connection manager.
Cluster application availability (CAA)	The CAA facility provides resource monitoring and application restart capabilities. It provides the same type of application availability provided by user-defined services in the TruCluster Available Server Software and TruCluster Production Server Software products. See Chapter 4 for a definition of the types of applications that can run in a cluster. See Chapter 5 for more information on CAA's role in making single-instance applications highly available.

**Table 1–1: Features in the TruCluster Server Version 5.0 Product (cont.)**

Feature	Description
Cluster alias	<p>The cluster alias subsystem lets TCP and UDP applications address the cluster as though it were a single system. When the cluster is created, a default alias is defined that addresses all cluster members. A site can define additional aliases that address some or all cluster members.</p> <p>See Chapter 6 for more information on cluster aliases.</p>
Highly available NFS server using cluster alias	<p>As shipped, the cluster is a highly available NFS server. CFS ensures that file systems exported from a TruCluster Server cluster are highly available to clients. Clients use the default cluster alias as the name of the NFS server when mounting file systems exported by the cluster.</p>
Memory Channel interconnect	<p>The Memory Channel interconnect is a high-speed interconnect designed specifically for the needs of clusters. The Memory Channel interconnect provides both broadcast and point-to-point connections between cluster members.</p> <p>TruCluster Server provides a Memory Channel application programming interface (API), which is the same as the TruCluster Production Server Software API.</p> <p>See Chapter 7 for more information on the Memory Channel interconnect. See the TruCluster Server <i>Highly Available Applications</i> manual for a description of the Memory Channel API.</p>
Distributed lock manager (DLM)	<p>TruCluster Server supports the DLM and its API, which is the same as the TruCluster Production Server Software API.</p> <p>See Chapter 8 for a description of the DLM. See the TruCluster Server <i>Highly Available Applications</i> manual for a description of the DLM API.</p>
Single-system management	<p>Because a cluster uses CFS, all systems' configuration files are available for management. The SysMan suite of graphical management utilities provides an integrated view of the cluster environment, letting you manage a single member or the entire cluster.</p> <p>See Chapter 9 for an overview of cluster installation and administration.</p>

**Table 1–1: Features in the TruCluster Server Version 5.0 Product (cont.)**

<b>Feature</b>	<b>Description</b>
Single Security Domain	Because a cluster uses CFS, there is a single copy of security administration files such as <code>/etc/passwd</code> and <code>/etc/group</code> . A user authenticated on one member has access to all members. A user with access to a file on one member has access to that file from any member. Access control lists (ACLs) are uniformly available to all members.
Expanded process IDs (PIDs)	PIDs are expanded to a full 32-bit value. PIDs are unique across a cluster. PIDs for each cluster member are based on the member ID and are allocated from a range of numbers unique to that member.



# 2

---

## Clusterwide File Systems, Storage, and Device Names

From a configuration and administration point of view, perhaps the most important new feature of TruCluster Server is the creation of a single, clusterwide namespace for files and directories. This namespace provides each cluster member with the same view of all file systems. In addition, there is a single copy of most configuration files. With few exceptions, the directory structure of a cluster is identical to that of a standalone system.

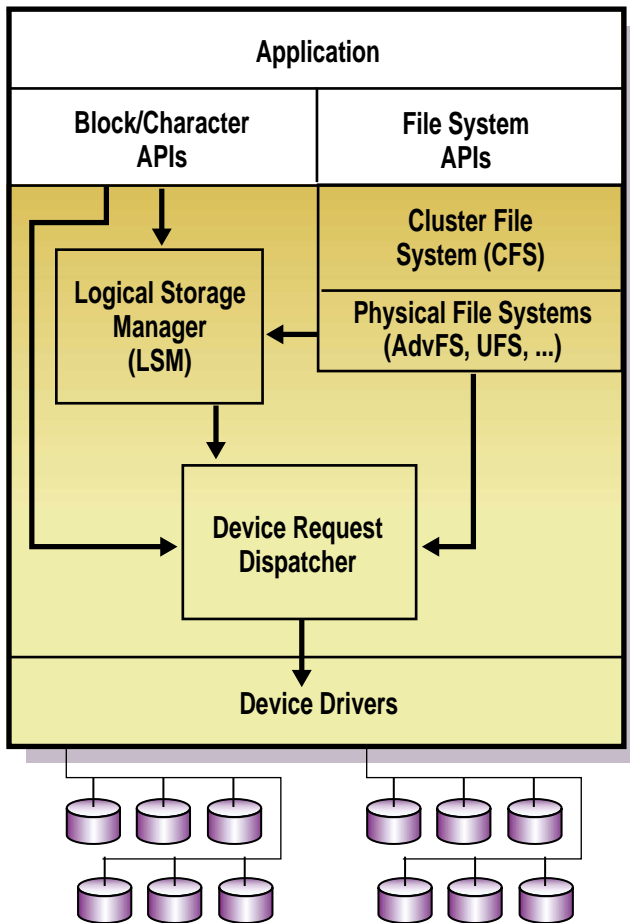
The clusterwide namespace is implemented by several new TruCluster Server technologies, including the cluster file system (CFS) and the device request dispatcher, both of which are described in this chapter.

This chapter discusses the following topics:

- Supported file systems (Section 2.1)
- Cluster File System (CFS) (Section 2.2)
- Device request dispatcher (Section 2.3)
- Context-dependent symbolic links (CDSLs) (Section 2.4)
- Device names (Section 2.5)
- Clusters and the Logical Storage Manager (LSM) (Section 2.7)

To begin to understand how storage software works in a cluster, examine Figure 2-1. This figure shows a high-level view of storage software layering in a cluster. One important thing to note in this figure is that the device request dispatcher controls all I/O to physical devices; all cluster I/O passes through this subsystem. You should also note that CFS layers on top of existing file systems such as the Advanced File System (AdvFS).

Figure 2–1: Storage Software Layering in a Cluster



ZK-1547U-AI

## 2.1 Supported File Systems

Table 2–1 summarizes how TruCluster Server supports different UNIX file systems.

**Table 2–1: UNIX File Systems Supported in a Cluster**

Type	How Supported	Failure Characteristics
Advanced File System (AdvFS)	Read/write	A file domain is served by the member that first mounts it. Upon member failure, CFS selects a new server for the domain. Upon path failure, CFS uses an alternate device request dispatcher path to the storage.
Network File System (NFS) server	Read/write	External clients use the default cluster alias as the host name when mounting file systems NFS-exported by the cluster. File system failover and recovery is transparent to external NFS clients.
NFS client	Read/write	When an file system that has been NFS-mounted in a cluster fails, the file system is automatically unmounted. The client must remount the file system to make it available. If the client uses <code>automount</code> , the remount will happen automatically.
UNIX File System (UFS)	Read-only	A file system is served for read-only access by the member that first mounts it. Upon member or path failure, CFS selects a new server for the file system. Upon path failure, CFS uses an alternate device request dispatcher path to the storage.
CD-ROM File System (CDFS)	Read-only	A file system is served for read-only access by the member that mounts the CD-ROM device. Because TruCluster Server does not support CD-ROM devices on a shared bus, a CD-ROM device becomes inaccessible to the cluster when the member to which it is locally connected fails, even if it is being served by another member. The device becomes accessible again when the member that failed rejoins the cluster.
PC-NFS server	Read/write	Clients can mount file systems NFS-exported by the cluster by using the default cluster alias. File system failover and recovery occurs transparently to external NFS clients.
Memory File System (MFS)	Not supported	

**Table 2–1: UNIX File Systems Supported in a Cluster (cont.)**

Type	How Supported	Failure Characteristics
/proc file system	Read/write (local)	Each cluster member has its own /proc file system, which is accessible only by that member.
File-on-File Mounting (FFM) file system	Read/write (local)	Can be mounted and accessed only on the local member.
Named pipes	Read/write (local)	Reader and writer must be on the same member.

## 2.2 Cluster File System

The Cluster File System (CFS) makes all files visible to and accessible by all cluster members. Each cluster member has the same view; it does not matter whether a file is stored on a device connected to all cluster members or on one that is private to a single member. By maintaining cache coherency across cluster members, CFS guarantees that all members at all times have the same view of file systems mounted in the cluster.

From the perspective of the CFS, each file system or AdvFS domain is served to the entire cluster by a single cluster member. Any cluster member can serve file systems on devices anywhere in the cluster. File systems mounted at cluster boot time are served by the first cluster member to have access to them. This means that file systems on devices on a bus private to one cluster member are served by that member. When you manually mount a file system, the cluster member that performs the mount becomes the CFS server for that file system.

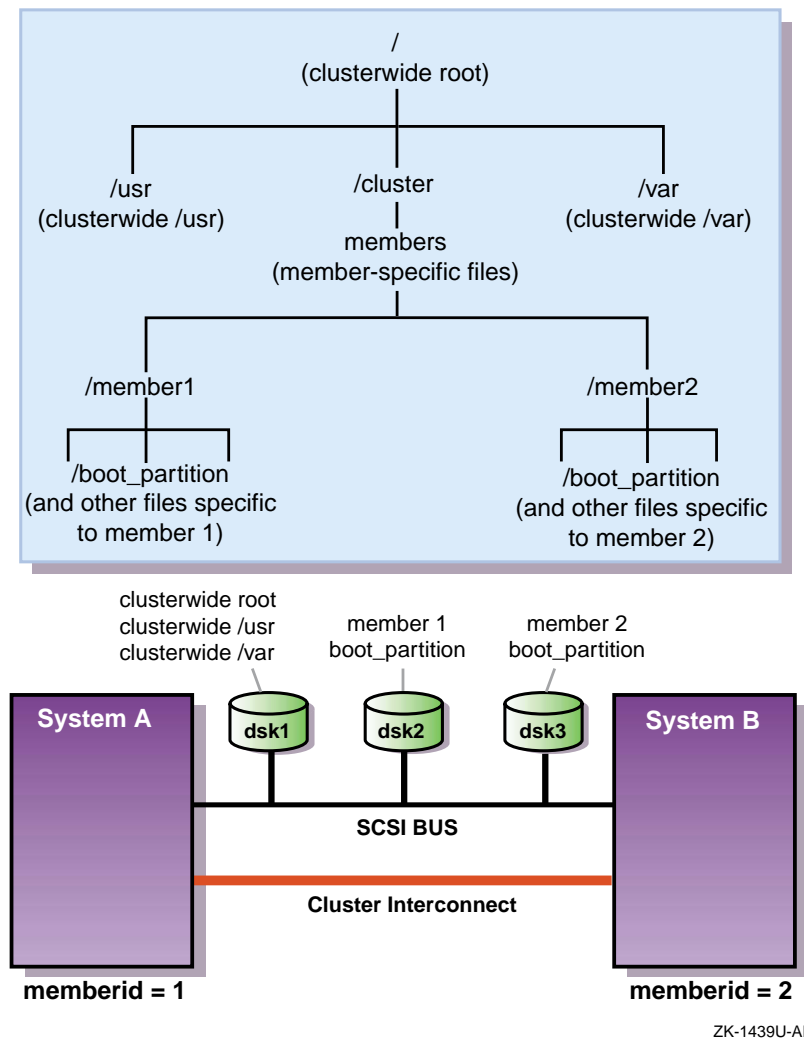
This client/server model means that a cluster member can be a client for some domains and a server for others. In addition, you can transition a member between the client/server roles. For example, if you enter the `/usr/sbin/cfsmgr` command without options, it returns the names of domains and file systems, where each is mounted, the name of the server of each, and the server status. You can use this information to relocate file systems to other CFS servers, which balances the load across the cluster.

Because CFS preserves full X/Open and POSIX semantics for file-system access, file management interfaces and utilities work in the same way they do on a standalone system.

Figure 2–2 shows the relationship between file systems contained by disks on a shared SCSI bus and the resulting cluster directory structure. Each member boots from its own boot partition, but then mounts that file system at its mount point in the clusterwide file system. Note that this figure is only an example to show how each cluster member has the same view of

file systems in a cluster. There are many physical configurations possible, and a real cluster would provide additional storage to mirror the critical root (/), /usr, and /var file systems.

**Figure 2–2: CFS Makes File Systems Available to All Cluster Members**



## 2.3 Device Request Dispatcher

In a TruCluster Server cluster, the **device request dispatcher** subsystem controls all I/O to physical devices. All cluster I/O passes through this subsystem, which enforces single-system open semantics so only one program can open a device at any one time. The device request dispatcher

makes physical disk and tape storage available to all cluster members, regardless of where the storage is physically located in the cluster. It uses the new device-naming model to make device names consistent throughout the cluster. This provides great flexibility when configuring hardware. A member does not need to be directly attached to the bus on which a disk resides to access storage on that disk.

When necessary, the device request dispatcher uses a client/server model. While CFS serves file systems and AdvFS domains, the device request dispatcher serves devices, such as disks, tapes, and CD-ROM drives. However, unlike the client/server model of CFS in which each file system or AdvFS domain is served to the entire cluster by a single cluster member, the device request dispatcher supports the notion of many simultaneous servers.

In the device request dispatcher model, devices in a cluster are either single-served or direct-access I/O devices. A single-served device, such as a tape device, supports access from only a single member, the server of that device. A direct-access I/O device supports simultaneous access from multiple cluster members. Direct-access I/O devices on a shared bus are served by all cluster members on that bus. You can use the `drdmgr` command to check the device request dispatcher view of a device.

In the following example, device `dsk17` is on a shared bus, and is served by three cluster members.

```
# /usr/sbin/drddmgr dsk17

View of Data from Node polishham as of 1999-06-04:16:03:46

Device Name: dsk17
Device Type: Direct Access IO Disk
Device Status: OK
Number of Servers: 3
  Server Name: provolone
  Server State: Server
  Server Name: polishham
  Server State: Server
  Server Name: pepicelli
  Server State: Server
Access Node Name: polishham
Open Partition Mask: 0
Statistics for Client Node: polishham
Number of Read Operations: 0
Number of Write Operations: 0
Number of Bytes Read: 0
Number of Bytes Written: 0
```

The device request dispatcher supports clusterwide access to both character and block disk devices. You access a raw disk device partition in a TruCluster Server configuration in the same way you do on a Tru64 UNIX Version 5.0 standalone system; that is, by using the device's special file name in the `/dev/rdisk` directory.

---

### Note

---

Before TruCluster Server Version 5.0, cluster administrators had to define special Distributed Raw Disk (DRD) services to provide this level of physical access to storage; with TruCluster Server Version 5.0 this access is built into the cluster architecture and is automatically available to all cluster members.

---

## 2.4 Context-Dependent Symbolic Link

Although the single namespace greatly simplifies system management, there are some configuration files and directories that should not be shared by all cluster members. For example, a member's `/etc/sysconfigtab` contains information about that system's kernel component configuration, and only that system should use that configuration. Consequently, the cluster must employ a mechanism that lets each member read and write the file named `/etc/sysconfigtab`, while actually reading and writing its own member-specific `sysconfigtab` file.

Tru64 UNIX Version 5.0 introduces a special form of a symbolic link called a **context-dependent symbolic link (CDSL)** that TruCluster Server uses to create a namespace with these characteristics. CDSLs allow a file or directory to be accessed by a single name, regardless of whether the name represents a clusterwide file or directory or a member-specific file or directory. CDSLs keep traditional naming conventions while providing the behind-the-scenes sleight of hand needed to make sure each member reads/writes its own copy of member-specific system configuration files.

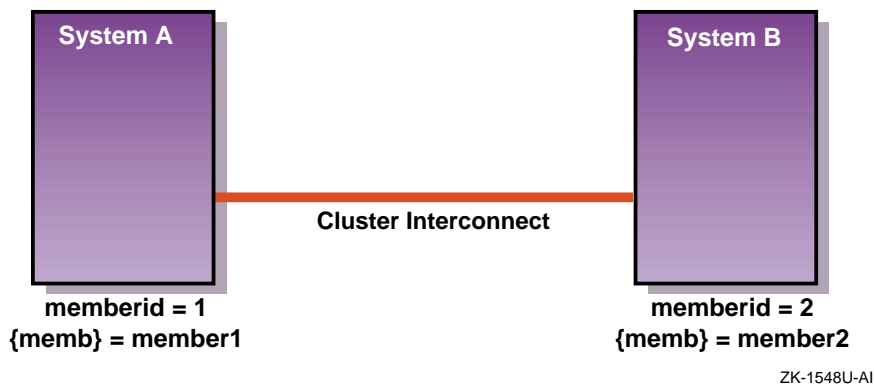
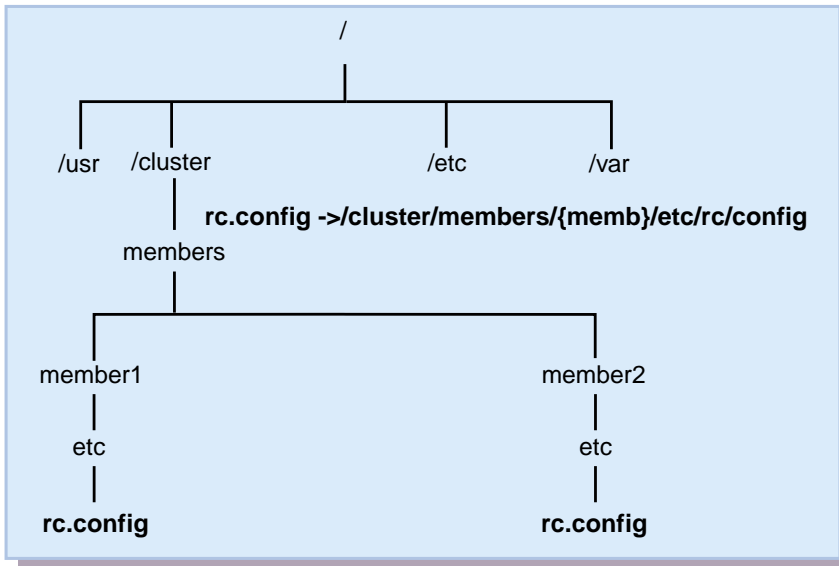
CDSLs contain a variable whose value is determined only during pathname resolution. The `{memb}` variable is used to access member-specific files in a cluster. The following example shows the CDSL for `/etc/rc.config`:

```
/etc/rc.config -> ../cluster/members/{memb}/etc/rc.config
```

CDSLs are useful when running multiple instances of an application on different cluster members, on different sets of data. The TruCluster Server *Highly Available Applications* manual describes how applications can use CDSLs to maintain member-specific data sets and log files.

When resolving a CDSL pathname, the kernel replaces the `{memb}` variable with the string `membern`, where `n` is the member ID of the current member. Therefore, on a cluster member whose member ID is 2, the pathname `/cluster/members/{memb}/etc/rc.config` resolves to `/cluster/members/member2/etc/rc.config`. Figure 2-3 shows the relationship between `{memb}` and CDSL pathname resolution.

Figure 2–3: CDSL Pathname Resolution



ZK-1548U-AI

As a general rule, before you move a file or directory, make sure that the destination is not a CDSL. Moving files to CDSLs requires special care on your part to ensure that the member-specific files are maintained. For example, consider the file `/vmunix` as shown in the following example.

```
/vmunix -> cluster/members/{memb}/boot_partition/vmunix
```

If you were to move (instead of copy) a kernel to `/vmunix`, you would replace the symbolic link with the actual file; `/vmunix` would no longer be a symbolic link to `/cluster/members/{memb}/boot_partition/vmunix`.

The `mkcdsl` command lets system administrators create CDSLs and update a CDSL inventory file. The `cdslinvchk` command verifies the current CDSL inventory. For more information on these commands, see `mkcdsl(8)` and `cdslinvchk(8)`.

For more information about CDSLs, see the Tru64 UNIX *System Administration* manual, `hier(5)`, `ln(1)`, and `symlink(2)`.

## 2.5 Device Names

This section provides an introduction to the new device-naming model introduced in Tru64 UNIX Version 5.0. For a detailed discussion of this new device-naming model, see the Tru64 UNIX *System Administration* manual.

Device names are consistent clusterwide; they are:

- Persistent beyond boot
- Stay with the device even when you move a disk or tape to a new location in the cluster

---

### Note

---

Although Tru64 UNIX Version 5.0 supports the old-style device names as a compatibility option, TruCluster Server Version 5.0 supports only the new-style names. Applications that depend on old-style device names (or the structure of `/dev`) must be modified to use the new device-naming model.

---

Previously, device names were determined by the position of the I/O controller on the system bus, the position of the device on the I/O bus, and the device's **logical unit number (LUN)**. Starting with Tru64 UNIX Version 5.0, device names are established when the operating system first discovers the device (for example, at initial system boot time or when the device is first added), and these names are independent of the physical configuration and convey no information about the architecture or logical path.

For example, prior to Tru64 UNIX Version 5.0, disks were named as follows:

- `/dev/rz2`
- `/dev/rz3`
- `/dev/rz4`

This naming had encoded within it the bus and LUN of the **SCSI** disk. For example, disk 2 in bus 0 was `rz2`; disk 0 in bus 1 was `rz8`; disk 0 in bus 2 was `rz16`, disk 3 in bus 2 was `rz19`, and so on.

The new device-name convention consists of a descriptive name for the device and an automatically assigned instance number. These two elements form the base name of the device, such as `dsk0`. Note that the instance number in a device's new name does not correlate to the unit number in its old name: the operating system assigns the instance numbers in sequential order, beginning with 0 (zero), as it discovers devices.

Table 2-2 shows some examples of new device names.

**Table 2-2: Examples of New Device Names**

Old Name	New Name	Description
<code>/dev/rz4c</code>	<code>/dev/disk/dsk4c</code>	c partition of the fifth disk recognized by the operating system
<code>/dev/rz19c</code>	<code>/dev/disk/dsk5c</code>	c partition of the sixth disk recognized by the operating system

The suffix assigned to the device name special files differs depending on the type of device, as follows:

- **Disks** — In general, disk device file names consist of the base name and a one-letter suffix from a through z; for example, `/dev/disk/dsk0a`. Disks use a through h to identify partitions. By default, floppy disk and CD-ROM devices use only the letters a and c; for example, `floppy0a` and `cdrom1c`.

For raw device names, the same device names exist in the class directory `/dev/rdisk`.

- **Tapes** — These device file names have the base name and a suffix comprised of the characters `_d` followed by a single digit; for example, `tape0_d0`. This suffix determines the density of the tape device, according to the entry for the device in the `/etc/ldr.dbase` file; for example:

Device	Density
<code>tape0</code>	default density
<code>tape0c</code>	default density with compression
<code>tape0_d0</code>	density associated with entry 0 in <code>/etc/ldr.dbase</code>
<code>tape0_d1</code>	density associated with entry 1 in <code>/etc/ldr.dbase</code>

Note that with the new device special file naming, there is a direct mapping from the old name suffix to the new name suffix, as follows:

Old Suffix	New Suffix
l (low)	_d0
m (medium)	_d2
h (high)	_d1
a (alternative)	_d3

There are two sets of device names for tapes that both conform to the new naming convention. The `/dev/tape` directory for rewind devices and the `/dev/ntape` directory for no-rewind devices. To determine which device special file to use, you can look in the `/etc/DDR.dbase` file.

Tru64 UNIX provides utilities to identify device names. Use the following commands to list a system's devices:

Command	Description	For More Information
<code>hwmgr -view hierarchy</code> or <code>hwmgr -view devices</code>	Lists a member's hardware configuration and correlates bus-target-LUN names with <code>/dev/disks/dsk</code> names.	<code>hwmgr(8)</code>
<code>dsfmgr -s -x</code>	Manages device special files using the Tru64 UNIX Version 5.0 file naming format. Displays detailed information from the database when the <code>-x</code> flag is used.	<code>dsfmgr(8)</code>

---

#### Note

---

Note that Logical Storage Manager (LSM) naming conventions have not changed in Tru64 UNIX Version 5.0.

---

## 2.6 Worldwide ID

Tru64 UNIX associates the new device name with the **worldwide ID (WWID)** of a disk. A disk's WWID is unique and is set by the manufacturers for devices that support it. Therefore, no two disks can have the same WWID. Using the WWID to identify a disk has two implications. Once a disk is recognized by the operating system, the disk's `/dev/disk/dsk` name will stay the same even if its SCSI address changes.

This also allows Tru64 UNIX to support multipathing to a disk where the disk is accessible through different SCSI controllers. If disks are moved within a TruCluster Server environment, their device names and how users access them will remain the same.

---

**Note**

---

The names of disks behind RAID controllers are associated with both the WWID of their controller module and their own bus, target, and LUN position. When they are moved, they do not retain their disk names. However, you can use the `hwmgr` utility to reassociate such a disk with its previous device name.

---

The following `hwmgr` command displays the WWIDs for a system or cluster:

```
# /sbin/hwmgr -get attr -a name
```

## 2.7 Clusters and the Logical Storage Manager

The **Logical Storage Manager (LSM)** provides shared access to all LSM volumes from any cluster member. LSM consists of physical disk devices, logical entities, and the mappings that connect both. LSM builds virtual disks, called volumes, on top of UNIX physical disks. LSM transparently places a volume between a physical disk and an application, which then operates on the volume rather than on the physical disk. For example, you can create a file system on an LSM volume rather than on a physical disk.

As previously shown in Figure 2-1, LSM is layered on top of the device request dispatcher. Using LSM in a cluster is like using LSM in a single system. The same LSM software subsets are used for both clusters and noncluster configurations, and you can make configuration changes from any cluster member. LSM keeps the configuration state consistent clusterwide.

Note that there are some points to keep in mind when using LSM in a cluster. See the TruCluster Server *Cluster Administration* manual for configuration and usage issues that are specific to LSM in a TruCluster Server environment.

---

## Connection Manager

Clustered systems share various data and system resources, such as access to disks and files. To achieve the coordination that is necessary to maintain resource integrity, the cluster must have clear criteria for membership and must disallow participation in the cluster by systems that do not meet that criteria.

This chapter provides the following information:

- An overview of the connection manager functions (Section 3.1)
- A discussion of vote types (Section 3.2)
- A discussion of how the connection manager calculates quorum (Section 3.3)
- Using a quorum disk (Section 3.4)
- Cluster partitions (Section 3.5)
- Monitoring the connection manager (Section 3.6)

### 3.1 Connection Manager

The **connection manager** is a distributed kernel component that ensures that cluster members communicate with each other and enforces the rules of cluster membership. The connection manager:

- Adds members to a cluster and removes members from a cluster
- Tracks which members in a cluster are active and which are not
- Maintains a cluster membership list that is consistent on all cluster members
- Provides timely notice of membership changes using Event Manager (EVM) events
- Detects and handles possible cluster partitions (see Section 3.5)

An instance of the connection manager runs on each cluster member. These instances maintain contact with each other, sharing information such as the cluster's membership list. The connection manager uses a three-phase commit protocol to ensure that all members have a consistent view of the cluster.

## 3.2 Votes

The connection manager allows processing and I/O to occur only when a majority of **votes** are present in the cluster. Votes are contributed to the cluster by cluster members and by the **quorum disk** if one is configured (see Section 3.4).

### 3.2.1 Member Votes

**Member votes** are the fixed number of votes that a given member contributes towards quorum. Cluster members can have either 1 or 0 (zero) member votes. Each member with a vote is considered to be a **voting member** of the cluster. A member with 0 (zero) votes is considered to be a **nonvoting member**.

Voting members can form a cluster. Nonvoting members can only join an existing cluster. Votes are assigned to members during cluster configuration. The `clu_quorum` command lets administrators display and modify member vote settings.

A member's votes are recorded in the `cluster_node_votes` kernel attribute in the `clubase` subsystem of its member-specific `etc/sysconfigtab` file.

### 3.2.2 Quorum Disk Votes

In certain cluster configurations (see Section 3.4), cluster availability is improved by configuring a quorum disk. **Quorum disk votes** are the fixed number of votes that a quorum disk contributes towards quorum. A quorum disk can have either 1 or 0 (zero) votes (a quorum disk configured at cluster creation is given 1 vote by default).

Quorum disk votes are recorded in the `cluster_qdisk_votes` kernel attribute in the `clubase` subsystem of each member's `etc/sysconfigtab` file.

### 3.2.3 Member-Specific Cluster Expected Votes

**Expected votes** are the number of votes the connection manager should expect when all cluster members are up and running and any quorum disk is configured. In other words, expected votes are the sum of all member votes held by cluster members, plus the vote of the quorum disk (if one is defined). Each member brings its own notion of expected votes to the cluster; it is important that all members agree on the same number of expected votes. The connection manager refers to the expected votes settings of booting cluster members to establish its own internal

clusterwide notion of expected votes, often referred to as **cluster expected votes**. The connection manager uses its cluster expected votes value when determining the number of votes the cluster requires to maintain quorum, as explained in Section 3.3.

The `clu_create` and `clu_add_member` commands automatically adjust each member's expected votes as a new voting member or quorum disk is configured in the cluster. The `clu_delete_member` command automatically lowers expected votes when a member is deleted. Similarly, the `clu_quorum` command adjusts each member's expected votes as a quorum disk is added or deleted, or member votes are assigned to or removed from a member. These commands ensure that the member-specific expected votes is the same on each cluster member and is the sum of all member votes and the quorum disk vote (if a quorum disk is configured).

A member's expected votes are recorded in the `cluster_expected_votes` kernel attribute in the `clubase` subsystem of its member-specific `etc/sysconfigtab` file.

### 3.3 Calculating Cluster Quorum

The connection manager detects and manages cluster partitions by employing a mechanism known as a **quorum algorithm**. The quorum algorithm is a mathematical method that the connection manager uses to determine the circumstances under which a given member can participate in a cluster, safely accessing clusterwide resources and performing useful work. The algorithm operates dynamically; that is, cluster events trigger its calculations, and the results of its calculations can change over the lifetime of a cluster.

The quorum algorithm operates as follows:

1. The connection manager selects a set of cluster members upon which it bases its calculations. This set includes all members with which it can communicate. For example, it does not include members that have not yet booted, members that are down, or members that it cannot reach due to a hardware failure (for example, a detached cluster interconnect cable or a bad Memory Channel adapter).
2. Each time a member boots and joins the cluster, or `clu_delete_member` deletes a member from a cluster, the connection manager calculates a value for cluster expected votes using the *largest* of the following values:
  - The maximum member-specific expected votes value from the set of proposed members selected in step 1.

- The sum of the member-specific member votes values from the set of proposed members selected in step 1, plus the quorum disk vote if a quorum disk is configured.
- The previous cluster expected votes value.

Consider a three-member cluster with no quorum disk. All members are up and fully connected; each member has one vote and has its member-specific expected votes set to 3. Cluster expected votes is currently 3.

If a new member with 1 vote boots, the connection manager calculates cluster expected votes as 4, the sum of the member votes in the cluster.

3. Whenever the connection manager recalculates cluster expected votes (or resets cluster expected votes as the result of a `clu_quorum -e` command), it calculates a value for quorum votes.

**Quorum votes** is a dynamically calculated clusterwide value, based on the value of cluster expected votes, that determines whether a given member can form, join, or continue to participate in a cluster. The connection manager computes the clusterwide quorum votes value using the following formula:

```
quorum votes = round_down((cluster-expected-votes+2)/2)
```

For example, consider the three-member cluster described in step 2. With cluster expected votes set to 3, quorum votes would be calculated as  $\text{round\_down}(3+2)/2$ , or 2. In the case where the fourth member was added successfully, quorum votes would be calculated as  $\text{round\_down}(4+2)/2$ , or 3.

---

#### Note

---

When a member is shut down, or goes down for any other reason, the connection manager does not decrease the value of quorum votes. Under normal cluster operation, the connection manager never decreases the quorum votes value; it only increases it. Only member deletion can lower the quorum votes value of a running cluster.

---

4. Whenever a cluster member senses that the number of votes it can see has changed (a member has joined the cluster, an existing member has been removed from the cluster, or a communications error is reported), it compares the number of votes it can see (**current votes** as displayed by the `clu_quorum` or `clu_get_info -full` command when issued on that member) to the clusterwide quorum votes value.

The action the member takes is based on the following conditions:

- If the value of current votes is greater than or equal to quorum votes, the member continues running or resumes (if it had been in a suspended state).
- If the value of current votes is less than quorum votes, the member suspends all process activity, all I/O operations to cluster-accessible storage, and all operations across network external to the cluster until sufficient votes are added (that is, enough members have joined the cluster or the communications problem is mended) to bring current votes to a value greater than or equal to quorum.

As a result of the quorum loss, the connection manager initiates a reconfiguration of the cluster membership. It may remove members from the cluster as it strives to reform the cluster within the strictures of the quorum algorithm.

Note that the comparison of current votes to quorum votes occurs on a member-by-member basis, although events may make it appear that quorum loss is a clusterwide event. When a cluster member loses quorum, all of its I/O is suspended and all network interfaces except the Memory Channel interfaces are turned off. No commands that must access a clusterwide resource work on that member. It may appear to be hung.

Depending upon how the member lost quorum, an administrator might remedy the situation by restoring communications between this member and the rest of the cluster, issuing a `clu_quorum` command on a surviving member, or booting a member with enough votes for the member in quorum loss to achieve quorum. If all cluster members have lost quorum, the options are limited to booting a new member with sufficient votes for the members in quorum loss to achieve quorum, rebooting the entire cluster, or resorting to the detailed procedures described in the TruCluster Server *Cluster Administration* manual.

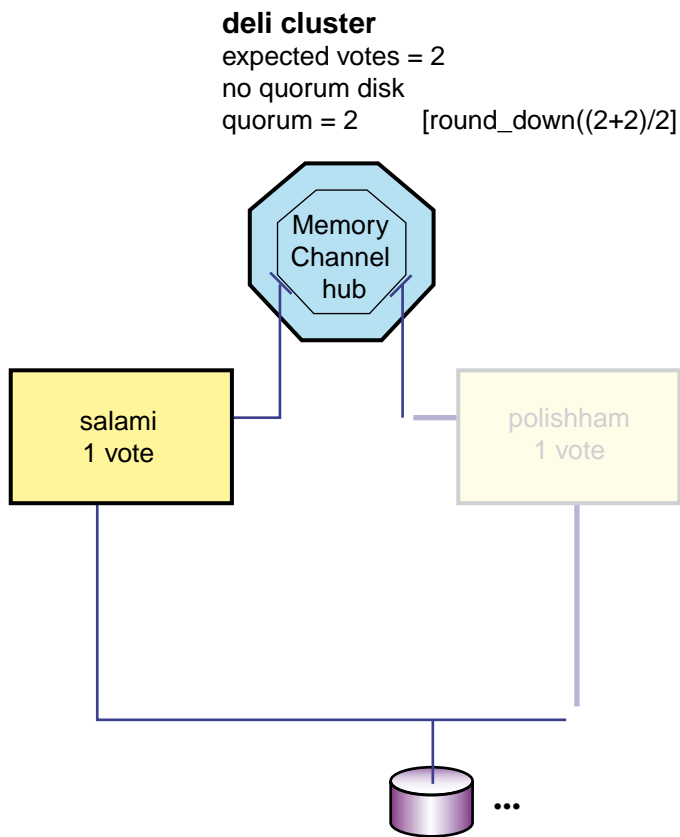
### 3.4 Using a Quorum Disk

In a two-member cluster configuration, where there are two expected votes and each member has one member vote, the loss of a single member will cause the cluster to lose quorum and all applications to be suspended. This type of configuration is not highly available.

To foster better availability in such a configuration, an administrator can designate a disk on the shared bus as a quorum disk. The quorum disk acts as a virtual cluster member whose purpose is to add one vote to the total number of expected votes. When a quorum disk is configured in a two-member cluster, the cluster can survive the failure of either the quorum disk or one member and continue operating.

For example, consider the two-member `deli` cluster without a quorum disk as shown in Figure 3-1.

**Figure 3-1: Two-Member `deli` Cluster Without a Quorum Disk**



ZK-1569U-AI

Cluster expected votes is 2; each member contributes 1 member vote. The connection manager calculates quorum votes as follows:

```
quorum votes = round_down((cluster-expected-votes+2)/2)
quorum votes = round_down((2+2)/2)
quorum votes = 2
```

The failure or shutdown of member `polishham` causes member `salami` to lose quorum. Cluster operations are suspended.

However, if the cluster includes a quorum disk (adding one vote to the total of cluster expected votes), the quorum votes value will remain at 2:

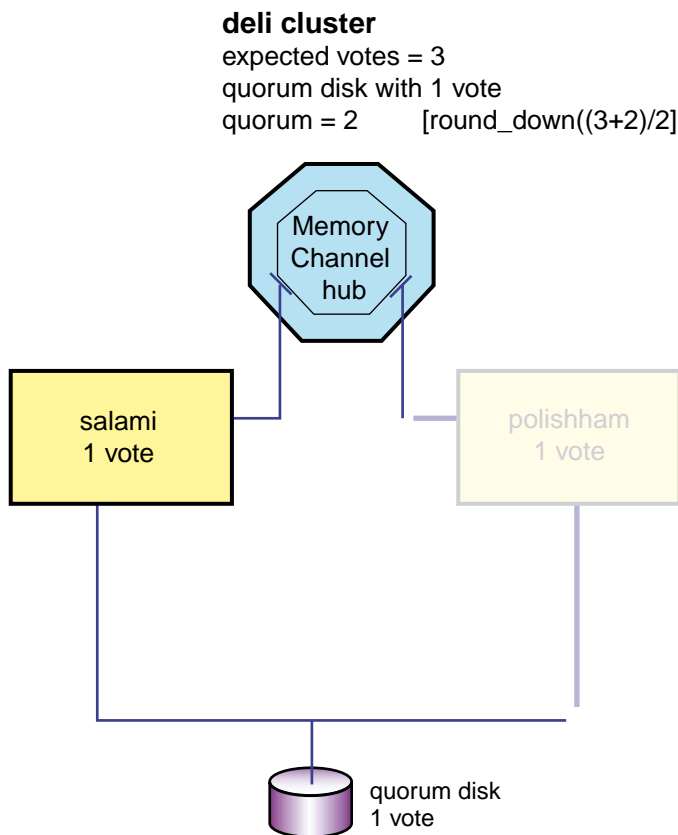
```

quorum votes = round_down((cluster-expected-votes+2)/2)
quorum votes = round_down((3+2)/2)
quorum votes = 2

```

Even if one member or the quorum disk left the cluster, sufficient node votes will remain to keep the cluster from losing quorum. The cluster, shown in Figure 3–2 can continue operation.

**Figure 3–2: Two-Member deli Cluster with Quorum Disk Survives Member Loss**



ZK-1575U-AI

The following restrictions apply to the use of a quorum disk:

- A cluster can have only one quorum disk.
- The quorum disk should be on a shared bus to which all cluster members are directly connected. If it is not, members that do not have a direct connection to the quorum disk may lose quorum before members that do have a direct connection to it.

- The quorum disk must not contain any data. The `clu_quorum` command will overwrite existing data when initializing the quorum disk. The integrity of data (or file system metadata) placed on the quorum disk from a running cluster is not guaranteed across member failures.

This means that the member boot disks and the disk holding the clusterwide root (/) cannot be used as quorum disks.

- The quorum disk can be quite small. The cluster subsystems use only 1 MB of the disk.
- A quorum disk can have either 1 vote or no votes. In general, a quorum disk should always be assigned a vote. You might assign an existing quorum disk no votes in certain testing or transitory configurations, such as a one-member cluster (in which a voting quorum disk introduces a second point of failure).
- You cannot use LSM on the quorum disk.

## 3.5 Cluster Partitions

A primary purpose of the connection manager is to detect cluster partitions and resolve them in a manner that maintains cluster operation and the integrity of shared data. A **cluster partition** is a situation in which an existing cluster can divide into two or more clusters. It can result from broken or disconnected hardware; for example, disconnected Memory Channel cables. The connection manager and its quorum algorithm, discussed in Section 3.3, handles cluster partitioning by placing members that have lost quorum into suspension and by reconfiguring the cluster, if possible, around the remaining members that can maintain quorum.

A more serious sort of partition is one in which members on either side of the partition are able to maintain quorum and form separate clusters. Neither cluster is aware of the other, so they cannot synchronize their activities. Members of either cluster can access data and system files on mutually shared storage without any coordination across the clusters. This can result in data loss and corruption.

Fortunately, the connection manager makes it impossible for such a hazardous partition to evolve from a running cluster, even one with misconfigured member vote values. In fact, it would take multiple failures, plus misconfigured member vote values, plus member reboots to risk such partitioning. Here, too, the recommended interfaces for setting and managing member votes and the quorum disk (`clu_create`, `clu_add_member`, `clu_delete_member`, and `clu_quorum`) prevent an administrator from misconfiguring vote values. This is one important reason to avoid manually editing any member's `sysconfigtab` file.

The TruCluster Server *Cluster Administration* manual provides detailed information on managing the votes in a cluster, and includes troubleshooting information for administrators who inadvertently set votes that result in reducing the availability of the cluster.

### 3.6 Monitoring the Connection Manager

The connection manager provides several kinds of output for administrators.

It posts Event Manager (EVM) events are posted for four types of event:

- Node joining cluster
- Node removed from cluster
- Quorum disk becoming unavailable (due to error, removal, and so on)
- Quorum disk becoming available again

Each of these events also results in console message output.

The connection manager prints various informational messages to the console during member boots and cluster transactions.

A cluster transaction is the mechanism for modifying some clusterwide state on all cluster members atomically; either all members adopt the new value or none do. The most common transactions are membership transactions, such as when the cluster is formed, members join, or members leave. Certain maintenance tasks also result in cluster transactions, such as the addition or removal of a quorum disk or the modification of the clusterwide expected votes value.

Cluster transactions are global (clusterwide) occurrences. Console messages are also printed on the console of an individual member in response to certain local events, such as when the connection manager notices a change in connectivity (to another node or to the quorum disk), or when it gains or loses quorum.



---

## Highly Available Applications

Applications on clusters can be divided into three basic types:

**single-instance application**

A single-instance application runs on only one cluster member at a time. In order to make this type of application highly available, the cluster must provide a mechanism for starting the application on another cluster member in the event that the current member can no longer run the application. The TruCluster Server high availability mechanism for single-instance applications is the cluster application availability (CAA) subsystem; see Chapter 5 for a description of CAA.

The TruCluster Server *Highly Available Applications* manual provides detailed information about moving applications from an earlier TruCluster product to a Version 5.0 TruCluster Server.

**multi-instance application**

A multi-instance application can run on multiple cluster members at the same time. A multi-instance application by definition is **highly available** because the failure of one cluster member does not affect the instances of the application running on other members. See Chapter 6 for a discussion of how cluster aliases provide transparent client access to multi-instance applications.

**distributed application**

A distributed application is specifically designed to run on a cluster, using different members for specific purposes. These applications use the Memory Channel, distributed lock manager (DLM), and cluster alias application programming interfaces to integrate application with the cluster resources.

TruCluster Server lets you run components of distributed applications in parallel, providing high availability while taking advantage of

cluster-specific synchronization mechanisms and performance optimizations.

See Chapter 6, Chapter 7, Chapter 8, and the TruCluster Server *Highly Available Applications* manual for more information on the subsystems and interfaces used to create distributed applications.

# 5

---

## Cluster Application Availability

This chapter provides the following information:

- A general overview of the cluster application availability (CAA) subsystem (Section 5.1)
- An introduction to CAA resources (Section 5.3)
- A description of resource profiles and their use (Section 5.4)
- A description of the action scripts used by CAA commands to manage applications and other resources (Section 5.5)

### 5.1 Overview

The cluster application availability (CAA) subsystem provides high availability for single-instance applications and monitoring of the state of other types of resources (such as network interfaces). A single instance of any application that can run on Tru64 UNIX can be made highly available in a cluster with CAA. The application runs on a single member of a cluster and cannot be run on more than one member at a time. (In a cluster, the daemons for BIND (`named`), DHCP (`joind`), and network locking (`rpc.lockd`) are managed by CAA.)

CAA can automatically relocate (fail over) an application to another cluster member in the event that a required resource, or the current member itself, becomes unavailable. This feature requires no changes to the application itself, and can be used with any single-instance application, that is, with any application that runs on only a single member of a cluster.

Each application under CAA control has a resource profile, which contains that application's resource requirements. CAA monitors the state of cluster members and resources to ensure that each application runs on a member that meets its resource requirements.

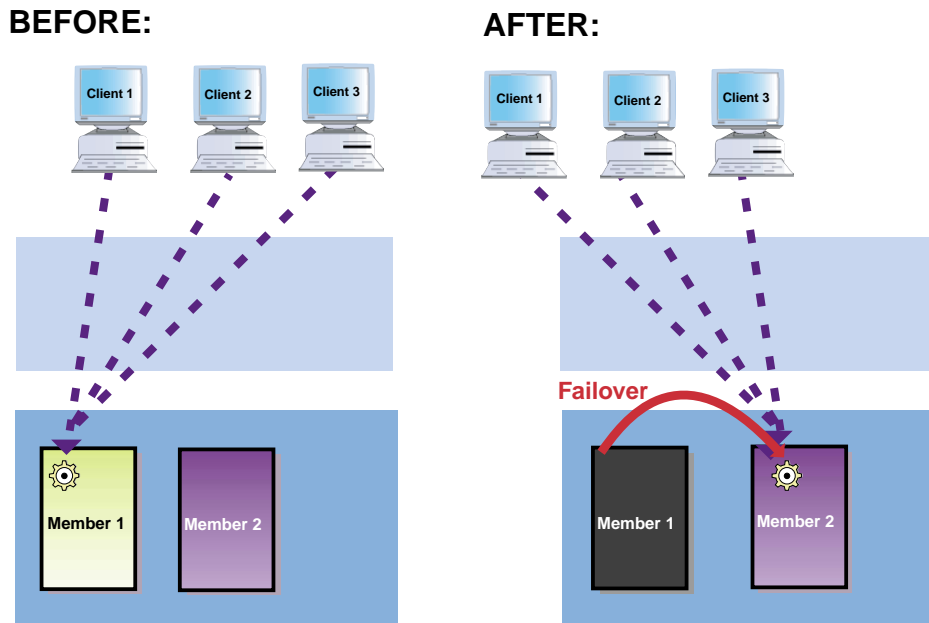
Cluster administrators use the command line interface or the graphical user interface to define resources and dependencies among them. An administrator can use these dependencies to determine where an application can run and the circumstances under which it is failed over to another cluster member.

**Note**

CAA's resource monitoring and application restart capabilities provide the same type of application availability provided by user-defined available server environment (ASE) services in previous cluster products.

Figure 5–1 shows how the failure of one member results in the failover of an application to the second member. If clients access the application through a cluster alias, the cluster alias subsystem automatically forwards connection requests to the second member.

**Figure 5–1: Application Failover with CAA**



 **Single Instance Service**

ZK-1446U-AI

## 5.2 CAA Architecture

The CAA subsystem consists of the following components:

resource manager

The resource manager consists of all the CAA daemons running on cluster members. Each CAA daemon (*caad*) starts, stops, relocates, and restarts

application resources when a required resource or cluster member fails. Each cluster member runs a CAA daemon. These daemons are independent but they communicate with each other, sharing information about the status of the resources.

The resource manager communicates with all the components of the CAA subsystem as well as the connection manager and the event manager (EVM).

#### command-line interface

The CAA subsystem provides the `caa_register`, `caa_start`, `caa_stop`, `caa_unregister`, `caa_relocate` and `caa_stat` commands to manage and monitor resources. See `caa(4)` for a list of all CAA reference pages.

The command-line interface interacts with resource profiles, action scripts and the resource manager.

#### graphical user interface

The SysMan Menu provides a graphical user interface (GUI) to perform system management tasks for the cluster, cluster members, and CAA applications. For more information on using the graphical user interface (GUI) for performing system management tasks for CAA applications, see `sysman(8)` and the online help for the SysMan Menu.

The CAA graphical user interface uses the command-line interface to interact with the resource manager, action scripts, and resource profiles.

#### resource profile

Resource profiles contain the information needed to monitor resources and control application failover. The `caa_profile` command and SysMan can create resource profiles, or they can be created in any text editor. Profiles that are created or modified using a text editor should be validated using `caa_profile -validate`.

Resource profiles are located in the `/var/cluster/caa/profile` directory. The file names of resource profiles take the form `resource_name.cap`.

A resource profile contains keyword/value attributes that define a resource, its dependencies, and how

the resource is managed by CAA. A resource profile is used by the command-line interface and resource manager.

#### action script

An action script is a set of commands used by CAA to start and stop an application. An action script is used by the command-line interface, and can be created and updated by the command-line interface, graphical user interface, or a text editor.

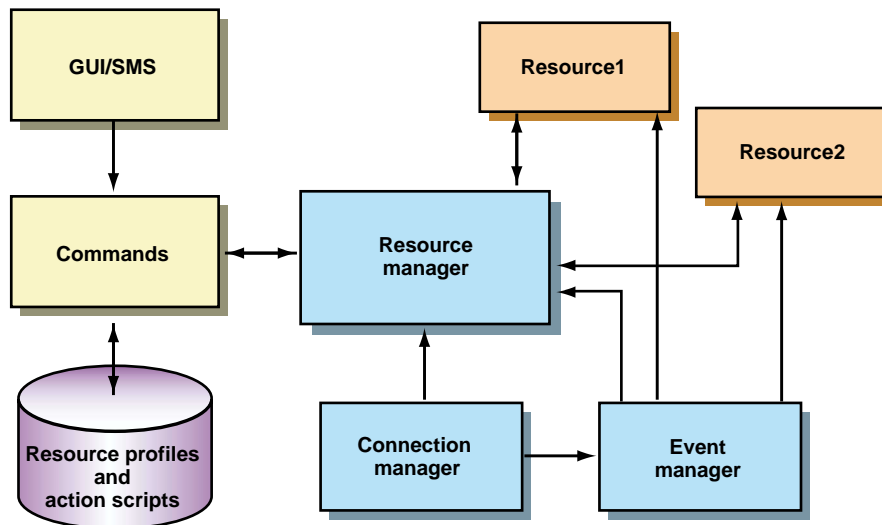
Action scripts control how applications are started, stopped, and probed. The name of an application's action script is defined in that application's resource profile.

Action scripts are located in the `/var/cluster/caa/script` directory. The file names of action scripts take the form `resource_name.scr`.

Although the connection manager and event manager are not part of the CAA subsystem, the subsystem makes extensive use of these facilities.

Figure 5-2 shows a graphical representation of the CAA architecture.

**Figure 5-2: CAA Architecture**



ZK-1585U-AI

## 5.3 Introduction to Resources

A **resource** is a cluster hardware or software component that provides a service to end users or to other software components. Examples of resources are disks, tapes, file systems, network interfaces, and application software. Currently supported resources are applications and network interfaces. Resources may depend on other resources in order to be started or to continue running.

An application resource can have other resources defined as required or optional dependencies. How this resource is managed by CAA depends on these required or optional resources. When a resource that an application requires becomes unavailable, CAA stops the application. CAA attempts to restart the application on another member. If CAA is unable to restart the application on another member because the other member is down or the placement policy forbids starting the application on that member, the application is stopped.

Optional resources are used in conjunction with required resources to help determine the optimal system on which to start an application. If an optional resource becomes unavailable the application does not fail over.

---

### Restrictions

---

Optional resources are not yet supported. Application resources cannot be specified as either required or optional resources.

---

All members in a TruCluster cluster can indirectly access any network attached to any member. Defining a network resource is useful when you want an application to run on a member with direct connectivity to the network. Use network resources as dependencies for those applications that require somewhat better performance from direct connectivity.

When you specify a network resource an optional resource for an application, CAA will start the application on a member that is directly connected to the subnet. If the subnet adapter fails, the application reverts to accessing the subnet remotely through routing. If you specify a network resource as a required resource and the network interface adapter fails, CAA relocates or stops the application if it cannot relocate the resource.

## 5.4 Resource Profiles

A **resource profile** is a text file containing a list of keyword/value pairs. These attributes define a resource, its dependencies, and how the resource is managed by CAA. After a resource profile is created, the profile must be registered for CAA to monitor and manage the resource.

For an application resource, a resource profile contains the application's name, resource dependencies (required and optional resources), monitoring thresholds, script timeout, placement policy, hosting member list, and action script.

The placement policy determines where an application is started. Supported policies are: `balanced`, `avored`, and `restricted`.

<code>balanced</code>	Balanced application resources are distributed among all active members. CAA attempts to start or restart the application resource on the cluster member currently running the fewest application resources.
<code>avored</code>	CAA refers to the hosting members list before starting or restarting the application resource. The first member on the hosting members list is most favored to run the application. If that member is unavailable, CAA places the application resource on any member that is available.  You must specify a hosting members list when you select a favored placement policy.
<code>restricted</code>	Similar to the favored placement policy, except that if none of the members on the hosting members list are available, CAA will not start or restart the application resource. A restricted placement policy ensures that the resource will never run on a member that is not on the list, unless you manually relocate it to that member.  You must specify a hosting members list when you select a restricted placement policy.

The hosting members are, in order of preference, members to consider when the application is (a) started, or (b) relocated. A hosting member list is used in conjunction only with the favored or restricted placement policies.

An action script is a set of commands used by CAA to start and stop an application. It must be located in the `/var/cluster/caa/script` directory.

See the TruCluster Server *Highly Available Applications* manual, `caa_profile(8)`, and `caa(4)` for detailed descriptions of the contents and creation of resource profiles.

## 5.5 Action Scripts

An action script is an executable script similar to that of system initialization files located in the `/sbin/init.d` directory. CAA uses action scripts to start and stop application resources.

An action script has multiple entry points that are executed by the CAA commands when an application resource needs to be started or stopped. The `start` entry point is used by `caa_start` and `caa_relocate` to start an application, and the `stop` entry point is used by `caa_stop` and `caa_relocate` to stop an application.

Both the `caa_profile` command and the SysMan suite of applications can be used to create simple action scripts when creating resource profiles. You may need to edit these action scripts to customize the start and stop procedures for an application.

Each action script has an associated timeout value defined in the application resource profile. If the action script does not finish executing within this time, CAA considers the start attempt a failure and will either attempt to start the application on another member or fail completely.

Action scripts must be located in the `/var/cluster/caa/script` directory. The file names of action scripts take the form `resource_name.scr`

The TruCluster Server *Highly Available Applications* manual provides examples of action scripts.

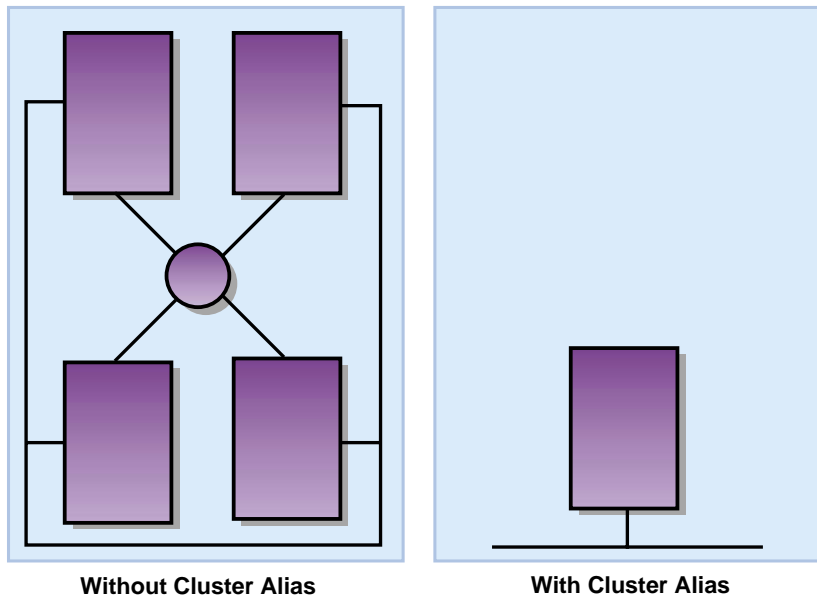


# 6

## Cluster Alias

A **cluster alias** is an IP address that makes some or all of the systems in a cluster look like a single system to the outside world. Figure 6–1 shows how a network client views the systems in a cluster with and without a cluster alias.

Figure 6–1: Client's View of a Cluster With and Without Cluster Alias



ZK-1471U-AI

Think of a cluster alias as a distributed virtual clusterwide network interface. A cluster alias is conceptually similar to an `ifconfig` alias, where a single physical network interface responds to more than one IP address.

### Note

Before TruCluster Server Version 5.0, TruCluster products used the `asemgr` command to control application failover. The `asemgr` command ran the `ifconfig` command to create IP aliases as needed. Because the cluster alias subsystem creates and

manages aliases on a clusterwide basis, there is no longer any need to explicitly establish and remove IP aliases with `ifconfig` when an application fails over.

---

Each system in a cluster can receive packets addressed to a cluster alias. The receiving system silently redirects packets to a cluster member running the requested application or service.

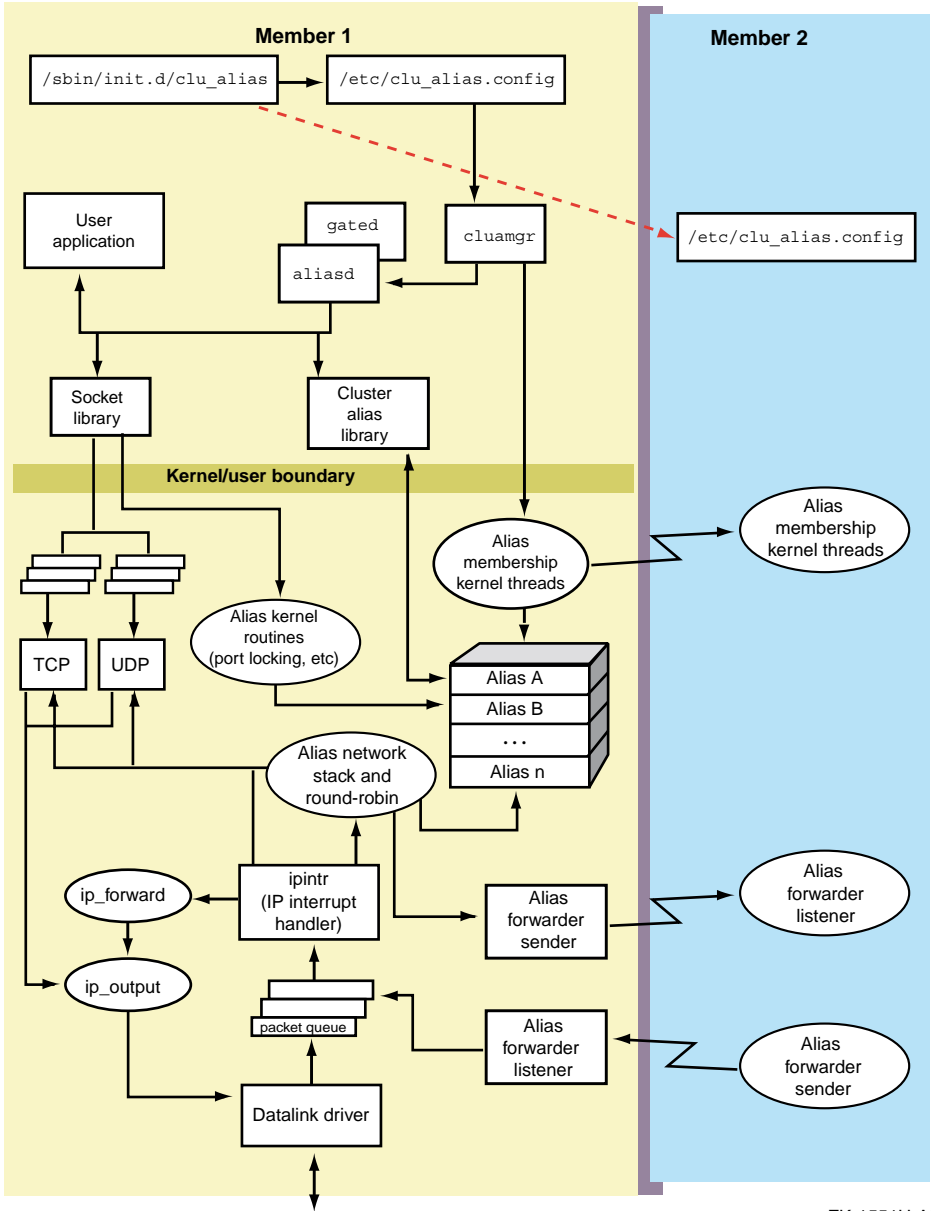
A cluster can have more than one cluster alias. One alias, the default cluster alias, is created during cluster installation and all members can receive packets addressed to this alias. The cluster administrator can create additional aliases as needed. One suggestion is to use the default alias for a while, and then decide whether your site can benefit from additional aliases. In many cases, the default alias is sufficient; the TruCluster Server *Cluster Administration* manual describes a situation where a site uses two aliases for load-balancing.

The cluster alias subsystem has the following main components:

- The kernel portion of the cluster alias subsystem, `clua`, which is a configurable kernel subsystem loaded at boot time.
- A user-level daemon, `aliasd`. The kernel communicates with this daemon to manage routing for cluster aliases. The `cluamgr` command provides flags that can modify the daemon's behavior. Each cluster member runs `aliasd`.
- An administrative interface that provides both a command-line and a graphical user interface (GUI) to manage aliases and alias attributes. The command-line interface is the `cluamgr` command. The GUI is accessed from the SysMan Menu.
- A member-specific alias configuration file, `/etc/clu_alias.config`, which contains the `cluamgr` commands that configure aliases, including the default cluster alias, for that member.
- A clusterwide application configuration file, `/etc/clua_services`, which assigns alias-related attributes to ports used by services.
- An application programming interface (API), `libclua`.

Figure 6-2 provides a functional overview of the cluster alias subsystem components.

Figure 6–2: Cluster Alias Functional Overview



ZK-1551U-AI

This remainder of this chapter discusses the following topics:

- A general overview of cluster aliases (Section 6.1)
- The default cluster alias (Section 6.2)

- The number of aliases a site can have (Section 6.3)
- The location of alias IP addresses (Section 6.4)
- Routing for alias IP addresses (Section 6.5)
- How the cluster alias subsystem routes for `in_single` and `in_multi` services (Section 6.6)
- The attributes assigned to aliases (Section 6.7)
- The attributes assigned to services (Section 6.8)
- How to run an RPC application on multiple cluster members (Section 6.9)
- How packets are redirected within the cluster (Section 6.10)

## 6.1 Overview

Cluster aliases free clients from having to connect to specific cluster members for services. Just as clients can request a variety of services from a single host, clients can request a variety of services from a cluster alias. For example, you can `telnet` or `rlogin` to a cluster alias as you would to single host.

Each system in a cluster explicitly joins the aliases to which it wants to belong. Once a system joins an alias, it is a **member** of that alias. Using the analogy that a cluster alias is similar to an address on virtual network interface, joining an alias is similar to issuing an `ifconfig up` command for that alias interface. The member can now receive packets addressed to the alias.

Clients send TCP connection requests or UDP messages to the IP address representing an alias. The cluster transparently routes the request or message to a cluster node that is a current member of that alias. The hop within the cluster uses the cluster interconnect, not network routing.

If a member of an alias is unavailable, the cluster stops sending packets to that member and routes packets to active members of that alias. As long as one member of an alias is active, the alias is available.

## 6.2 The Default Cluster Alias

There is one special alias, called the **default cluster alias**. During installation, the cluster is given a name, which is stored in `/etc/sysconfigtab` as the value of the `cluster_name` attribute. The installation procedure adds an entry to `/etc/hosts` which associates this cluster name with a user-specified default cluster alias IP address. For example, for a cluster named `deli` whose alias IP address is

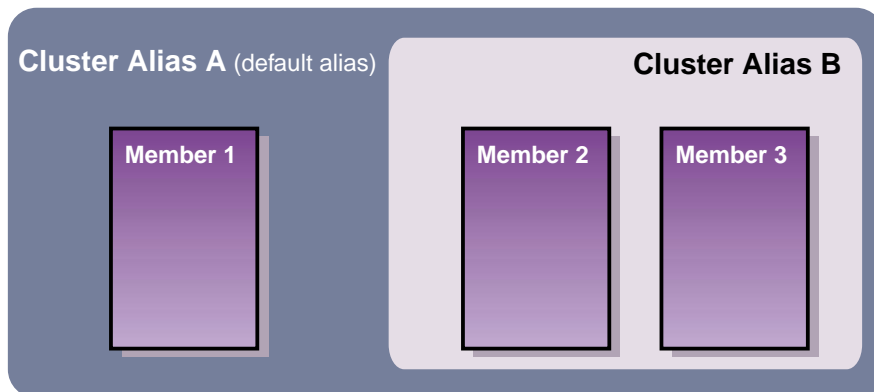
16.140.112.209, the installation procedure adds the following entry to /etc/hosts:

```
16.140.112.209 deli
```

Each cluster member is a member of the default cluster alias. The command that makes a cluster member a member of the default cluster alias is in each member's /etc/clu\_alias.config file. All cluster members automatically join the default cluster alias at boot time.

Figure 6-3 shows a three-node cluster with two cluster aliases. All members belong to alias A, the default cluster alias, but only two members belong to alias B.

**Figure 6-3: Cluster Using Two Aliases**



ZK-1443U-AI

Several standard Internet services use the IP address of the default cluster alias as the source address for outgoing packets. Therefore, all cluster alias IP addresses, including that of the default cluster alias, must be on a network accessible to cluster clients; that is, clients must be able to route to this subnet. For this reason, cluster alias IP addresses cannot be on the Memory Channel subnet used by the cluster for internal communication.

In order to preserve single-system semantics and avoid NFS locking problems, when a cluster is configured as NFS server, the default cluster alias is the IP address through which clients must request NFS services. NFS mount requests directed at individual cluster members are rejected.

---

**Note**

Some clients, for example PCs, broadcast UDP requests when trying to find an NFS server. The cluster responds to these requests by returning the IP address of the default cluster alias.

This ensures that later NFS client requests are sent to the default cluster alias.

---

### 6.3 Number of Aliases

After cluster installation, a site can define as many aliases as it needs. The cluster administrator determines which cluster members join which aliases.

For many clusters, the default cluster alias provides sufficient access for cluster clients. Whether or not a cluster will benefit from having additional aliases depends on the symmetry of the cluster, and whether you want all members to handle client requests for all services. Additional aliases are useful in the following situations:

- In a heterogeneous cluster where some devices or applications are best served through a subset of cluster members.
- If you want to restrict services to a subset of cluster members in order to reduce the internal forwarding of requests and packets.

### 6.4 Location of Alias IP Addresses

A cluster alias address can be in one of two types of subnets:

**common subnet**      A subnet connected to a physical network interface. Using a common subnet for cluster aliases works well when the cluster is connected to only a single local area network, and that network is managed as a single IP address domain.

Cluster alias routing in a common subnet is based on proxy ARP support. For each alias, one cluster member acts as the proxy ARP master for that alias.

**virtual subnet**      A cluster alias resides in a virtual subnet if its address is in a subnet that is not associated with any physical interfaces. A virtual subnet is registered as a normal subnet with local routers.

If the `cluamgr virtual` option is assigned to an alias address, a cluster member will advertise a host route and a network route to the alias.

Regardless of the type of subnet, it must be configured so that packets from clients can be routed to alias addresses. Services that use cluster aliases

will not be accessible to clients if those alias addresses are on a virtual or a common subnet that clients cannot reach.

A cluster alias address should not be a broadcast address or a multicast address, nor should it reside in the Memory Channel subnet.

## 6.5 Routing for Alias Addresses

An **alias router** is a cluster member that makes a cluster alias address known to the network and receives incoming packets for that alias. By default, all cluster members are configured as alias routers at boot time.

A cluster member does not have to be a member of an alias in order to route for that alias. However the cluster member must have an entry for that alias in its `/etc/clu_alias.config` file. In the following example, this cluster member routes for `alias1` and `alias2`, but will receive requests/packets only for `alias1`:

```
/usr/sbin/cluamgr -a alias=alias1,join  
/usr/sbin/cluamgr -a alias=alias2
```

Which cluster members route for which aliases in common subnets is determined by the router priority assigned to each alias on each cluster member. Section 6.7 describes router priority.

The cluster alias daemon, `aliasd`, transparently handles the routing configuration for cluster aliases, automatically adding any needed host routes for cluster aliases to that member's `/etc/gated.conf.membern` file. The daemon starts `gated` using this file as `gated`'s configuration file rather than the member's `/cluster/members/{memb}/etc/gated.conf` file.

---

### Note

---

The `aliasd` daemon supports only the Routing Information Protocol (RIP).

---

### 6.5.1 Common Subnet Routing

Cluster alias routing in a common subnet is based on proxy ARP support. For each alias, one cluster member acts as the proxy ARP master for that alias.

The node elected to advertise the alias configures the alias's IP address to be advertised using proxy ARP over a network interface in the same subnet. Other cluster members that join the alias, and that are configured to be

routing members, become capable of taking over the proxy ARP function if necessary. When a designated alias router or network interface fails, other potential routers are notified. They then elect a new router for the alias.

If a cluster is connected to two external common subnets, and if an alias address resides on one of those common subnets, interfaces directly connected to that subnet will use proxy ARP to advertise the alias. Interfaces connected to all subnets will also use host routes for the alias.

## 6.5.2 Virtual Subnet Routing

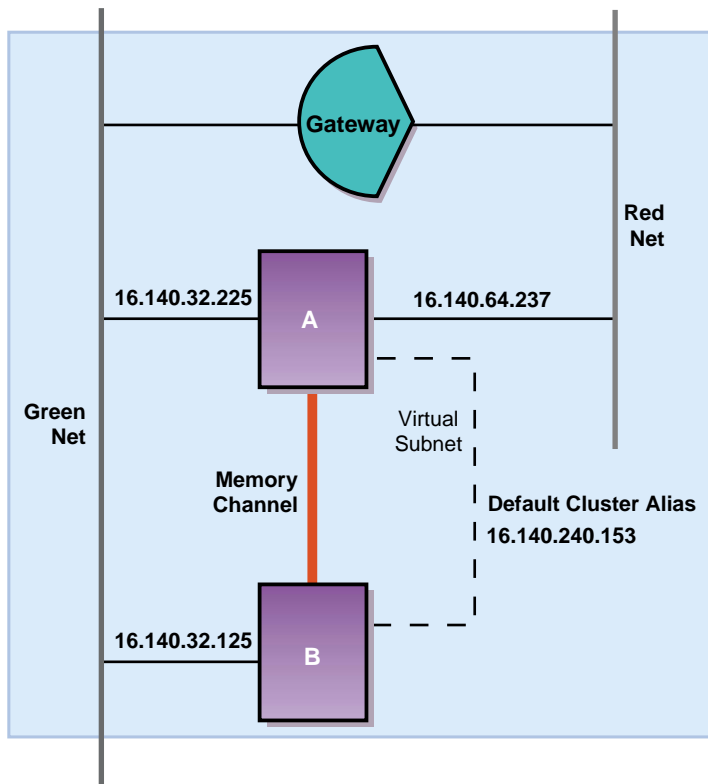
The alias daemon, `aliasd`, creates a `/etc/gated.conf.membern` file for each cluster member. The alias configuration process modifies this configuration file to advertise each alias address in a virtual subnet as a host route. No manual modification is required; each member's alias daemon automatically modifies that member's `/etc/gated.conf.membern` file to advertise a route to each cluster alias host address through each network interface on that member.

If the `cluamgr virtual` option is assigned to an alias address, the cluster member will advertise a network route to the virtual subnet.

## 6.5.3 Routing Example

Figure 6-4 shows a cluster with interfaces on three networks, two public common networks and one private virtual network. The text following the figure describes how the alias subsystem routes requests within the cluster.

Figure 6–4: Alias Routing Example



ZK-1472U-AI

The default cluster alias IP address is on a virtual subnet. Because clients on the Red and Green networks are not on the same subnet as the alias, a host route to the address is advertised on both networks.

Note that although all alias addresses on virtual subnets are advertised through host routes, using a host route does not necessarily mean that an address resides on a virtual subnet.

## 6.6 in\_single and in\_multi Services

Service ports accessed through a cluster alias are defined as either **in\_single** or **in\_multi**. These service port attributes affect the routing of network requests to applications, not whether an application can run on more than one member at the same time. From the point of view of the cluster alias subsystem:

- When a service's port is designated as **in\_single**, only one alias member will receive connection requests or packets for that service. If that member becomes unavailable, the cluster alias subsystem selects

another member of that alias to receive all connection requests or packets.

- When a service's port is designated as `in_multi`, the alias subsystem routes connection requests and packets to all eligible members of the alias.

By default, the cluster alias subsystem treats all services as `in_single`. In order for the cluster alias subsystem to treat a service's port as `in_multi`, the port must either be registered as `in_multi` in `/etc/clua_services` or through a call to `clua_registerservice()`. See Section 6.8 for more information on service attributes.

---

**Note**

---

The `/etc/clua_services` file is the cluster alias extension of the `/etc/services` file. The `clua_services` file extends the `services` syntax to assign alias-related attributes to ports.

---

A service whose port is designated as `in_multi` can take advantage of cluster aliasing to distribute incoming TCP connection requests and UDP packets among members of the alias. The alias subsystem provides load-balancing through a weighted round-robin algorithm that distributes requests/packets among alias members. If one member of an alias cannot respond to client requests, the cluster alias software transparently distributes requests/packets among the remaining alias members.

---

**Note**

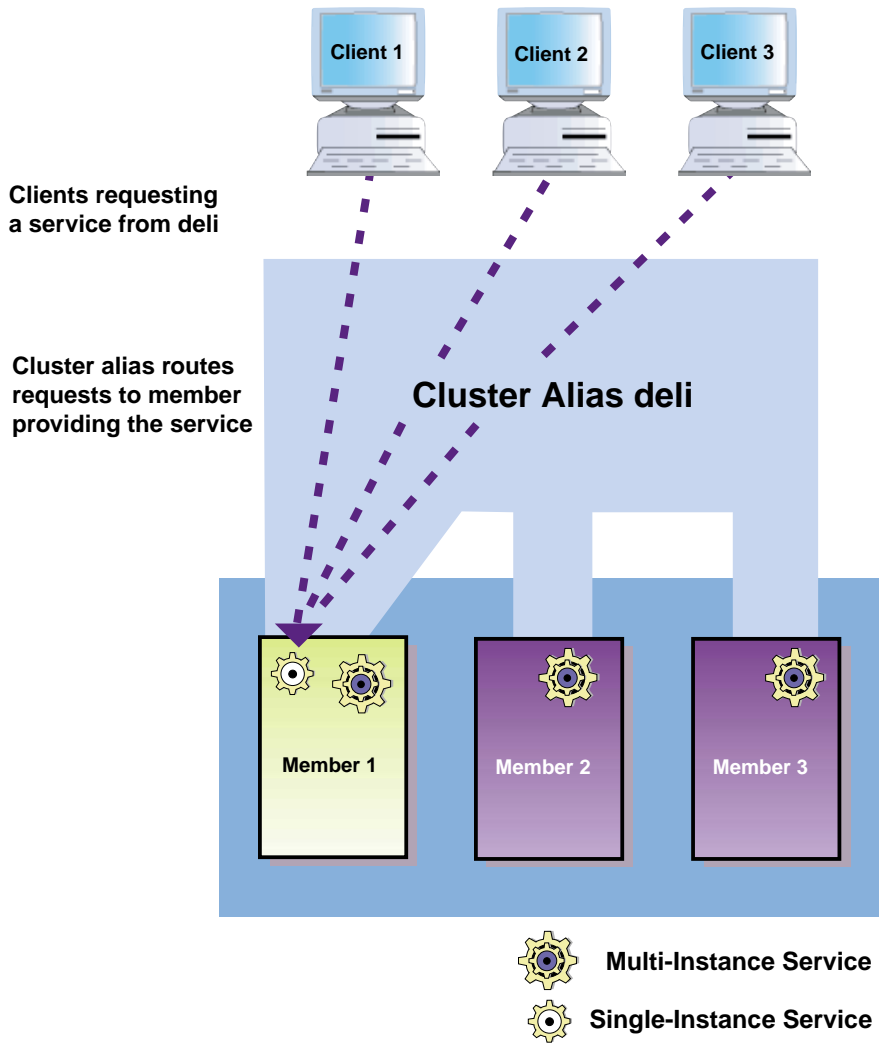
---

Cluster alias and CAA are separate subsystems with complementary but different functions. CAA is an application-control tool; cluster alias is a routing tool. CAA decides where an application will run; cluster alias decides how to get there. You cannot use CAA to control routing within the cluster; you cannot use cluster aliases to control where an application is running in the cluster. The TruCluster Server *Cluster Administration* manual provides more information on the differences between cluster alias and CAA.

---

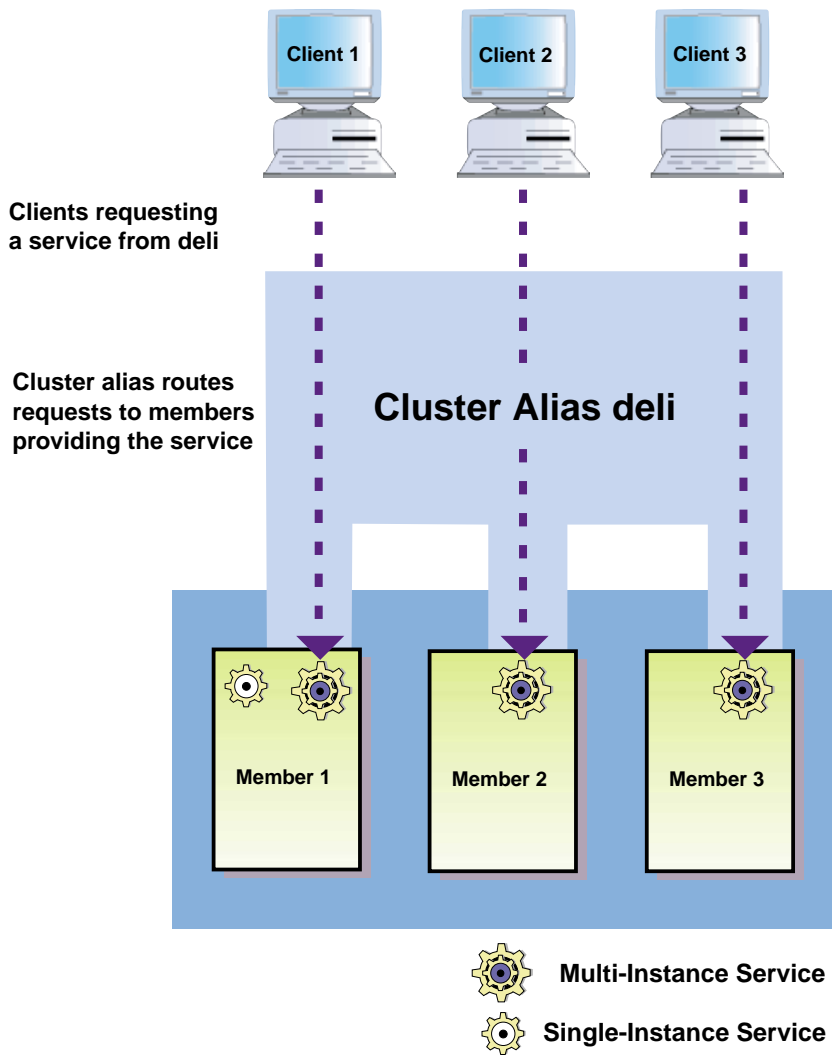
The following two figures show how the alias subsystem distributes client requests for `in_single` and `in_multi` services. For the `in_single` service (Figure 6-5), all requests are sent to the alias member currently running the service. For the `in_multi` service (Figure 6-6), requests are distributed among all alias members.

Figure 6–5: in\_single Service Accessed Through Default Cluster Alias



ZK-1444U-AI

Figure 6–6: in\_multi Service Accessed Through Default Cluster Alias



ZK-1445U-AI

## 6.7 Alias Attributes

Alias attributes are member-specific. Each cluster member has its own view of an alias. For example, one cluster member could route for an alias but not be a member of that alias, but another cluster member could both route for that alias and be an end recipient for requests or messages addressed to that alias.

Aliases and their attributes are managed through the `cluamgr` command and the SysMan Menu. The SysMan Menu calls `cluamgr` as needed. Each member's `/etc/clu_alias.config` file contains the alias configuration information for that member.

The following attributes control the routing and distribution of connection requests and packets among members of an alias. The descriptions are paraphrased from those in `cluamgr(8)`, which describes these and other alias attributes.

- router priority**      The router priority (`rpri`) controls the proxy ARP router selection for an alias on a common subnet. For each alias in a common subnet, the cluster member with the highest router priority for that alias will route for that alias. (This option is not valid for an alias whose address is in a virtual subnet.)
- When a cluster has more than one cluster alias, you can use router priority to spread the routing overhead for aliases among cluster members.
- selection priority**      The selection priority (`selp`) determines the order in which members of an alias receive new connection requests. The selection priority establishes a hierarchy within the members of an alias. Connection requests are distributed among those members sharing the highest selection priority value. If an alias has three members, two with `selp=10` and one with `selp=5`, no connection requests or messages are given to the `selp=5` member as long as either of the `selp=10` members is available.
- You can use selection priority values to set up a failover order for members of a particular cluster alias.
- selection weight**      The selection weight (`selw`) indicates the number of connections (on average) this member is given before connections are given to the next alias member with the same `selp` value. (The `selp` value determines the order in which members are eligible to receive requests or messages; the `selw` value determines how many requests or messages a member gets once it is eligible.)

Selection weight applies only to applications that are registered as `in_multi` services. (All traffic for an `in_single` service must go to the cluster member running that service.)

Selection weight and routing priority address two different load-balancing issues; selection weight is a way to balance application overhead within a cluster, and router priority is a way to balance alias-routing overhead within a cluster.

In general, the default routing priority provides acceptable performance. The selection weight is probably more useful when balancing application loads within a heterogeneous cluster consisting of both large and small systems.

## 6.8 Service Attributes

The `/etc/clua_services` file is similar in concept and syntax to the `/etc/services` file. The `clua_services` file provides a method for associating alias-related attributes with the port numbers used by services. (When application source code is available, the `clua_registerservice()` function serves the same purpose.) Any service with a fixed port assignment can have an entry in `/etc/clua_services`.

With the exception of the `out_alias` attribute, these attributes apply to services accessed through any cluster alias. The `out_alias` attribute, which applies only to connections originating from the cluster, is specific to the default cluster alias.

You can associate the following attributes with a service's port:

### **in\_single**

A service that, from the cluster alias point of view, runs on only one cluster member at a time, but can fail over to another instance of the service on another member should the active service go away. (Active, in this context, relates only to messages addressed to the cluster alias. All instances of a service are always active for their node's local IP address(es) unless the `in_nolocal` attribute is also set.) As each service binds to the application's port, the first is flagged as active for the alias, and the others flagged as inactive. If the active service fails, one of the inactive service daemons is marked as active.

Any port not explicitly listed in `clua_services` as `in_multi`, or registered as an `in_multi` service

through a call to the `clua_registerservice()` function, is treated as an `in_single` service.

**in\_multi**

Indicates a service that can run concurrently on two or more cluster members. For a service using UDP, each packet might go to a different alias member. For a service using TCP, each connection is bound to a single alias member, but different connections to the service from the same client might be established on different alias members.

An `in_multi` service must be explicitly registered, either in the `/etc/clua_services` file or through the `clua_registerservice()` function.

**in\_noalias**

Indicates a service port that will not receive inbound alias messages.

**in\_nolocal**

Indicates that the port will not honor connection requests to non-alias addresses. For TCP, the port will not accept connections; for UDP, the port will drop messages unless addressed to a cluster alias.

**out\_alias**

Indicates that the default cluster alias is used as the source address whenever this port is used as a destination. Normally, outbound connections (or UDP messages) use the local IP address of the cluster member on which the client is running. It is often beneficial to use the cluster alias address as the source address for outbound traffic from the cluster (for example, to simplify authentication).

It is important to remember that the `out_alias` attribute applies *only* when the connection (assuming TCP, not UDP) is originated *from* the cluster; that is, the cluster is the client. If a process running on a cluster member initiates an outbound connection, and the destination port (the port representing that half of the connection that is not in the cluster) is flagged in the cluster's `/etc/clua_services` file as `out_alias`, the connection will use the default alias as its source address.

The same logic holds true when the outbound traffic is a UDP send, because each send can be viewed as a micro-connection.

Consider the following example. An application running on member A in cluster B connects to a service on noncluster node C using port `xxx`. If port `xxx` is in `/etc/clua_services` with the `out_alias` attribute set, the connection to node C comes from node B (the default cluster alias). Otherwise, the connection to node C comes from node A (the cluster member).

The `out_alias` attribute has no bearing whatsoever on the port associated with the cluster half of a connection. Assume that you have more than one alias: the default cluster alias (`deli`) and one that addresses a subset of cluster members (`another-alias`). A client outside the cluster connects via TCP to a service port using the IP address of `another-alias`. All return traffic within the context of that connection will use the IP address of `another-alias` as the source address. (Otherwise, there would be no connection because a connection is uniquely identified by its source address/port and its destination address/port. If a client initiates a connection using `another-alias` as the address, the cluster must respond using `another-alias` as the address.)

The following example shows the use of `out_alias` as a way to apply single-system semantics to X applications displayed from cluster members.

In `/etc/clua_services`, the `out_alias` attribute is set for the X server port (6000). A user on a system outside the cluster wants to run an X application on a cluster member and display back to his or her system. Because the `out_alias` attribute is set on port 6000 in the cluster, the user must specify the name of the default cluster alias when running the `xhost` command to allow X clients access to his or her local system. For example, for a cluster named `deli`, the user would run the following command on the local system:

```
$ xhost +deli
```

This use of `out_alias` allows any X application from any cluster member to display on that user's system. A cluster administrator who wants users to allow access on a per-member basis could either comment out the `Xserver` line in `/etc/clua_services`, or remove the `out_alias` attribute from that line (and then run `cluamgr -f` on each cluster member to make the change take effect.)

**static** Indicates that the port cannot be assigned as a dynamic port. Any well-known port between 512 and 1024 that is either assigned to a specific network service in `/etc/services` or is actively listened to by an application should be declared as `static` in `/etc/clua_services`.

The `in_multi`, `in_single`, and `in_noalias` attributes are mutually exclusive. The `in_nolocal` and `in_noalias` attributes are mutually exclusive. See `clua_services(4)` and `clua_registerservice(3)` for more information about the use of these attributes.

## 6.9 RPC Services and Cluster Alias

RPC services can call either the `clusvc_getcommport()` function or the `clusvc_getresvcommport()` function to bind to a port. (Use `clusvc_getresvcommport()` when binding to a reserved (privileged) port, a port number in the range 0-1023.) Both functions call `clua_registerservice()` to automatically set the `CLUASRV_MULTI` (`in_multi`) service attribute.

Use the `clusvc_getcommport()` and `clusvc_getresvcommport()` functions in the following circumstances:

- The RPC service does not use a well-known port (services that use well-known ports can have `in_multi` entries in `/etc/clua_services`).
- Multiple instances of the RPC service will run in a cluster.
- Requests for the RPC service will be directed to a cluster alias, which will provide load-balancing among the instances of the service.

These two functions make it possible to run an RPC application on multiple cluster members, all accessible via cluster alias. In addition to ensuring that each instance of an RPC application uses the same common port, the functions also inform the portmapper that the application is a

multi-instance, alias application. (Both functions call `clua_registerservice()` to set the `CLUASRV_MULTI` (`in_multi`) service attribute.) If you do not use one of these functions to bind to the port, you can still run multiple instances of the application, but only the instance will receive requests directed to a cluster alias.

## 6.10 Redirecting Packets Within a Cluster

A packet addressed to a cluster alias can arrive at any cluster member. This member must determine which cluster member should receive and process the packet. The alias subsystem monitors calls to `bind()` and `listen()`, and bases its decision on which members of an alias are available and the type of packet received. The following table shows how packets are redirected within a cluster.

New TCP/IP connection	Look at the packet and make a list of eligible members for the target port. Look for active listens on the port. Use the weighted round-robin algorithm to select a member from the list of active listening members. Forward the packet to the selected member.
Existing TCP/IP connection	Determine which alias member owns this connection. Forward the packet to the member.
UDP	Look at the packet and make a list of eligible members for the alias. Use the weighted round-robin algorithm to select the member that should get this packet. Forward the packet to the member.
ICMP (some ICMP packets must be handled in cluster-alias context)	Look at the packet and determine whether to handle it or forward it to another member. If needed, forward the packet to the member.

---

## Memory Channel

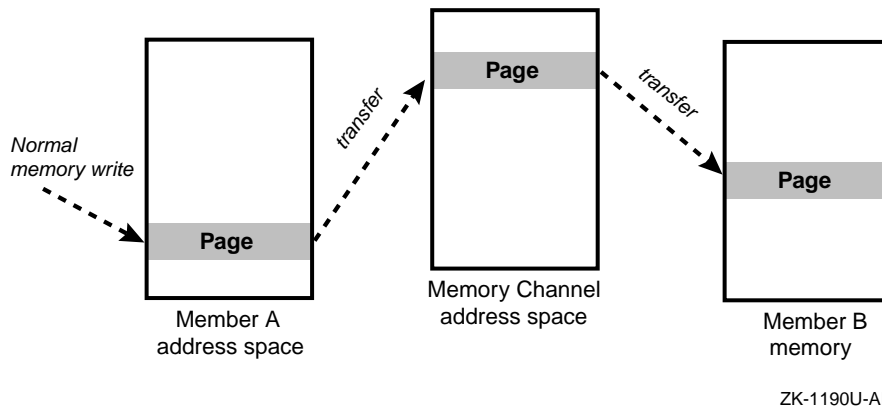
All cluster members must have a direct connection to all other members in order to facilitate communications among members, and to provide a fast and reliable transport for passing messages throughout the cluster. This version of the TruCluster Server product supports the Memory Channel interconnect, a specialized interconnect designed specifically for the needs of clusters. The Memory Channel interconnect provides both broadcast and point-to-point connections between cluster members. Future releases will support other types of cluster interconnect.

The Memory Channel interconnect:

- Allows a cluster member to set up a high-performance, memory-mapped connection to other cluster members. These other cluster members can, in turn, map transfers from the Memory Channel interconnect directly into their memory. A cluster member can thus obtain a write-only window into the memory of other cluster systems. Normal memory transfers across this connection can be accomplished at extremely low latency (3 to 5 microseconds).
- Has built-in error checking, virtually guaranteeing no undetected errors and allowing software error detection mechanisms, such as checksums, to be eliminated. The detected error rate is very low (on the order of one error per year per connection).
- Supports high-performance mutual exclusion locking (by means of spinlocks) for synchronized resource control among cooperating applications.

Figure 7-1 shows the general flow of a Memory Channel transfer.

**Figure 7–1: Memory Channel Logical Diagram**



A Memory Channel adapter must be installed in a PCI slot on each member system. A link cable connects the adapters. If the cluster contains more than two members, a Memory Channel hub is also required.

A redundant, multirail Memory Channel configuration can further improve reliability and availability. It requires a second Memory Channel adapter in each cluster member and link cables to connect the adapters. A second Memory Channel hub is required for clusters containing more than two members.

The Memory Channel multirail model operates on the concept of physical rails and logical rails. A physical rail is defined as a Memory Channel hub with its cables and Memory Channel adapters and the Memory Channel driver for the adapters on each node. A logical rail is made up of one or two physical rails.

A cluster can have one or more logical rails, up to a maximum of four. Logical rails can be configured in the following styles:

- Single-rail
- Failover pair

If a cluster is configured in the single-rail style, there is a one-to-one relationship between physical rails and logical rails. This configuration has no failover properties; if the physical rail fails, the logical rail fails. Its primary use is for high-performance computing applications using the Memory Channel application programming interface (API) library and not for highly available applications.

If a cluster is configured in the failover pair style, a logical rail consists of two physical rails, with one physical rail active and the other inactive. If the active physical rail fails, a failover takes place and the inactive physical

rail is used, allowing the logical rail to remain active after the failover. This failover is transparent to the user. The failover pair style is the default for all multirail configurations.

A cluster fails over from one Memory Channel interconnect to another if a configured and available secondary Memory Channel interconnect exists on all member systems, and one of the following situations occurs in the primary interconnect:

- More than ten errors are logged within one minute.
- A link cable is disconnected.
- The hub is turned off.

After the failover completes, the secondary Memory Channel interconnect becomes the primary interconnect. Another interconnect failover cannot occur until you fix the problem with the interconnect that was originally the primary.

If more than ten Memory Channel errors occur on any member system within a one-minute interval, the Memory Channel error recovery code attempts to determine if a secondary Memory Channel interconnect has been configured on the member as follows:

- If a secondary Memory Channel interconnect exists on all member systems, the member system that encountered the error marks the primary Memory Channel interconnect as bad and instructs all member systems (including itself) to fail over to their secondary Memory Channel interconnect.
- If any member system does not have a secondary Memory Channel interconnect configured and available, the member system that encountered the error displays a message indicating that it has exceeded the Memory Channel hardware error limit and panics.

See the TruCluster Server *Hardware Configuration* manual for information on how to configure the Memory Channel interconnect in a cluster.

The Memory Channel API library implements highly efficient memory sharing between Memory Channel API cluster members, with automatic error-handling, locking, and UNIX style protections. See the TruCluster Server *Highly Available Applications* manual for a discussion of the Memory Channel API library.



---

## Distributed Lock Manager

The distributed lock manager (DLM) provides functions that allow cooperating processes in a cluster to synchronize access to a shared resource, such as a raw disk device or a program. For the DLM to effectively synchronize access to a shared resource, all processes in the cluster that share the resource must use DLM functions to control access to the resource. For example, a distributed database application might use lock manager services to coordinate access to the shared disks participating in a database.

An application secures a lock on a named shared resource. Resource names can be single-dimensional or tree-structured. A resource tree allows you to create a hierarchy of locks and sublocks that reflect the structure of a shared resource. The DLM supplies functions that:

- Provide mutual exclusion, restricted sharing, and full sharing of data access
- Notify a process holding a lock when its lock is blocking another process's access to a resource
- Notify a process that has queued a lock request to a resource when its request has been granted
- Convert a lock's mode between less restrictive and more restrictive lock modes
- Return information about locks

The DLM employs a distributed, centralized tree design. It does not replicate lock information on each cluster member. Rather, the cluster member that manages a lock tree maintains all information about that tree. The member that holds a given lock is aware of only its contribution of that lock to the resource. Any member system can serve as the master for any lock tree, which distributes the overall lock management load.

The DLM uses a distributed directory service to quickly locate the directory node for a resource tree. A directory table associates a root resource name with the cluster member that is the manager of the resource. This directory table is identical on all cluster members.

The DLM is designed to handle member failures. If a lock holder fails, its locks are released. If a member system fails, a new lock master for locks previously mastered on that member is chosen and provided with all pertinent lock information.

---

## Cluster Installation and Administration

This chapter provides an overview of cluster installation and administration.

### 9.1 Installation

Previous versions of TruCluster supported three installation types: full installation, rolling upgrade, and simultaneous upgrade. The rolling upgrade and simultaneous upgrade procedures provided a way to preserve the existing cluster or ASE configuration information while installing a later version of the product.

TruCluster Server Version 5.0 supports two installation types: full installation and a manual upgrade procedure. For TruCluster Server Version 5.0, significant portions of both the cluster architecture and the base operating system architecture have been modified (for example, the Cluster File System (CFS) and device names). For this reason, the recommended installation path is a full installation of the base operating system followed by a full installation of the TruCluster Server. Future releases of TruCluster Server will support rolling upgrades. The manual upgrade installation procedure lets you reuse existing hardware, and manually preserve current configuration settings and apply them to a TruCluster Server cluster.

One major difference in installing TruCluster Server Version 5.0 is that you install Tru64 UNIX Version 5.0 on only one system in the cluster. Because CFS creates shared clusterwide file systems, once a cluster is created, additional members boot into the cluster and have access to these files. (In previous releases, you had to install the base operating system on all cluster members, and there were no clusterwide file systems.)

For TruCluster Server, the initial creation of a cluster, the adding of members, and the removing of members are accomplished through three interactive installation scripts: `clu_create`, `clu_add_member`, and `clu_delete_member`. The scripts provide online help and write log files to the `/cluster/admin` directory.

The following list outlines the steps needed to form a new TruCluster Server Version 5.0 cluster:

1. Using the information in the TruCluster Server *Hardware Configuration*, configure the system and storage hardware and firmware.
2. Selecting AdvFS file systems, install Tru64 UNIX on a private disk on the system that will become the first cluster member.
3. Configure the Tru64 UNIX system, including network and time services. Load and configure the applications you plan to use in the cluster.
4. Load the TruCluster Server license and software.

---

**Note**

---

Each cluster member must have both a Tru64 UNIX license and a TruCluster Server license.

---

5. Run the `clu_create` command to create a boot disk for the first cluster member and to create and populate the clusterwide root (`/`), `/usr`, and `/var` AdvFS file systems.
6. Halt the Tru64 UNIX system and boot the disk containing the first member's cluster boot partition. As the system boots, it forms a single-member cluster and mounts the clusterwide root (`/`), `/usr`, and `/var` file systems.
7. Log in to the single-member cluster and run the `clu_add_member` command to add members to the cluster.

See the TruCluster Server *Software Installation* manual for more information on installing TruCluster Server.

## 9.2 Administration

Having a clusterwide file namespace greatly simplifies cluster management. A cluster has just one copy of most system configuration files. For example, you manage the entire cluster as a single security domain through one `/etc/group` file and one `/etc/passwd` file.

User access to files is independent of which node a user is logged in on, and which node is serving the file. File permissions and access control lists (ACLs) are uniform across the cluster.

Audit logs are kept in a common location; each member's host name is appended to its log files to avoid confusion when tracking audit events.

In most cases, the fact that you are administering a cluster rather than a single system becomes apparent because of the occasional need to manage one of the following aspects of a TruCluster Server environment. Each item is followed by one or more of the cluster-specific commands used to manage or monitor it. With the exception of the installation scripts, you can use the SysMan Menu and SysMan Station GUIs to perform the related command-line functions.

- Cluster creation and configuration, which supports creating the initial cluster member, adding and deleting members, and querying the cluster configuration. (`clu_create`, `clu_add_member`, `clu_delete_member`, and `clu_check_config`)
- Cluster application availability (CAA), which allows you to define and manage highly available applications. (`caa_profile`, `caa_register`, `caa_start`, `caa_stat`, `caa_relocate`, `caa_stop`, and `caa_unregister`)
- Cluster aliases, which provide a single system view from the network. (`cluamgr`)
- Cluster quorum and votes, which determine what constitutes a valid cluster and membership in that cluster, and thereby allows access to cluster resources. (`clu_quorum`)
- Optional load-balancing of the device request dispatcher subsystem. (`drdmgr`)
- Optional load-balancing of CFS servers. (`cfsmgr`)

In addition to the previous items, there are some command-level exceptions to the Single System Image (SSI) model. SSI means that, when possible, the cluster appears to the user like a single computer system. For example, when you execute the `wall` command, the message is sent only to users logged in on the cluster member where the command executes. To send a message to all users logged in on all cluster members, use the `wall -c` form of the command. The same logic applies to the `shutdown` command; you can shut down an individual member or the entire cluster.

See the TruCluster Server *Cluster Administration* manual for more information on configuring and managing a TruCluster Server cluster.



---

## Glossary

The terms in this glossary are commonly used in a TruCluster software environment.

### **action script**

Shell scripts used by CAA to control how applications are started, stopped, and checked. Action scripts are located in the `/var/cluster/caa/script` directory. The file names of action scripts take the form `resource_name.scr`.

### **adapter**

A device that converts the protocol and hardware interface of one bus type into that of another bus.

### **address switches**

Electrical switches on the side or rear of some disk drives that determine the SCSI address setting for the drive.

### **alias router**

A cluster member that makes a cluster alias address known to the network and receives incoming packets for that alias. By default, all cluster members are configured as alias routers at boot time.

### **availability**

The characteristic of a computing system that allows it to provide computing services (such as applications) to clients with little or no disruption.

See also *highly available*

### **bus**

Flat or twisted-wire cable or a backplane composed of individual parallel circuits. A bus connects computer system components to provide communications paths for addresses, data, and control information.

### **CDSL**

See *context-dependent symbolic link (CDSL)*

### **client**

A computer system that uses resources provided by another computer, called a server.

**cluster**

A loosely coupled collection of servers that share storage and other resources that make applications and data highly available. A cluster consists of communications media, member systems, peripheral devices, and applications. The systems communicate over a high-performance interconnect.

**cluster alias**

An IP address used to address all or a subset of the members in a cluster. A cluster alias makes some or all of the systems in a cluster look like a single system to the outside world.

**cluster application availability (CAA)**

The cluster application availability (CAA) subsystem provides high availability for single-instance applications and monitoring of the state of other types of resources (such as network interfaces). A single instance of any application that can run on Tru64 UNIX can be made highly available in a cluster with CAA.

**cluster expected votes**

See *expected votes*

**Cluster File System (CFS)**

A cluster virtual file system that sits above the physical file systems and provides clusterwide access (with assistance from the device request dispatcher) to all mounted file systems in a cluster. CFS maintains cache coherency across all cluster members, thus ensuring that all members have an identical, consistent view of file systems directly connected to the cluster.

**cluster interconnect**

Private physical bus employed by cluster members for intracluster communications.

**cluster member**

The basic computing resource in a cluster. A member system must be physically connected to a cluster interconnect and at least one shared SCSI bus.

In common usage, a system configured with TruCluster Server software that is capable of joining a cluster. From the point of view of the connection manager, a system that has either formed a single-member cluster or has been granted membership in an existing cluster. The connection manager dynamically determines cluster membership based on communications among the cluster members. Only an active cluster member can access the shared resources of a cluster.

**cluster partition**

A situation in which an existing cluster can divide into two or more clusters.

**cluster router**

In the context of cluster aliases, a cluster member that makes a cluster alias IP address known to the network and receives incoming packets addressed to the alias. By default, all cluster members are cluster routers.

**common subnet**

In the context of cluster aliases, an existing physical subnet. Cluster alias IP addresses are either in a common subnet or in a virtual subnet.

**connection manager**

Cluster software component that coordinates participation of systems in the cluster, and maintains cluster integrity when systems join or leave the cluster.

**context-dependent symbolic link (CDSL)**

A special form of a symbolic link whose target pathname includes an environment variable, {memb}, which is resolved at run time. In a cluster, CDSLs make it possible to maintain per-system configuration and data files within the shared CFS root (/), /usr, and /var file systems.

**current votes**

The number of votes contributed by current cluster members and the quorum disk as seen by this member.

**device request dispatcher**

A kernel subsystem that controls all I/O access to storage devices in a cluster. The device request dispatcher supports clusterwide access to both character and block disk devices.

Note: Do not confuse the device request dispatcher with the Distributed Raw Disk (DRD) services provided in the TruCluster Production Server product. The device request dispatcher is fully integrated with the kernel and removes the need for having a specific service to make storage accessible to cluster members.

**default cluster alias**

A special cluster alias created during cluster installation. All cluster members are, by default, members of the default cluster alias.

**differential SCSI bus**

A SCSI bus where the signal's level is determined by the potential difference between two wires.

**distributed application**

An application that is specifically designed to run on a cluster, using different members for specific purposes. These applications use the Memory Channel, distributed lock manager (DLM), and cluster alias application programming interfaces to integrate application with the cluster resources.

**distributed lock manager**

Cluster software component that synchronizes access to shared resources among cooperating processes throughout the cluster.

**DLM**

See *distributed lock manager*

**expected votes**

The sum of all member votes held by cluster members, plus the vote of the quorum disk (if one is defined).

**event manager**

The event manager (EVM) facility lets kernel-level and user-level processes and components post events, and provides a means for processes to subscribe for notification when selected events occur. The facility provides an event viewer, an API, and command-line utilities. See *EVM(5)* for more information.

**EVM**

See *event manager*

**failover**

A transfer of the responsibility to provide services. A failover occurs when a hardware or software failure causes a service to restart on another member system.

**failover pair**

A Memory Channel logical rail configuration that consists of two physical rails, with one physical rail active and the other inactive. If the active physical rail fails, a failover takes place and the inactive physical rail is used.

**fast SCSI**

An optional mode of SCSI-2 that allows transmission rates of up to 10 MB per second.

**fast bus speed**

A bus speed that uses the fast synchronous transfer option, enabling I/O devices to attain high peak-rate transfers (10 MB per second) in synchronous mode.

**firmware**

Software code stored in hardware.

**highly available**

In the TruCluster software, the ability to survive any single hardware or software failure.

A cluster can be considered highly available if the hardware and software provides protection against any single failure, such as a system or disk failure or a SCSI cable disconnection.

A service can be considered highly available if the hardware it depends on provides protection against any single failure, and the service is configured to fail over in case of a failure.

**hot swap**

The ability to replace a device on a shared bus while the bus is active.

**in\_multi service**

When a service's port is designated as `in_multi`, the cluster alias subsystem routes connection requests and packets to all eligible members of the alias.

**in\_single service**

When a service's port is designated as `in_single`, the cluster alias subsystem ensures that only one alias member will receive connection requests or packets for that service.

**local bus**

See *private SCSI bus*

**lock file**

A file that indicates that operations on one or more other files are restricted or prohibited. The presence of the lock file can be used as the indication, or the lock file can contain information describing the nature of the restrictions.

**logical rail**

One or more Memory Channel physical rails. Logical rails are configured as a single-rail or as a failover pair.

**Logical Storage Manager**

The Logical Storage Manager (LSM) is a disk storage management tool that protects against data loss, improves disk I/O performance, and customizes the disk configuration.

System administrators use LSM to perform disk management functions without disrupting users or applications accessing data on those disks.

**LSM**

See *Logical Storage Manager*

**LSM disk group**

A group of Logical Storage Manager (LSM) disks that share a common configuration. The configuration information for an LSM disk group consists of a set of records describing objects including LSM disks, LSM volumes, LSM plexes, and LSM subdisks that are associated with the LSM disk group. Each LSM disk group has an administrator-assigned name that can be used to reference that LSM disk group.

**LSM volume**

A Logical Storage Manager (LSM) volume is a special device that contains data used by a UNIX file system, a database, or other applications. LSM transparently places an LSM volume between applications and a physical disk. Applications then operate on the LSM volume rather than on the physical disk. For example, a file system is created on an LSM volume rather than on a physical disk.

An LSM volume presents block and raw interfaces that are compatible in their use with disk partition special devices. Because an LSM volume is a virtual device, it can be mirrored, spanned across disk drives, moved to use different storage, and striped using administrative commands. The configuration of an LSM volume can be changed using LSM utilities without disrupting applications or file systems that are using the LSM volume.

**LSM plex**

A Logical Storage Manager (LSM) plex is a copy of an LSM volume's logical data address space, sometimes known as a mirror. An LSM volume can have up to eight LSM plexes associated with it. A read can be satisfied from any LSM plex, while a write is directed to all LSM plexes.

**logical unit number**

A physical or virtual peripheral device addressable through a target. LUNs use their target's bus connection to communicate on a SCSI bus.

**LUN**

See *logical unit number*

**member**

See *cluster member*

**member ID**

An integer, in the range 1-63, used to identify a cluster member system. Each member has a unique member ID, which is assigned during the installation procedure.

**member votes**

The number of quorum votes assigned to a cluster member.

**Memory Channel interconnect**

A peripheral component interconnect (PCI) cluster interconnect that provides fast and reliable communications between cluster members. Physically, the interconnect consists of a Memory Channel adapter installed in a PCI slot in each member system, one or more Memory Channel link cables to connect the adapters, and an optional Memory Channel hub.

**mount point**

A directory file that is the name of a mounted file system.

**multi-instance application**

An application that can run on multiple cluster members at the same time. A multi-instance application by definition is highly available because the failure of one cluster member does not affect the instances of the application running on other members.

**network**

Two or more computing systems that are linked for the purpose of exchanging information and sharing resources.

**network interface**

The network adapter and the software that allows a system to communicate over a network.

**nonvoting member**

A cluster member with 0 (zero) votes is considered to be a nonvoting member.

See also *voting member*

**partition**

An abnormal condition in which nodes in an existing cluster divide into two independent clusters.

**peripheral component interconnect**

An industry-standard expansion I/O bus that is a synchronous, asymmetrical I/O channel.

**PCI**

See *peripheral component interconnect*

**personality module**

The module on a storage shelf that provides the interface between a differential SCSI bus and the storage shelf single-ended SCSI bus.

Switches on the module enable SCSI bus termination and control SCSI bus IDs for the storage shelf.

**physical rail**

A Memory Channel hub with its cables and Memory Channel adapters and the Memory Channel driver for the adapters on each node.

See also *logical rail*

**placement policy**

A placement policy determines where an application under CAA control is run. Supported policies are: *balanced*, *favored*, and *restricted*.

**private SCSI bus**

A SCSI bus that connects private storage to the local system.

**private storage**

A storage device on a private SCSI bus. Storage devices include hard disks, floppy disks, and compact disk drives, tape drives, and other devices.

**proxy ARP**

The Address Resolution Protocol (ARP) is used to map dynamically between IP addresses and Ethernet addresses. An ARP request contains the IP address of an interface on the target host. The host that recognizes this IP address should respond with its Ethernet address. All other hosts should ignore the ARP request.

Proxy ARP is, in essence, when a system or router lies about being the system with an interface that matches the IP address in the ARP request. The proxy ARP system responds to the ARP request by returning its own Ethernet address. The system then routes the packets to the real target system. Proxy ARP is useful for subnetting and also when adding routers to a topology where some hosts are not yet configured to use the routers.

In a cluster, proxy ARP is the mechanism used by the physical cluster members to handle requests addressed to cluster aliases whose addresses reside on common subnets.

**quorum**

A cluster state in which members are allowed to access clusterwide shared resources and thus perform useful work. The cluster has quorum when the connection manager determines that the member and quorum disk votes in the cluster equal or exceed the required number of quorum votes.

See also *quorum votes*

**quorum algorithm**

A mathematical method that the connection manager uses to determine the circumstances under which a given member can participate in a cluster, safely accessing clusterwide resources and performing useful work.

**quorum disk**

A disk whose `h` partition contains cluster status and quorum information. Each cluster can have a maximum of one quorum disk. The quorum disk is assigned votes that are used when calculating quorum.

**quorum disk votes**

The number of votes that a quorum disk contributes towards quorum.

**quorum loss**

A cluster state in which no member is allowed to access clusterwide shared resources. A cluster enters a quorum loss state when the connection manager determines that the member and quorum disk votes in the cluster are less than the required number of quorum votes.

See also *quorum votes*

**quorum votes**

The number of votes required to form or maintain a cluster. The formula for calculating quorum votes is:

```
quorum votes = round_down((cluster-expected-votes+2)/2)
```

**RAID**

See *redundant array of inexpensive disks (RAID)*

**RAID array controller**

A SCSI bus adapter between a differential SCSI bus and the single-ended RAID array storage shelves. It responds to host commands to access the RAID array disk or tape devices.

**redundant array of inexpensive disks (RAID)**

A technique that organizes disk data to improve performance and reliability. RAID has three attributes:

- It is a set of physical disks viewed by the user as a single logical device or multiple logical devices.
- Disk data is distributed across the physical set of drives in a defined manner.
- Redundant disk capacity is added so data can be recovered if a drive fails.

**redundant**

Describes duplicate hardware that provides spare capacity that can be used when a component fails.

**resource**

A cluster hardware or software component that provides a service to end users or to other software components. Examples of resources are disks, tapes, file systems, network interfaces, and application software.

**resource profile**

Each application under CAA control has a resource profile, which contains that application's resource requirements. The file contains keyword/value pairs used by CAA to monitor resources and control application failover. Resource profiles are located in the `/var/cluster/caa/profile` directory. The file names of resource profiles take the form `resource_name.cap`.

**script**

A program that is interpreted and executed by the shell.

**SCSI**

See *Small Computer System Interface*

**SCSI-2**

An extension to the original SCSI standard featuring multiple systems on the same bus and hot swap. Hot swap is the ability to replace a device on a shared bus while the bus is active. The SCSI-2 standard is ANSI standard X3.T9.2/86-109.

**SCSI adapter**

A storage adapter, commonly referred to as a host bus adapter (HBA), that provides a connection between an I/O bus and a SCSI bus.

**SCSI bus**

A bus that supports the transmission and signaling requirements of a SCSI protocol.

**SCSI bus speed**

The data transfer speed for a SCSI bus. SCSI bus speed can be either slow, up to 5 MB/s; fast, up to 10 MB/s; fast and wide, up to 20 MB/s; or UltraSCSI, up to 40 MB/s.

**SCSI controller**

See *SCSI adapter*

**SCSI device**

A SCSI controller, peripheral controller, or intelligent peripheral that can be attached to a SCSI bus.

**SCSI ID**

Unique address from 0-15 that identifies a device on a SCSI bus.

**server**

A computing system that provides a specific set of applications or data to clients.

**shared SCSI bus**

A SCSI bus that is connected to more than one member system and, optionally, one or more storage devices.

**shared storage**

Disks that are connected to a shared SCSI bus.

**signal converter**

Converts signals between a single-ended SCSI bus and a differential SCSI bus.

**single-ended SCSI bus**

A signal path in which one data lead and one ground lead are utilized to make a device connection. This transmission method is economical, but is more susceptible to noise than a differential SCSI bus.

**single-instance application**

An application that is run on only one cluster member at a time. The cluster application availability (CAA) subsystem can provide high availability for single-instance applications by controlling the initial startup and failover characteristics for a single-instance application.

**single rail**

A Memory Channel logical rail configuration where there is a one-to-one relationship between physical rails and logical rails. This configuration has no failover properties; if the physical rail fails, the logical rail fails.

**Small Computer System Interface**

An American National Standards Institute (ANSI) standard interface for connecting disks and other peripheral devices to a computer system. SCSI-based devices can be configured in a series, with multiple devices on the same bus.

**SRM**

External interface to console firmware for operating systems that expect firmware compliance with the Alpha System Reference Manual (SRM).

**standard mode**

A Memory Channel interconnect configuration that uses a Memory Channel hub to connect Memory Channel adapters. To set up a Memory

Channel interconnect in standard mode, use a link cable to connect each Memory Channel adapter to a linecard installed in a Memory Channel hub.

**StorageWorks**

The modular storage subsystem (MSS), which consists of a family of mass storage products that can be configured to meet current and future storage needs.

**subset**

A software module that can be installed, which is compatible with the Tru64 UNIX `setld` software installation utility.

**system bus**

The private (nonshared) interconnect used on the CPU subsystem. This bus connects the processor module, the memory module, and the I/O module.

**target**

A device that can be addressed by a SCSI ID on a SCSI bus.

**terminator**

Resistor array device used for terminating a SCSI bus. A SCSI bus must be terminated at its two physical ends.

**trilink connector**

A connector that joins two cables to a single device, or allows terminating a shared SCSI bus external to the adapter or RAID controller.

**tunneling**

In the context of cluster aliases, moving an mbuf chain between cluster members after receipt.

**UltraSCSI**

A differential SCSI bus standard that uses smaller diameter cables with smaller connectors and allows bus speeds up to 40 MB/s at 25 meters.

**UltraSCSI hub**

A specialized signal converter with multiple connectors. An UltraSCSI hub converts differential input SCSI signals from a host bus adapter to single-ended, then converts the single-ended signals back to differential for the output connection to a RAID array controller. An UltraSCSI hub allows radial connection of UltraSCSI devices and increases the separation between host and storage.

**virtual hub mode**

A Memory Channel interconnect configuration that does not use a Memory Channel hub to connect Memory Channel adapters. Virtual hub mode is supported only for clusters that have two member systems. To set up a

Memory Channel interconnect in virtual hub mode, use a Memory Channel link cable to connect the Memory Channel adapter in one member system to the corresponding Memory Channel adapter in the other member system.

**virtual subnet**

In the context of cluster aliases, a subnet with no physical connections. Cluster alias IP addresses are either in a common subnet or in a virtual subnet.

**votes**

Votes (either 0 or 1) are contributed to the cluster by cluster members and by the **quorum disk** if one is configured. The connection manager uses votes to calculate quorum.

**voting member**

Each member with a vote is considered to be a voting member of the cluster.

See also *nonvoting member*

**worldwide ID**

A worldwide ID (WWID) is a unique identifier assigned to a disk by its manufacturer.

**WWID**

See *worldwide ID*

**Y cable**

A cable that joins two cables to a single device, or allows terminating a shared SCSI bus external to the adapter or RAID controller.



---

# Index

## A

---

- access control lists, 9-2
- action script, 5-4
- AdvFS
  - supported read/write, 2-2
- aliasd
  - definition, 6-2
  - supports RIP, 6-7
- applications
  - high availability with CAA, 5-1
  - routing requests to with cluster alias, 6-9
  - types of, 4-1
- audit logs, 9-2

## B

---

- balanced placement policy, 5-6

## C

---

- CAA
  - action script, 5-4
  - caad daemon, 5-3
  - compared to cluster alias, 6-10n
  - description, 5-1
  - placement policy, 5-6
  - resource manager, 5-3
  - resource profile, 5-3
- caad daemon, 5-3
- CDFS
  - supported read-only, 2-2
- CDSL, 2-7
- CFS
  - cfsmgr command, 2-4
  - description, 2-4

- preserves X/Open and POSIX semantics, 2-4
- clua\_registerservice() function, 6-10
- clua\_services file, 6-14
- cluamgr command, 6-2
- cluster alias
  - compared to CAA, 6-10n
  - default, 6-2, 6-5
  - description, 6-4
  - how many, 6-6
  - packet redirection, 6-18
  - routing for, 6-7
- cluster alias attributes
  - router priority, 6-13
  - selection priority, 6-13
  - selection weight, 6-13
- cluster alias service attributes
  - in\_multi, 6-15
  - in\_noalias, 6-15
  - in\_nolocal, 6-15
  - in\_single, 6-14
  - out\_alias, 6-15
  - static, 6-17
- cluster application availability (*See CAA*)
- Cluster File System (*See CFS*)
- cluster\_expected\_votes attribute, 3-3
- cluster\_node\_votes attribute, 3-2
- cluster\_qdisk\_votes attribute, 3-2
- clusvc\_getcommport() function, 6-17
- clusvc\_getresvcommport() function, 6-17
- common subnet
  - alias routing, 6-7
  - definition, 6-6
- connection manager

description, 3-1  
monitoring, 3-9  
current votes, 3-4

## D

---

default cluster alias  
description, 6-5  
relationship to out\_alias service  
attribute, 6-16  
suggestion for use, 6-2  
device names  
clusters support only new-style,  
2-9  
consistent clusterwide, 2-9  
disk, 2-10  
drdmgr command, 2-6  
examples of new-style, 2-10  
how determined, 2-9  
identifying, 2-11  
new naming model, 2-9  
tape, 2-10  
device request dispatcher  
description, 2-5  
drdmgr command, 2-6  
distributed application  
description, 4-1  
distributed lock manager  
( See *DLM* )  
DLM  
description, 8-1  
documentation  
HTML versions of, viii  
documents  
related, viii  
drdmgr command  
example, 2-6  
dsfmgr command, 2-11t

## E

---

/etc/clua\_services, 6-10  
similar in concept to  
/etc/services, 6-14

/etc/gated.conf.member<n>, 6-7  
created and maintained by  
aliasd, 6-8  
expected votes  
calculating, 3-3  
description, 3-3  
member-specific, 3-3

## F

---

favored placement policy, 5-6  
FFM  
supported for local use only, 2-2

## G

---

gated daemon, 6-7  
gated.conf.member<n> file, 6-7

## H

---

HTML  
documentation in, viii  
hwmgr command, 2-11t

## I

---

in\_multi  
attribute, 6-15  
service, 6-9  
in\_noalias attribute, 6-15  
in\_nolocal attribute, 6-15  
in\_single  
attribute, 6-14  
service, 6-9  
installation, 9-1  
synopsis of steps, 9-2

## L

---

logical rail, 7-2  
failover pair, 7-2  
single-rail, 7-2  
Logical Storage Manager

( *See LSM* )  
LSM  
description, 2-12

## M

---

membership  
monitoring, 3-9  
Memory Channel  
description, 7-1  
logical rail, 7-2  
physical rail, 7-2  
Memory File System  
( *See MFS* )  
MFS  
not supported, 2-2  
multi-instance application  
description, 4-1

## N

---

named pipes  
supported for local use only, 2-2  
NFS client  
supported read/write, 2-2  
NFS server  
supported read/write, 2-2

## O

---

out\_alias attribute, 6-15

## P

---

partition  
cluster member boot device, 2-5  
of cluster, 3-8  
PC-NFS  
supported read/write, 2-2  
physical rail, 7-2  
PIDs  
unique per cluster member, 1-5t  
pipes

named pipes supported for local  
use only, 2-2  
placement policy  
balanced, 5-6  
favored, 5-6  
restricted, 5-6  
/proc file system  
supported for local use only, 2-2  
process IDs  
( *See PIDs* )

## Q

---

quorum  
calculating, 3-3  
loss of, 3-4  
quorum algorithm  
description, 3-3  
quorum disk  
and LSM, 3-8  
configuring, 3-7  
number of votes, 3-8  
using, 3-5  
quorum disk votes, 3-2  
quorum loss, 3-4  
quorum votes, 3-4  
calculating, 3-3

## R

---

resource  
defined for CAA, 5-5  
resource manager, 5-3  
resource profile, 5-3  
example, 5-5  
restricted placement policy, 5-6  
RIP  
supported by aliasd, 6-7  
router priority attribute, 6-13  
RPC services  
interaction with cluster alias,  
6-17

## **S**

---

- selection priority attribute, 6-13
- selection weight attribute, 6-13
- services
  - in\_single and in\_multi, 6-9
- single-instance application
  - description, 4-1
- single-system management, 1-5, 9-2
- static attribute, 6-17
- subnet
  - common, 6-6
  - virtual, 6-6

## **U**

---

- UFS
  - supported read-only, 2-2

## **V**

---

- virtual subnet

- alias routing, 6-8
- definition, 6-6
- votes, 3-2
  - current, 3-4
  - expected, 3-3
  - node, 3-2
  - quorum, 3-4
  - quorum disk, 3-2

## **W**

---

- Worldwide ID, 2-11

## **X**

---

- xhost command
  - when to use the default cluster
    - alias as the host name, 6-16

---

## How to Order Tru64 UNIX Documentation

You can order documentation for the Tru64 UNIX operating system and related products at the following Web site:

<http://www.businesslink.digital.com/>

If you need help deciding which documentation best meets your needs, see the Tru64 UNIX *Documentation Overview* or call **800-344-4825** in the United States and Canada. In Puerto Rico, call **787-781-0505**. In other countries, contact your local Compaq subsidiary.

If you have access to Compaq's intranet, you can place an order at the following Web site:

<http://asmorder.ngo.dec.com/>

The following table provides the order numbers for the Tru64 UNIX operating system documentation kits. For additional information about ordering this and related documentation, see the *Documentation Overview* or contact Compaq.

---

Name	Order Number
Tru64 UNIX Documentation CD-ROM	QA-6ADAA-G8
Tru64 UNIX Documentation Kit	QA-6ADAA-GZ
End User Documentation Kit	QA-6ADAB-GZ
Startup Documentation Kit	QA-6ADAC-GZ
General User Documentation Kit	QA-6ADAD-GZ
System and Network Management Documentation Kit	QA-6ADAE-GZ
Developer's Documentation Kit	QA-6ADAG-GZ
Reference Pages Documentation Kit	QA-6ADAF-GZ

---



---

## Reader's Comments

### TruCluster Server

Technical Overview

AA-RHGVA-TE

Compaq welcomes your comments and suggestions on this manual. Your input will help us to write documentation that meets your needs. Please send your suggestions using one of the following methods:

- This postage-paid form
- Internet electronic mail: [readers\\_comment@zk3.dec.com](mailto:readers_comment@zk3.dec.com)
- Fax: (603) 884-0120, Attn: UBPG Publications, ZK03-3/Y32

If you are not using this form, please be sure you include the name of the document, the page number, and the product name and version.

#### Please rate this manual:

	Excellent	Good	Fair	Poor
Accuracy (software works as manual says)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Clarity (easy to understand)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Organization (structure of subject matter)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Figures (useful)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Examples (useful)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Index (ability to find topic)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Usability (ability to access information quickly)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

#### Please list errors you have found in this manual:

Page	Description
_____	_____
_____	_____
_____	_____
_____	_____

#### Additional comments or suggestions to improve this manual:

\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

What version of the software described by this manual are you using? \_\_\_\_\_

Name, title, department \_\_\_\_\_

Mailing address \_\_\_\_\_

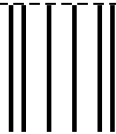
Electronic mail \_\_\_\_\_

Telephone \_\_\_\_\_

Date \_\_\_\_\_

----- Do Not Cut or Tear - Fold Here and Tape -----

**COMPAQ**



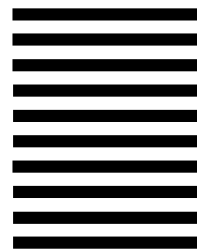
NO POSTAGE  
NECESSARY IF  
MAILED IN THE  
UNITED STATES

**BUSINESS REPLY MAIL**

FIRST CLASS MAIL PERMIT NO. 33 MAYNARD MA

POSTAGE WILL BE PAID BY ADDRESSEE

COMPAQ COMPUTER CORPORATION  
UBPG PUBLICATIONS MANAGER  
ZKO3-3/Y32  
110 SPIT BROOK RD  
NASHUA NH 03062-9987



----- Do Not Cut or Tear - Fold Here -----

Cut on Dotted Line