Sun Cluster 2.2 Cluster Volume Manager Guide

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303-4900
U.S.A.

Part No: 806–2329
July 14 1999

Contents

Preface

This manual, *Sun Cluster 2.2 Cluster Volume Manager Guide*, describes Cluster Volume Manager (CVM) as it is used with Sun Cluster systems. CVM is a variation of and incorporates the Sun StorEdge Volume Manager™ (SSVM). It is produced by Veritas. CVM is designed specifically for use in a clustered system environment.

This manual is written for two audiences: experienced system administrators with networking knowledge and programmers who are designing applications that incorporate and/or use Sun Cluster.

Before You Read This Book

In order to fully use the information in this document, you must be thoroughly familiar with the topics described in the books referenced in the section Section 0.5 “Related Documents” on page 0-ix as appropriate to your purchased product. For procedural information and document references, refer to the appropriate manual. For terms and definitions, refer to the glossary in the *Sun Cluster 2.2 System Administration Guide.*

Using UNIX Commands

This document may not contain information on basic UNIX™ commands and procedures such as shutting down the system, booting the system, and configuring devices.

See one or more of the following for this information:

- *Solaris™ 2.x Handbook for SMCC Peripherals*
- AnswerBook™ online documentation for the *Solaris* 2.x software environment
- Other software documentation that you received with your system

Typographic Conventions

TABLE P–1 Typographic Conventions

Typeface or Symbol	Meaning	Examples
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output.	Edit your `.login` file. Use `ls -a` to list all files. `% You have mail`.

`AaBbCc123`	What you type, when contrasted with on-screen computer output.	`%` **`su`**
		`Password:`

AaBbCc123	Book titles, new words or terms, words to be emphasized. Command-line variable; replace with a real name or value.	Read Chapter 6 in the *User's Guide.* These are called *class* options. You *must* be `root` to do this. To delete a file, type `rm` *filename.*

Shell Prompts

TABLE P–2 Shell Prompts

Shell	Prompt
C shell	*machine_name*%
C shell superuser	*machine_name*#
Bourne shell and Korn shell	$
Bourne shell and Korn shell superuser	#

Related Documents

The following tables list documents that contain information helpful to the system administrator and service provider.

TABLE P–3 Related Sun Cluster Documents

Product Family	Title	Part Number
Cluster Hardware and Software	*Sun Enterprise Cluster System Site Preparation, Planning, and Installation Guide*	805-6512
	Sun Enterprise Cluster Hardware Service Manual	805-6511
	Sun Cluster 2.2 System Administration Guide	805-4238
	Sun Cluster 2.2 Software Installation Guide	805-4239
	Sun Cluster 2.2 API Developer's Guide	805-4241
	Sun Cluster 2.2 Error Messages Manual	805-4242
	Sun Cluster 2.2 Release Notes	805-4243

TABLE P–4 Diagnostic, Database, and Other Documents

Product Family	Title	Part Number
Cluster Related Hardware and Software	*SMCC SPARC Hardware Platform Guide Solaris 2.6*	805-7289
	Sun StorEdge Volume Manager 2.6 User's Guide	805-5705
	Sun StorEdge Volume Manager 2.6 Installation Guide	805-5707
	Sun StorEdge Volume Manager 2.6 Release Notes	805-5708
	Sun StorEdge Volume Manager 2.6 System Administrator's Guide	805-5706
	Sun StorEdge Volume Manager Storage Administrator 1.0 User's Guide	805-5709
	Sun StorEdge Volume Manager Storage Administrator 1.0 Release Notes	805-5710
SunVTS Diagnostics	*SunVTS Version 2.1 User's Guide Solaris 2.6*	805-3011
Oracle7™ Parallel Server	*Oracle7 for Sun SPARC Solaris 2.x Installation and Configuration Guide, Release 7.x*	A23325
	Oracle7 Parallel Server Administrator's Guide, Release 7.x	A12852
Oracle8™ Parallel Server	*Oracle8 Parallel Server Concepts & Administration, Release 8.0*	A54639
	Oracle Parallel Server Administrator's Guide, Release 8.0	A54641

Ordering Sun Documents

SunDocs^{SM} is a distribution program for Sun Microsystems technical documentation. Contact SunExpress for easy ordering and quick delivery. You can find a listing of available Sun documentation on the World Wide Web.

TABLE P–5 SunExpress Contact Information

Country	Telephone	Fax
Belgium	02-720-09-09	02-725-88-50
Canada	1-800-873-7869	1-800-944-0661
France	0800-90-61-57	0800-90-61-58
Germany	01-30-81-61-91	01-30-81-61-92
Holland	06-022-34-45	06-022-34-46
Japan	0120-33-9096	0120-33-9097
Luxembourg	32-2-720-09-09	32-2-725-88-50
Sweden	020-79-57-26	020-79-57-27
Switzerland	0800-55-19-26	0800-55-19-27
United Kingdom	0800-89-88-88	0800-89-88-87
United States	1-800-873-7869	1-800-944-0661
World Wide Web: http://www.sun.com/sunexpress/		

Sun Documentation on the Web

The docs.sun.com web site enables you to access Sun technical documentation on the World Wide Web. You can browse the docs.sun.com archives or search for a specific book title or subject at http://docs.sun.com.

Sun Welcomes Your Comments

We are interested in improving our documentation and welcome your comments and suggestions. You can email your comments to us at smcc-docs@sun.com. Please include the part number of your document in the subject line of your email.

CHAPTER **1**

Installation and Release Notes

This chapter provides the installation and release information for Cluster Volume Manager (CVM) Release 2.2.1 for Solaris 2.6.

This release of CVM has been tested on, and requires, Solaris 2.6.

CVM is a cluster-aware version of Veritas Volume Manager (VxVM), and is designed to be used in Oracle Parallel Server configurations.

This document is a supplement to the Sun StorEdge Volume Manager Release Notes and Installation Guide. For general information on SSVM, refer to the Sun StorEdge Volume Manager 2.6 manual set.

Note - Before you attempt to install the package, you should read this entire document.

The document covers the following topics:

- Section 1.1 “Getting Help” on page 1-1
- Section 1.2 “New Features and Changes” on page 1-2
- Section 1.3 “The Visual Administrator and CVM” on page 1-2
- Section 1.4 “Installing Cluster Volume Manager” on page 1-3
- Section 1.5 “Software Limitations and Known Problems” on page 1-10

1.1 Getting Help

For additional information contact your local Enterprise Service representative or Enterprise Service Authorized Service Provider.

1.2 New Features and Changes

For information on the features associated with Cluster Volume Manager, refer to Chapter 2. New features in Release 2.2.1 include:

- CVM is now compatible with the Sun StorEdge™ A3000.

  Before using CVM with the Sun StorEdge A3000, refer to the *Sun StorEdge Volume Manager Release Notes* (Release 2.5 or 2.6 as applicable) for information on how to install and use SSVM (and CVM) with the Sun StorEdge A3000.

- CVM now supports up to four nodes per cluster. However, storage devices that are physically connected to all nodes are required for shared disk groups to work on more than two nodes.

1.3 The Visual Administrator and CVM

The Visual Administrator provides the following support for CVM:

- The Visual Administrator root window highlights shared disk group view buttons with a green shaded border (to distinguish them from unshared disk groups). In monochrome, a shared disk group button has a grey shaded border instead.
- The view window for a shared disk group displays the string `Shared Disk Group` in its title bar. On color monitors, the background color for a shared disk group view is green; on monochrome monitors, there is no change in the background color.
- A disk group can be initialized as cluster-shareable for an active cluster. To accommodate this, the Initialize Disk Group form contains a Shared disk group: field that can be set to `Yes` or `No`. If set to `Yes`, the disk group is defined as cluster-shareable upon initialization. The system administrator is responsible for ensuring that disks specified as members of the cluster-shareable disk group are physically accessible from the hosts that make up the cluster.
- A disk group can be imported as shared through the Visual Administrator. To accommodate this, the Import Disk Group form has a Shared disk group: field that can be set to `Yes` or `No`. If set to `Yes`, the disk group is imported as cluster-sharable. This is only valid if the cluster is active on the host where the import takes place. The administrator is responsible for ensuring that all disks in a shared disk group are physically accessible by all hosts; a host that cannot access all disks in a shared disk group cannot join the cluster.

1.4 Installing Cluster Volume Manager

This section describes how to install or update CVM. The packages on the CD-ROM can be installed on systems running Solaris 2.6. CVM 2.2.1 requires Sun Cluster 2.2 software. Complete the installation or upgrade to Sun Cluster 2.2 before attempting to install CVM 2.2.1.

CVM installation consists of two parts:

1. Installing the combined package onto the system. Refer to Section 1.4.2 “Installing CVM for the First Time” on page 1-4, or Section 1.4.3 “Upgrading to CVM Release 2.2.1” on page 1-4.
2. Configuring and setting up CVM. Refer to Section 1.4.4 “Creating rootdg” on page 1-6, and Section 1.4.5 “Configuring Shared Disks” on page 1-7.

1.4.1 Pre-installation

If you are installing CVM for the first time, refer to the *Sun StorEdge Volume Manager Installation Guide* for additional pre-installation information.

- Most of the commands involved in the installation of CVM are in the `/sbin` or `/usr/sbin` directories. You should add these directories to your `PATH` environment variable.

  If you are using a Bourne Shell (`sh` or `ksh`), use the command:

```
PATH=/sbin:/usr/sbin:$PATH export PATH
```

If you are using a C Shell (`csh` or `tcsh`), use the command:

```
setenv PATH /sbin:/usr/sbin:${PATH}
```

- A system using CVM has one or more disk groups, including the root disk group (`rootdg`). The `rootdg` must exist and cannot be shared between systems. At least one disk must exist within `rootdg` while CVM is running. Before installing CVM, you should decide where to place `rootdg` for each node in the cluster.

  You can create `rootdg` by encapsulating the root disk as described in Section 1.4.4 “Creating rootdg” on page 1-6. Before beginning the installation, you must decide on the layout of shared disk groups. There may be one or more shared disk groups.
- If you plan to use Dirty Region Logging (DRL) with CVM, consider leaving a small amount of space on the disk for these logs. The log size is proportional to the volume size and the number of nodes (each log has one recovery map plus one active map per node).

For a two-gigabyte volume in a two-node cluster, a log size of five blocks (one block per map) would be required. For every additional two gigabytes of volume size, the log size should then increase by approximately one block per map (so a four-gigabyte volume with two nodes would have a log size of ten blocks) up to a maximum of 96 blocks. For larger volumes, DRL changes the log granularity to accommodate the increased size without exceeding the maximum log size. A four-node cluster requires larger logs. See Section 2.1.4 “Dirty Region Logging and CVM” on page 2-8, for more information about log sizes.

Note - To use CVM with a SPARCstorage™ Array, you must use firmware level 3.4 or later.

1.4.2 Installing CVM for the First Time

Note - CVM Release 2.2.1 requires Solaris 2.6, so it may be necessary to upgrade the operating environment before you install CVM.

1. **Load and mount the CVM 2.2.1 CD-ROM.**

   It should then be visible as the file system mounted on `/cdrom`.

2. **Go to the directory containing the CVM packages:**

```
# cd /cdrom/cdrom0/CVM_2_2_1/Product
```

3. **Use** `pkgadd` **to install the following packages:**

```
# pkgadd -d . SUNWvxvm SUNWvxva SUNWvmman SUNWvmdev
```

Caution - Packages must be installed in the order specified.

4. **Go to Section 1.4.4 “Creating rootdg” on page 1-6, to proceed with the CVM installation.**

1.4.3 Upgrading to CVM Release 2.2.1

Note that CVM Release 2.2.1 running on Sun Cluster 2.2 requires Solaris 2.6, so it may be necessary to upgrade the operating environment at the same time. The recommended procedure is to upgrade the operating environment first (if needed), then install or upgrade to Sun Cluster 2.2, and finally, upgrade CVM.

Note - If you have encapsulated one or more disks you must execute through Step 6 on page @–5 before doing the operating environment upgrade.

1.4.3.1 Upgrading From Earlier Versions of CVM

If you have Sun Cluster 2.0 or 2.1 installed, upgrade the CVM software and the operating environment as follows:

1. **Make sure you have enough space in** `/opt` **to upgrade the operating environment.**

2. **If any of the file systems** `/`**,** `/usr`**,** `/var`**, or** `/opt` **are defined on volumes, make sure that at least one plex for each of those volumes is formed from a single subdisk that begins on a cylinder boundary.**

   This is a required step. Part of the upgrade process includes temporarily placing file systems onto volumes that are using direct disk partitions. The Solaris operating environment requires that disk partitions start on cylinder boundaries. This conversion is handled automatically by the upgrade scripts, as necessary. If the upgrade scripts detect any problems (such as lack of cylinder alignment), the scripts display an explanation of the problem and the upgrade process stops.

3. **Load and mount the CVM 2.2.1 CD-ROM.**

   It should then be visible as the file system mounted on `/cdrom`.

4. **Run the** `upgrade_start` **script to prepare the previous release of CVM for its removal:**

   ```
   # /cdrom/cdrom0/CVM_2_2_1/Tools/scripts/upgrade_start
   ```

   The `upgrade_start` script looks for volumes containing file systems. If certain key file systems must be converted back to using partitions, this script handles the conversions.

5. **Reboot to single-user mode (using a command such as** `uadmin 2 3`**).**

6. **Remove the volume manager package(s).**

   ```
   phys-hahost1# pkgrm SUNWvmdev SUNWvmman SUNWvxva SUNWvxvm
   ```

7. **Shut down and halt the machine (using a command such as** `uadmin 2 0`**).**

8. **(Optional) Upgrade the operating environment to Solaris 2.6, if necessary.**

   Refer to the Solaris installation documentation for instructions on how to upgrade the Solaris software environment.

9. **Go to the directory containing the CVM packages (on the CVM CD-ROM):**

```
# cd /cdrom/cdrom0/CVM_2_2_1/  Product/
```

10. **Use** pkgadd **to install the following packages:**

```
# pkgadd -d . SUNWvxvm SUNWvxva SUNWvmman SUNWvmdev
```

11. **Complete the upgrade by entering:**

```
# /cdrom/cdrom0/CVM_2_2_1/Tools/scripts/upgrade_finish
```

12. **Reboot to multi-user mode.**

    At this point, your pre-upgrade configuration should be in effect and any file systems previously defined on volumes should be defined and mounted.

13. **Go to Section 1.4.4 “Creating rootdg” on page 1-6, to proceed with the CVM installation.**

1.4.4 Creating rootdg

After loading the CVM software, you must create the default disk group, `rootdg`. One approach is to place the root disk under CVM control through the process of encapsulation. The disk group resulting from the encapsulation will then become the `rootdg` disk group. However, if frequent upgrades to this package are anticipated, this may not be convenient because it is more difficult to upgrade to new versions or recover from certain errors when the root disk is encapsulated. If you do not wish to encapsulate the root disk, you can encapsulate any other disk using `vxinstall` to create the rootdg (required for CVM to come up. Another approach is to create a simple volume manager disk (on a partition of a disk that is not shared and has not been encapsulated) and then use this for `rootdg`.

This section describes how to create the root disk group by using encapsulation; we do not recommend using the simple disk approach. After creating `rootdg`, go to Section 1.4.5 “Configuring Shared Disks” on page 1-7.

1.4.4.1 Encapsulating Your Root Disk

To encapsulate your root disk, create `rootdg` as follows:

1. **Invoke** `vxinstall` **and follow the instructions in the "Custom Installation" section of the *Sun StorEdge Volume Manager 2.6 Installation Guide* to encapsulate only the root disk.**

   For all other disks, select the `Leave these disks alone` option.

2. **After using** `vxinstall` **to encapsulate the root disk, reboot the system.**

   The `vxinstall` command will automatically create `rootdg.`

1.4.5 Configuring Shared Disks

If you are installing CVM for the first time or adding disks to an existing cluster, you must configure new shared disks. If you are upgrading CVM, verify that your shared disks still exist.

Note - The shared disks should be configured from one node only. Since the CVM software cannot tell whether a disk is shared or not, you must specify which are the shared disks.

Make sure that nobody else is accessing the shared disks from another node while you are performing the configuration.

▼ How to Verify Existing Shared Disks

If you are upgrading from a previous release of CVM to CVM 2.2.1, verify that your shared disk groups still exist:

1. **Start the cluster on all nodes.**

2. **Type the following command on all nodes**:

```
# vxdg list
```

   This should display the shared disk groups that existed before. DRL logs that were created with earlier versions of CVM may be too small for CVM 2.2.1. For additional information, refer to Section 2.1.4 "Dirty Region Logging and CVM" on page 2-8.

▼ How to Convert Existing SEVM 2.x Disks to Shared Disks

If you are upgrading from SEVM 2.x to CVM 2.2.1 and want to share existing disk groups, configure the shared disks as follows:

1. **Start the cluster on at least one node.**

   For a two-node cluster, start the cluster on one node; for a four-node cluster, start the cluster on three nodes.

2. **List all disk groups**:

   ```
   # vxdg list
   ```

3. **Deport disk groups to be shared**:

   ```
   # vxdg deport groupname
   ```

4. **Import disk groups to be shared**:

   ```
   # vxdg -s import groupname
   ```

   This will mark the disks in the shared disk groups as shared and stamp them with the ID of the cluster, enabling other nodes to recognize the shared disks.

   If there are dirty region logs, make sure they are active. If not, replace them with bigger ones.

5. **Display the shared flag for all the shared disk groups**:

   ```
   # vxdg list
   ```

   The disk groups are now ready to be shared.

6. **If the cluster is running with one node only, bring up the other cluster nodes.**

7. **When the each node is ready, enter the command** `vxdg list` **on it.**

   This should display the same list of shared disk groups that appeared earlier.

▼ How to Configure New Disks

If you are installing and setting up CVM for the first time, configure the shared disks as follows:

1. **Start the cluster on at least one node. If the cluster contains more than one node, perform Steps 3 and 4 only on the master node.** `vxdctl -c mode` **reports the operating mode of CVM.**

2. **Run** `vxdisksetup` **to initialize each shared disk on any node; run** `vxdctl enable` **on all nodes afterwards.**

   If you have decided not to put configuration information on every disk, or if you want larger areas for this information, `vxdisksetup` enables you to specify your choices.

3. **Create disk groups on the shared disks.**

   You can use `vxdg` or the Visual Administrator to do this. Use the `-s` option of `vxdg` to create shared disk groups.

4. **Create volumes in the disk groups.**

   You can use `vxassist` or the Visual Administrator to do this.

   The volumes must be of type `gen.` Do not create RAID5 volumes. Before creating any log subdisks, read Section 2.1.4 “Dirty Region Logging and CVM” on page 2-8.

5. **If the cluster is running with one node only, bring up the other cluster nodes.**

   When each node is ready, enter the command `vxdg list` on it. This should display the same list of shared disk groups that appeared earlier.

1.4.6 Disk Reservation

This section is applicable for two-node configurations only.

As part of failure fencing, Sun Cluster reserves shared disks when only one node is active. This prevents “rogue” hosts from accessing the shared disks. When this happens, the command `vxdisk list` on a node that has left the cluster may show all disks on such a controller as having an `error` status. The more detailed options of `vxdisk` will show the flag `unavailable`. When a new node joins the cluster, the Sun Cluster software releases the controllers. CVM attempts to access these disks, and if that is successful, the disks return to an `online` status. If one system boots while the other system has the disks reserved, the disks may be invisible to the booting system, and `vxdisk` may display none of the shared disks. When the system joins the cluster, the shared disks become visible.

1.5 Software Limitations and Known Problems

The following caveats and usage issues are known for this release of CVM Release 2.2.1:

- If CVM has deported a disk group because the disk group has lost access to one or more of its disks (due to a node leaving the cluster), the only way to try to regain access to the deported disks that are still attached to nodes in the cluster is to force-import the deported disk group. However, forcing an import in this situation is dangerous because it can cause mirrors to become unsynchronized in such a way that it cannot be determined which mirror has correct data.
- It is possible to have private (non-shared) disk groups on physically shared disks. If these disks are on controllers that have been designated for fencing (for example, reserved by Sun Cluster), the owner of the private disk group may not be able to access it when it is not in the cluster.
- CVM does not currently support RAID5 volumes.
- Only `gen` volume types are supported in shared disk groups. The use of `fsgen` volumes can cause system deadlocks.
- When a node leaves the cluster due to clean shutdown or abort, the surviving node performs a cluster reconfiguration. If the leaving node attempts to rejoin before the cluster reconfiguration is complete, the outcome depends on whether the leaving node is a slave or master.

  If the leaving node is a slave, the attempt will fail with one of the following pairs of error messages:

```
Resource temporarily unavailable

[vxclust] return from cluster_establish is configuration
daemon error -1
Resource temporarily unavailable

master has disconnected
```

A retry at a later time should succeed.

If the leaving node is a master, the attempt will generate disk-related error messages on both nodes and the remaining node will abort. The joining node will eventually join and may become master.

- If `vxconfigd` is stopped on both the master and slave nodes and then restarted on the slave first, its displays will not be reliable until `vxconfigd` has started on the master and the slave has reconnected (which may take about 30 seconds). In particular, shared disk groups will be marked “disabled” and no information about them will be available. vxconfigd should therefore be started on the master first.

- When a node aborts from the cluster, open volume devices in shared disk groups on which I/O is not active are not removed until the volumes are closed. If this node later joins the cluster as the master while these volumes are still open, the presence of these volumes does not cause a problem. However, if the node tries to rejoin the cluster as a slave, this may fail with the error message:

```
cannot assign minor #
```

This is accompanied by the console message:

```
WARNING:minor number ### disk group group in use
```

- The current disk hot-sparing mechanism does not work well for *partial* disk failures. The model was written using the presumption that disks fail totally, rather than partially, and that partial errors can usually be fixed by writing back a failing sector. Usually this is a good assumption, but some users have encountered situations where only a few sectors failed and hot-sparing did not occur.

- When vxconfigd is stopped and restarted, it may disable large disk groups (for example, disk groups containing hundreds of volumes).

  Workaround: Restart vxconfigd with the cleartempdir option. If needed, deport and reimport the disk groups and start all volumes.

- Under some circumstances, a node abort may lead to a panic. This is relatively rare, but can occur- if I/O cannot be quiesced in a timely manner and the node needs to be brought down to ensure data integrity.

CHAPTER 2

Cluster Volume Manager

This chapter provides an overview of Cluster Volume Manager (CVM), utilities and daemons, error messages, troubleshooting, and terminology used with CVM.

The following topics are covered in this document:

This document assumes that the system administrator has a general understanding of the Sun StorEdge Volume Manager (SSVM). Refer to the SSVM documentation set for more information.

2.1 Cluster Volume Manager Overview

Cluster Volume Manager (CVM) is an extension of SSVM that enables multiple hosts to simultaneously access and manage a given set of disks under volume manager control. A *cluster* is a set of hosts sharing a set of disks; each host is referred to as a *node* in the cluster. The nodes are connected across a network. If one node fails, the other node(s) can still access the disks. CVM presents the same logical view of the disk configurations (including changes) on all nodes.

Note - CVM currently supports up to four nodes per cluster.

2.1.1 Shared Volume Manager Objects

When used with CVM, volume manager objects are capable of being shared by all of the nodes in a given cluster.

CVM allows for two types of disk groups:

- *Private disk groups*, which belong to only one node. A private disk group is only imported by one system. Disks in a private disk group can be physically accessible from one or more systems, but actual access is restricted to one system only.
- *Cluster-shareable disk groups*, which are shared by all nodes. A cluster-shareable (or *shared*) disk group is imported by all cluster nodes. Disks in a cluster-shareable disk group must be physically accessible from all systems that can join the cluster.

With CVM, most disk groups are shared. However, the root disk group (`rootdg`) is always a private disk group.

The volume manager disks within a shared disk group are shared by all of the nodes in a cluster, so multiple nodes in a cluster can access a given volume manager disk simultaneously. Likewise, all volumes in a shared disk group can be shared by all nodes in a cluster. A volume manager disk or volume is considered *shared* when it is accessed by more than one node at the same time.

Note - Only raw device access is performed through CVM. CVM does not support shared volumes with file systems. CVM does not currently support RAID5 volumes in cluster-shareable disk groups. RAID5 volumes can, however, be used in private disk groups attached to specific nodes of a cluster.

2.1.2 How Cluster Volume Management Works

CVM works together with an externally-provided *cluster manager*, which is a daemon that informs CVM of changes in cluster membership. Each node starts up independently and has its own copies of the operating environment, CVM, and the cluster manager. When a node *joins* a cluster, it gains access to shared disks. When a node *leaves* a cluster, it no longer has access to those shared disks. The system administrator joins a node to a cluster by starting the cluster manager on that node.

Figure 2–1 illustrates a simple cluster arrangement. All of the nodes are connected by a network. The nodes are then connected to a cluster-shareable disk group. To the cluster manager, all nodes are the same. However, CVM requires that one node act as the *master node*; the other nodes are *slave nodes*. The master node is responsible for coordinating certain volume manager activities. CVM software determines which node performs the master function (any node is capable of being a master node); this role only changes if the master node leaves the cluster. If the master leaves the cluster, one of the slave nodes becomes the new master. In this example, Node 1 is the master node and Node 2, Node 3, and Node 4 are the slave nodes.

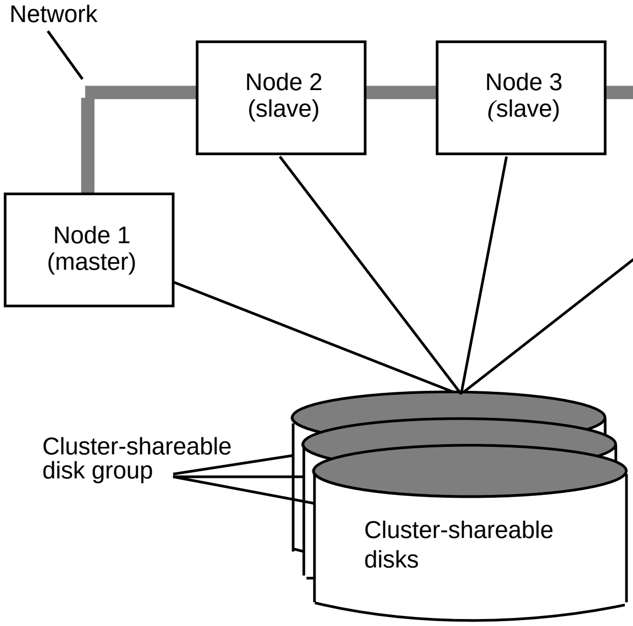

Figure 2–1 Example of a Four-Node Cluster

The system administrator designates a disk group as cluster-shareable using the vxdg utility. Refer to Section 2.2.3 “The vxdg Command” on page 2-13 for further information. Once a disk group is imported as cluster-shareable for one node, the disk headers are marked with the cluster ID. When other nodes join the cluster, they will recognize the disk group as being cluster-shareable and import it. The system administrator can import or deport a shared disk group at any time; the operation takes places in a distributed fashion on all nodes.

Each physical disk is marked with a unique disk ID. When the cluster starts up on the master, it imports all the shared disk groups (except for any that have the `noautoimport` attribute set). When a slave tries to join, the master sends it a list of the disk IDs it has imported and the slave checks to see if it can access all of them. If the slave cannot access one of the imported disks on the list, it abandons its attempt to join the cluster. If it can access all of the disks on the list, it imports the same set of shared disk groups as the master and joins the cluster. When a node leaves the cluster, it deports all its imported shared disk groups, but they remain imported on the surviving nodes. This is true if either type of node, master or slave, leaves the cluster.

Any reconfiguration to a shared disk group is performed with the cooperation of all nodes. Configuration changes to the disk group happen simultaneously on all nodes and the changes are identical. These changes are atomic in nature, so they either occur simultaneously on all nodes or do not occur at all.

All members of the cluster have simultaneous read and write access to any cluster-shareable disk group. Access by the active nodes of the cluster is not affected by a failure in any other node. The data contained in a cluster-shareable disk group is available as long as at least one node is active in the cluster. Regardless of which node accesses the cluster-shareable disk group, the configuration of the disk group looks the same. Applications running on each node can access the data on the volume manager disks simultaneously.

Note - CVM does not protect against simultaneous writes to shared volumes by more than one node. It is assumed that any consistency control is done at the application level (using a distributed lock manager, for example).

2.1.2.1 Configuration and Initialization

Before any nodes can join a new cluster for the first time, the system administrator must supply certain configuration information. This information is supplied during cluster manager setup and is normally stored in some type of cluster manager configuration database. The precise content and format of this information is dependent on the characteristics of the cluster manager.

The type of information required by CVM is as follows:

- Cluster ID
- Node IDs
- Network addresses of nodes
- Port addresses

When a node joins the cluster, this information is automatically loaded into CVM on that node at node startup time.

Node initialization is effected through the cluster manager startup procedure, which brings up the various cluster components (such as CVM, the cluster manager, and a distributed lock manager) on the node. Once initialization is complete, applications can be started. The system administrator invokes the cluster manager startup procedure on each node to be joined to the cluster.

For CVM, initialization consists of loading the cluster configuration information and joining the nodes in the cluster. The first node to join becomes the master node, and later nodes (slaves) join to the master. If two nodes join simultaneously, CVM software chooses the master. Once the join for a given node is complete, that node has access to the shared disks.

2.1.2.2 Cluster Reconfiguration

Any time there is a change in the state of the cluster (in the form of a node leaving or joining), a *cluster reconfiguration* occurs. The cluster manager for each node monitors other nodes in the cluster and calls the cluster reconfiguration utility, `vxclust`, when there is a change in cluster membership. `vxclust` coordinates cluster reconfigurations and provides communication between CVM and the cluster manager. The cluster manager and `vxclust` work together to ensure that each step in the cluster reconfiguration is completed in the correct order.

During a cluster reconfiguration, I/O to shared disks is suspended. It is resumed when the reconfiguration completes. Applications can therefore appear to be frozen for a short time.

If other operations (such as volume manager operations or recoveries) are in progress, the cluster reconfiguration can be delayed until those operations have completed. Volume reconfigurations (described later) do not take place at the same time as cluster reconfigurations. Depending on the circumstances, an operation can be held up and restarted later. In most cases, cluster reconfiguration takes precedence. However, if the volume reconfiguration is in the commit stage, it will complete first.

For further information on cluster reconfiguration, refer to Section 2.2.1 “The `vxclust` Command” on page 2-11.

2.1.2.3 Volume Reconfiguration

Volume reconfiguration is the process of creating, changing, and removing the volume manager objects in the configuration (such as disk groups, volumes, mirrors, and so forth.). In a cluster, this process is performed with the cooperation of all nodes. Volume reconfiguration is distributed to all nodes; identical configuration changes occur on all nodes simultaneously.

The `vxconfigd` daemons play an active role in volume reconfiguration. For the reconfiguration to succeed, `vxconfigd` must be running on all nodes.

The reconfiguration is initiated and coordinated by the *initiating node*, which is the node on which the system administrator is running the utility that is requesting changes to volume manager objects.

The utility on the initiating node contacts its local `vxconfigd` daemon, which performs some local checking to make sure that a requested change is reasonable. For instance, it will fail an attempt to create a new disk group when one with the same name already exists. `vxconfigd` on the initiating node then sends messages with the details of the changes to the `vxconfigd` daemons on all other nodes in the cluster. The `vxconfigd`s on each of the non-initiating nodes then perform their own checking. For example, a non-initiating node checks that it does not have a private disk group with the same name as the one being created; if the operation involves a new disk, each node checks that it can access that disk. When all of the `vxconfigd`s on all the nodes agree that the proposed change is reasonable, each `vxconfigd` notifies its kernel and the kernels then cooperate to either commit or abort the transaction. Before the transaction can commit, all of the kernels ensure that no I/O is underway. The master is responsible for initiating a reconfiguration and coordinating the transaction commit.

If `vxconfigd` on any node goes away during a reconfiguration process, all nodes will be notified and the operation will fail. If any node leaves the cluster, the operation will fail unless the master has already committed the operation. If the master leaves the cluster, the new master (which was a slave previously) either completes or fails the operation. This depends on whether or not it received notification of successful completion from the previous master. This notification is done in such a way that if the new master did not receive it, neither did any other slave.

If a node attempts to join the cluster while a volume reconfiguration is being performed, the results depend on how far the reconfiguration has progressed. If the kernel is not yet involved, the volume reconfiguration is suspended and restarted when the join is complete. If the kernel is involved, the join waits until the reconfiguration is complete.

When an error occurs (such as when a check on a slave fails or a node leaves the cluster), the error is returned to the utility and a message is issued to the console on the initiating node to identify the node on which the error occurred.

2.1.2.4 Node Shutdown

The system administrator can shut down the cluster on a given node by invoking the cluster manager shutdown procedure on that node. This terminates cluster components after cluster applications have been stopped. CVM supports *clean node shutdown*, which is the ability of a node to leave the cluster gracefully when all access to shared volumes has ceased. The host is still operational, but cluster applications cannot be run on it.

CVM maintains global state information for each volume. This enables CVM to accurately determine which volumes need recovery when a node crashes. When a node leaves the cluster due to a crash or by some other means that is not clean, CVM determines which volumes can have writes that have not completed and the master resynchronizes those volumes. If Dirty Region Logging (DRL) is active for any of those volumes, it will be used.

Clean node shutdown should be used after, or in conjunction with, a procedure to halt all cluster applications. Depending on the characteristics of the clustered application and its shutdown procedure, it can be a long time before the shutdown is successful (minutes to hours). For instance, many applications have the concept of *draining*, where they accept no new work, but complete any work in progress before exiting. This process can take a long time if, for instance, a long-running transaction is active.

When the CVM shutdown procedure is invoked, it checks all volumes in all shared disk groups on the node that is being shut down and then either proceeds with the shutdown or fails:

- If all volumes in shared disk groups are closed, CVM makes them unavailable to applications. Since it is known on all nodes that these volumes are closed on the leaving node, no resynchronizations are performed.
- If any volume in a shared disk group is open, the CVM shutdown procedure returns failure. The shutdown procedure can be retried repeatedly until it succeeds. There is no timeout checking in this operation—it is intended as a service that verifies that the clustered applications are no longer active.

Note - Once the shutdown has succeeded and the node has left the cluster, it is not possible to access the shared volumes until it joins the cluster again.

Since shutdown can be a lengthy process, other reconfigurations can take place while shutdown is in progress. Normally, the shutdown attempt is suspended until the other reconfiguration completes. However, if it is too far advanced, the shutdown can complete first.

2.1.2.5 Node Abort

If a node does not leave cleanly, it is either because the host crashed or because some cluster component decided to make the node leave on an emergency basis. The ensuing cluster reconfiguration calls the CVM abort function. This function makes an attempt to halt all access to shared volumes at once, though the operation waits until I/O that is at the disk completes. I/O operations that have not yet been started are failed, and the shared volumes are removed. Applications that were accessing the shared volumes, therefore, fail with errors.

After a node abort or crash, the shared volumes must be recovered (either by a surviving node or by a subsequent cluster restart) because it is very likely that there are unsynchronized mirrors.

2.1.2.6 Cluster Shutdown

When all the nodes in the cluster leave, the determination of whether or not the shared volumes should be recovered has to be made at the next cluster startup. If all nodes left cleanly, there is no need for recovery. If the last node left cleanly and resynchronization resulting from the non-clean leave of earlier nodes was complete, there is no need for recovery. However, recovery must be performed if the last node did not leave cleanly or if resynchronization from previous leaves was not complete.

2.1.3 Disks and CVM

The nodes in a cluster must always agree on the status of a disk. In particular, if one node cannot write to a given disk, all nodes must stop accessing that disk before the results of the write operation are returned to the caller. Therefore, if a node cannot contact a disk, it should contact another node to check on the disk's status. If the disk has failed, no node will be able to access it and the nodes can agree to detach the disk. If the disk has not failed, but rather the access paths from some of the nodes have failed, the nodes cannot agree on the status of the disk. Some policy must exist to resolve this type of discrepancy. The current policy is to detach the disk, even though it has not failed. While this resolves the status issue, it leaves mirrored disks with fewer mirrors and non-mirrored volumes with no access to data.

Note - This policy is the current implementation option. In a future release of CVM, the system administrator will be able to select another policy. For example, an alternative policy would be for any node that cannot access the disk to leave the cluster; this maintains integrity of data access at the cost of keeping some nodes outside the cluster.

2.1.4 Dirty Region Logging and CVM

Dirty Region Logging (DRL) is an optional property of a volume that provides speedy recovery of mirrored volumes after a system failure. Dirty Region Logging is supported in cluster-shareable disk groups. This section provides a brief overview of DRL and outlines differences between SSVM DRL and the CVM implementation of DRL. For more information on DRL, refer to Chapter 1 in the applicable *Sun StorEdge Volume Manager 2.6 System Administrator's Guide.*

DRL keeps track of the regions that have changed due to I/O writes to a mirrored volume and uses this information to recover only the portions of the volume that need to be recovered. DRL logically divides a volume into a set of consecutive regions and maintains a dirty region log that contains a status bit representing each region of the volume. *Log subdisks* store the dirty region log of a volume that has DRL enabled. A volume with DRL has at least one log subdisk, that is associated with one of the volume's plexes.

Before writing any data to the volume, the regions being written are marked dirty in the log. If a write causes a log region to become dirty when it was previously clean, the log is synchronously written to disk before the write operation can occur. On system restart, the volume manager recovers only those regions of the volume which are marked as dirty in the dirty region log.

The CVM implementation of DRL differs slightly from the SSVM implementation. The following sections outline some of the differences and discuss some aspects of the CVM implementation.

2.1.4.1 Log Format and Size

As with SSVM, the CVM dirty region log exists on a log subdisk in a mirrored volume.

A SSVM dirty region log has a recovery map and a single active map. A CVM dirty region log, however, has one recovery map and multiple active maps (one for each node in the cluster). Unlike SSVM, CVM places its recovery map at the beginning of the log.

The CVM dirty region log size is typically larger than a SSVM dirty region log, as it must accommodate active maps for all nodes in the cluster plus a recovery map. The size of each map within the dirty region log is one or more whole blocks. `vxassist` automatically takes care of allocating a sufficiently large dirty region log.

The log size depends on the volume size and the number of nodes. The log must be large enough to accommodate all maps (one map per node plus a recovery map). Each map should be one block long for each two gigabytes of volume size. For a two-gigabyte volume in a two-node cluster, a log size of five blocks (one block per map) should be sufficient; this is the minimum log size. A four-gigabyte volume in a four-node cluster requires a log size of ten blocks, and so on.

2.1.4.2 Compatibility

Except for the addition of a CVM specific magic number, CVM DRL headers are the same as their SSVM counterparts.

It is possible to import a SSVM disk group (and its volumes) as a CVM shared disk group and vice versa. However, the dirty region logs of the imported disk group can be considered invalid and a full recovery can result.

If a CVM shared disk group is imported by a SSVM system, SSVM will consider the logs of the CVM volumes to be invalid and will conduct a full volume recovery. After this recovery completes, the volume manager will use the CVM Dirty Region Logging.

CVM is capable of performing a DRL recovery on a non-shared SSVM volume. However, if a SSVM volume is moved to a CVM system and imported as shared, the dirty region log will probably be too small to accommodate all the nodes in the cluster. CVM will therefore mark the log invalid and do a full recovery anyway. Similarly, moving a DRL volume from a two-node cluster to a four-node cluster will result in too small a log size, which CVM will handle with a full volume recovery. In both cases, the system administrator is responsible for allocating a new log of sufficient size.

2.1.4.3 Recovery With DRL

When one or more nodes in a cluster crash, DRL needs to be able to handle the recovery of all volumes in use by those nodes when the crash(es) occurred. On initial cluster startup, all active maps are incorporated into the recovery map; this is done during the `volume start` operation.

Nodes that crash (that is, leave the cluster as *dirty*) are not allowed to rejoin the cluster until their DRL active maps have been incorporated into the recovery maps on all affected volumes. The recovery utilities compare a crashed node's active maps with the recovery map and make any necessary updates before the node can rejoin the cluster and resume I/O to the volume (which overwrites the active map). During this time, other nodes can continue to perform I/O.

The CVM kernel tracks which nodes have crashed. If multiple node recoveries are underway in a cluster at a given time, their respective recoveries and recovery map updates can compete with each other. The CVM kernel therefore tracks changes in the DRL recovery state and prevents I/O operation collisions.

The master performs volatile tracking of DRL recovery map updates for each volume and prevents multiple utilities from changing the recovery map simultaneously.

2.2 Cluster Volume Manager Utilities and Daemons

The following utilities and/or daemons have been created or modified for use with CVM:

- `vxclust`
- `vxconfigd`
- `vxdg`
- `vxdisk`
- `vxrecover`
- `vxdctl`
- `vxstat`

The following sections contain information about how each of these utilities is used in a cluster environment. For further details on any of these utilities, refer to their manual pages.

Note - Most volume manager commands require superuser privileges.

2.2.1 The `vxclust` Command

Note - `vxclust` is not a portable utility. Although `vxclust` performs the same or similar operations in relation to the SSVM, it needs to be modified or rewritten to operate with each particular cluster manager.

`vxclust` is the volume manager cluster reconfiguration utility. `vxclust` coordinates changes in cluster membership and provides communication between the cluster manager and CVM. Whenever there is a cluster reconfiguration, `vxclust` is called by the cluster manager. `vxclust` notifies the volume manager kernel and the `vxconfigd` daemon whenever cluster reconfiguration takes place.

Every time there is a cluster reconfiguration, every node currently in the cluster runs the `vxclust` utility at each of several well-orchestrated steps. Cluster manager facilities ensure that the same step is executed on all nodes at the same time. A given step only starts when the previous one has completed on all nodes. At each step in the reconfiguration, `vxclust` determines what CVM should do next. After informing CVM of its next action, `vxclust` waits for the outcome (success, failure, or retry) and communicates that to the cluster manager.

If a node does not respond to a `vxclust` request within a specific timeout period, that node aborts. `vxclust` then decides whether to restart the reconfiguration or give up, depending on the circumstances. If the cause of the reconfiguration is a local, uncorrectable error, `vxclust` stops the action. If a node cannot complete an operation because another node has left, the surviving node times out. In this case, `vxclust` requests a reconfiguration with the expectation that another node will leave. If no other node leaves, `vxclust` will cause the local node to leave.

If a reconfiguration step fails, `vxclust` returns an error to the cluster manager. The cluster manager can decide to abort the node, causing its immediate departure from the cluster. Any I/O in progress to the shared disk fails and access to the shared disks is stopped.

`vxclust` decides what actions to take when it is informed of changes in the cluster. If a new master node is required (due to failure of the previous master), `vxclust` determines which node becomes the new master.

2.2.2 The vxconfigd Command

The volume manager configuration daemon, vxconfigd, maintains configurations of volume manager objects. vxconfigd receives cluster-related instructions from the vxclust utility. A separate copy of vxconfigd resides on each node; these copies communicate with each other through networking facilities. For each node in a cluster, Volume manager utilities communicate with the vxconfigd running on that particular node; utilities do not attempt to connect with vxconfigd daemons on other nodes. During startup of the cluster, vxclust tells vxconfigd to begin cluster operation and tells it whether it is a master or slave node.

When a node is initialized for cluster operation, vxclust tells the kernel and vxconfigd that the node is about to join the cluster and provides vxconfigd with the following information (from the cluster manager configuration database):

- Cluster ID
- Node IDs
- Master node ID
- Node's role
- Network address of the vxconfigd on each node

  On the master node, vxconfigd sets up the shared configuration (that is, imports the shared disk groups) and informs vxclust when it is ready for slaves to join.

  On slave nodes, vxclust tells the kernel and vxconfigd when the slave node can join the cluster. When the slave node joins the cluster, vxconfigd and the volume manager kernel communicate with their counterparts on the master in order to set up the shared configuration.

  When a node leaves the cluster, vxclust notifies the kernel and vxconfigd on all the other nodes. The master node then performs any necessary cleanup. If the master node leaves the cluster, vxclust chooses a new master node and notifies the kernel and vxconfigd on all nodes of the choice.

  vxconfigd also participates in volume reconfiguration. Refer to Section 2.1.2.3 "Volume Reconfiguration" on page 2-5, for information on vxconfigd's role in volume reconfiguration.

2.2.2.1 vxconfigd Recovery

The volume manager vxconfigd daemon can be stopped and/or restarted at any time. While vxconfigd is stopped, volume reconfigurations cannot take place and other nodes cannot join the cluster until vxconfigd is restarted. In the cluster, the vxconfigd daemons on the slaves are always connected to the vxconfigd daemon on the master. It is therefore not advisable to stop the vxconfigd daemon on any clustered node.

If vxconfigd is stopped for some reason, different actions are taken depending on which node has a stopped daemon:

- If vxconfigd is stopped on the slave(s), the master takes no action. When vxconfigd is restarted on the slave, the slave's vxconfigd attempts to reconnect to the master's and re-acquire the information about the shared configuration. (The view of the kernel of the shared configuration is unaffected, and so is access to the shared disks.) Until the slave vxconfigd has successfully rejoined to the master, it has very little information about the shared configuration and any attempts to display or modify the shared configuration can fail. In particular, if the shared disk groups are listed (using vxdg list), they will be marked as disabled; when the rejoin has completed successfully, they will be marked as enabled.
- If vxconfigd is stopped on the master, vxconfigd on the slave(s) attempts to rejoin to the master periodically. This will not succeed until vxconfigd is restarted on the master. In this case, the slave vxconfigd information about the shared configuration has not been lost, so configuration displays are accurate.
- If vxconfigd is stopped on both the master and the slave(s), the slave will not display accurate configuration information until vxconfigd has been restarted on both nodes and they have reconnected again.

  When vxclust notices that vxconfigd is stopped on a node, vxclust restarts vxconfigd.

Note - With SSVM, the -r reset option to vxconfigd restarts vxconfigd and creates all states from scratch. This option is not available while a node is in the cluster because it causes the loss of cluster information; if this option is used under these circumstances, vxconfigd will not start.

2.2.3 The vxdg Command

The vxdg utility manages volume manager disk groups. You can use vxdg to specify that a disk group is cluster-sharable. The -s option to vxdg is provided to initialize or import a disk group as "shared."

If the cluster software has been run to set up the cluster, create a shared disk group with the following command:

```
vxdg -s init diskgroup [medianame=]accessname
```

where *diskgroup* is the disk group name; *medianame* is the administrative name chosen for the disk; and *accessname* is the disk access name (or device name).

Disk groups can be imported as shared using vxdg -s import. If the disk groups were set up before the cluster software was run, you can import the disk groups into the cluster arrangement with the command:

```
vxdg -s import diskgroup
```

where *diskgroup* is the disk group name or ID. On subsequent cluster restarts, the disk group will automatically be imported as shared. Note that it can be necessary to deport the disk group (using `vxdg deport` *diskgroup*) before invoking this command.

A disk group can be converted from shared to private by deporting it through `vxdg deport` and then importing it with `vxdg import` *diskgroup*.

Note - The system cannot tell if a disk is shared. To protect data integrity when dealing with disks that can be accessed by multiple systems, an system administrator must be careful to use the correct designation when adding a disk to a disk group. If an administrator attempts to add a disk that is not physically shared to a shared disk group, the volume manager allows this on the node where the disk is accessible if that node is the only one in the cluster. However, other nodes will not be able to join the cluster. Furthermore, if an administrator attempts to add the same disk to different disk groups on two nodes at the same time, the results are undefined. All configurations should therefore be handled on one node only.

`vxdg` has a force option (`-f`) that can be used to force-import a disk group or force-add a disk to a disk group.

Note - The force option (`-f`) should be used with caution and should be used only if the system administrator is fully aware of the possible consequences.

When a cluster is restarted, CVM can refuse to auto-import a disk group for one of the following reasons:

- A disk in that disk group is no longer accessible because of hardware errors on the disk. In this case, you can import the disk group again with the force option as follows:

```
vxdg -s -f import diskgroup
```

- Some of the nodes to which disks in the disk group are attached are not currently in the cluster, so the disk group cannot access all of its disks. In this case, a forced import is unsafe and should not be attempted (because it can result in inconsistent mirrors).

  If CVM will not add a disk to an existing disk group (because that disk is not attached to the same node(s) as the other disks in the disk group), you can force-add the disk as follows:

```
vxdg -f adddisk -g diskgroup [medianame=]accessname
```

`vxdg` can also be used to list shared disk groups. The following command displays one line of information for each disk group:

```
vxdg list
```

The output from this command is as follows:

```
NAME         STATE             ID
rootdg       enabled           774215886.1025.teal
group2       enabled,shared    774575420.1170.teal
group1       enabled,shared    774222028.1090.teal
```

Shared disk groups are designated with the flag `shared`.

The following command displays one line of information for each shared disk group:

```
vxdg -s list
```

The output for this command is as follows:

```
NAME         STATE            ID
group2       enabled,shared   774575420.1170.teal
group1       enabled,shared   774222028.1090.teal
```

The following command shows information about one specific disk group, including whether it is shared or not:

```
vxdg list diskgroup
```

where *diskgroup* is the disk group name.

The output for `vxdg list group1` on the master (for the disk group `group1`) is as follows:

```
Group:     group1
dgid:      774222028.1090.teal
import-id: 32768.1749
flags:     shared
copies:    nconfig=default nlog=default
config:    seqno=0.1976 permlen=1456 free=1448 templen=6 loglen=220
config disk c1t0d0s2 copy 1 len=1456 state=clean online
config disk c1t1d0s2 copy 1 len=1456 state=clean online
log disk c1t0d0s2 copy 1 len=220
log disk c1t1d0s2 copy 1 len=220
```

Note that the `flags:` field is set to `shared`. The output for the same command is slightly different on a slave.

2.2.4 The vxdisk Command

The vxdisk utility manages volume manager disks. You can use vxdisk to determine whether a disk is part of a cluster-shareable disk group, as follows:

```
vxdisk list accessname
```

where *accessname* is the disk access name (or device name).

The output from this command (for the device c1t0d0s2) is as follows:

```
Device:    c1t0d0s2
devicetag: c1t0d0
type:      sliced
clusterid:    cvm
disk:      name=disk01 id=774215890.1035.teal
group:     name=group1 id=774222028.1090.teal
flags:     online ready private autoconfig shared imported
pubpaths:  block=/dev/dsk/c1t0d0s4 char=/dev/rdsk/c1t0d0s4
privpaths: block=/dev/dsk/c1t0d0s3 char=/dev/rdsk/c1t0d0s3
version:   2.1
iosize:    min=512 (bytes) max=248 (blocks)
public:    slice=4 offset=0 len=2050272
private:   slice=3 offset=1 len=2015
update:    time=778564769 seqno=0.1614
headers:   0 248
configs:   count=1 len=1456
logs:      count=1 len=220
Defined regions:
 config   priv 000017-000247[000231]: copy=01 offset=000000 enabled
 config   priv 000249-001473[001225]: copy=01 offset=000231 enabled
 log      priv 001474-001693[000220]: copy=01 offset=000000 enabled
```

Note that the clusterid: field is set to cvm (the name of the cluster) and the flags: field includes an entry for shared. When a node is not joined, the flags: field contains the autoimport flag instead of imported.

2.2.5 The vxrecover Command

The vxrecover utility recovers plexes and volumes after disk replacement.

When a node leaves the cluster, it can leave some mirrors in an inconsistent state. The vxrecover utility performs recovery on all volumes in this state. The -c option causes vxrecover to perform recovery for all volumes in cluster-shareable disk groups. vxclust automatically calls vxrecover -c, when necessary.

Note - While vxrecover is active, there can be some degradation in system performance.

2.2.6 The `vxdctl` Command

The `vxdctl` utility manages some aspects of the volume configuration daemon, `vxconfigd`. The `-c` option can be used to request cluster information. You can use `vxdctl` as follows to determine whether `vxconfigd` is enabled and/or running:

```
vxdctl -c mode
```

Depending on the circumstances, the output is similar to the following:

```
mode: enabled: cluster active - MASTER

mode: enabled: cluster active - SLAVE

mode: enabled: cluster inactive

mode: enabled: cluster active - role not set
```

Note - If `vxconfigd` is disabled, no cluster information is displayed.

2.2.7 The `vxstat` Command

`vxstat` returns statistics for specified objects. In the CVM environment, `vxstat` gathers statistics from all of the nodes in the cluster. The statistics give the total usage, by all nodes, for the requested objects. If a local object is specified, its local usage is returned.

`vxstat` enables the caller to optionally specify a subset of nodes:

```
vxstat -g diskgroup -n node[,node...]
```

where *node* is an integer. If a comma-separated list of nodes is supplied, `vxstat` displays the sum of the statistics for the nodes in the list.

In the following example, `vxstat` is instructed to obtain statistics for Node 2, volume `vol1`:

```
vxstat -g rootdg -n 2 vol1
```

This can produce output similar to the following:

```
OPERATIONS           BLOCKS         AVG TIME(ms)
TYP  NAME           READ     WRITE      READ     WRITE    READ  WRITE
vol  vol1           2421         0    600000         0    99.0    0.0
```

vxstat can also obtain and display statistics for the entire cluster, as follows:

```
vxstat -b
```

The statistics for all nodes are added together. For example, if Node 1 did 100 I/Os and Node 2 did 200 I/Os, vxstat -b would return 300.

2.3 Error Messages

This section presents error messages that can occur with CVM. Each message is accompanied by an explanation and a suggested user action.

Note - Some of these messages can appear on the console; others are returned by vxclust.

```
error in cluster processing
```

Clarification

This can be due to an operation inconsistent with the current state of the cluster (such as an attempt to import or deport a shared disk group from the slave). It can also be caused by an unexpected sequence of commands from vxclust.

Action

Make sure that the operation can be performed in the current environment.

```
cannot find disk on slave node
```

Clarification

A slave node cannot find a shared disk. This is accompanied by the syslog message:

```
vxvm:vxconfigd cannot find disk disk
```

Action

Make sure that the same set of shared disks is online on both nodes.

Examine the disks on both the master and the slave with the command `vxdisk list` and make sure that the same set of disks with the `shared` flag is visible on both nodes. If not, check connections to the disks.

```
disk in use by another cluster
```

Clarification

An attempt was made to import a disk group whose disks are stamped with the ID of another cluster.

Action

If the disk group is not imported by another cluster, retry the import using the `-C` (clear import) flag.

```
vxclust not there
```

Clarification

An error during an attempt to join the cluster caused `vxclust` to fail. This can be caused by the failure of another node during a join or by the failure of `vxclust`.

Action

Retry the join. An error message on the other node can clarify the problem.

```
unable to add portal for cluster
```

Clarification

`vxconfigd` was not able to create a portal for communication with the `vxconfigd` on the other node. This can happen in a degraded system that is experiencing shortages of system resources (such as memory or file descriptors).

Action

If the system does not appear to be degraded, stop and restart `vxconfigd` and try again.

```
vol recovery in progress
```

Clarification

A node that crashed attempted to rejoin the cluster before its DRL map was merged into the recovery map.

Action

Retry the join again later (when the merge operation has completed).

```
cannot assign minor #
```

Clarification

The slave attempted to join, but an existing volume on the slave has the same minor number as a shared volume on the master.

This message should be accompanied by the following console message:

```
WARNING vxvm:vxconfigd minor number ### disk group group in use
```

Action

Before retrying the join, use `vxdg reminor` *diskgroup* ### (see the `vxdg(1M)` manual page) to choose a new minor number range either for the disk group on the master or for the conflicting disk group on the slave. If there are open volumes in the disk group, the reminor operation will not take effect until the disk group is deported and updated either explicitly or through system restart.

```
master sent no data
```

Clarification

During the slave join protocol, a message without data was received. This message is only likely to be seen in the case of a programming error.

Action

Contact Customer Support for more information.

```
join in progress
```

Clarification

An attempt was made to import or deport a shared disk group during a cluster reconfiguration.

Action

Retry later.

```
join not allowed now
```

Clarification

A slave attempted to join the cluster when the master was not ready. The slave will retry automatically. If the retry succeeds, the following message should appear:

```
vxclust: slave join complete
```

Action

No action is necessary if the join eventually completes. Otherwise, investigate the cluster monitor on the master.

```
Disk reserved by other host
```

Clarification

An attempt to put a disk online whose controller has been reserved by another host will fail with this error.

Action

No action is necessary. The cluster manager will free the disk and the volume manager will put the disk online when the node joins the cluster.

```
vxvm:vxconfigd: group group exists
```

Clarification

The slave tried to join the cluster, but there is already a shared disk group in the cluster with the same name as one of its private disk groups.

Action

Use the `vxdg newname` operation to rename either the shared disk group on the master or the private disk group on the slave.

```
WARNING: vxvm:vxio: Plex plex detached from volume volume
NOTICE: vol_kmsg_send_wait_callback: got error 22
NOTICE: commit: NOTE: Reason found for abort: code=6
```

Clarification

These messages can appear during a plex detach operation on a slave.

Action

These messages provide information and require no user action.

```
WARNING: vxvm:vxio: read error on Plex plex of shared volume volume
offset 10 \ length 1
WARNING: vxvm:vxio: Plex plex detached from volume volume
NOTICE: commit: NOTE: Reason found for abort: code=2
NOTICE: ktcvm_check: sent to slave node: node=1 mid=196
```

Clarification

These messages can appear during a plex detach operation on the master.

Action

These messages provide information and require no user action.

2.4 Recovery

The basic recovery actions required for CVM are very similar to the recovery action required for SSVM under identical circumstances. However, when running CVM in clustered mode, additional steps are required and certain restrictions apply. The most significant difference is that recovery of shared disk groups must be initiated from the master node.

There are multiple ways to monitor the status of volume manager objects. The most frequently used mechanisms include `vxprint` command, `VxVa` (GUI) and `vxnotify` command. Most error conditions generate console messages or send messages to the messages file. CVM automatically recovers from some of events (for example, failure of one of the cluster nodes), through the Sun Cluster framework, other events need system administrator intervention. Recovery is done by using one or more of the following utilities: `vxva`, `vxdiskadm`, `vxreattach, vxrecover,` and/or `vxvol`. Some of these utilities internally call other utilities like `vxdg`, `vxdisk`, `vxplex,` and so on. To understand what is needed to recover from a particular situation you must have a solid understanding of volume manager utilities. For volume recovery and/or disk replacement procedures, refer to the *Sun StorEdge Volume Manager 2.6 User's Guide* and the *Sun StorEdge Volume Manager 2.6 System Administrator's Guide.*

The following section describes the process of recovering from some of the most commonly encountered situations.

2.4.1 Disk/Controller/Storage Device Failure

Failure of a disk, controller, or other storage device may make one or more devices inaccessible from one or more nodes. If a device was being accessed at the time of failure, that device is detached from the disk group. The data layout of a mirrored device should be such that no single failure can make both and/or all mirrors unavailable.

The first step of recovery is to make the failed device(s) accessible again, which includes:

- Replacing failed hardware components (if any)
- Executing storage device specific recovery/startup actions (for example using the Recovery Guru on Sun StorEdge A3000 or the `luxadm` on Sun StorEdge A5000)
- Updating the Solaris device tree (`drvconfig/boot -r`)

  For the exact sequence of steps to perform, refer to the storage-specific administration manual.

  The volume manager needs to recognize when a device is accessible, Usually, this is achieved by running `vxdctl enable`, after which CVM can perform the recovery action involving the device. Devices can be reattached using `vxreattach`, `vxdiskadm` (option 5), or the `vxva` GUI. All of these utilities attach the disk using `vxdg -k adddisk`. Once the disk has been attached, the volumes must be recovered using `vxrecover`. The exact operations required to recover depend on the kstate (kernel state) and/or the state of `dm/volume/plex`. For an explanation of the state values, refer to the *Sun StorEdge Volume Manager 2.6 System Administrator's Guide.* A brief discussion on recovering from various states follows.

If you notice devices in the NODEVICE state, you must reattach them using `vxreattach/vxdiskadm/vxva`. `vxreattach` is convenient to use, as it tries to figure out disk media and device access names. However, if a disk was replaced, you must attach it using `vxdg/vxdiskadm/vxva`. When using `vxva/vxdiskadm` you must specify which disk to use for the disk media. Disks that are in the REMOVED state must be attached by using the vxva/vxdisk

Note - If the replacement disk is not initialized, you must first initialize it.

A volume enters kstate ENABLED when it is started, and becomes DISABLED when it is stopped (or as a result of critical errors that render it unusable). If one or more volumes are not in kstate ENABLED, they can be started by using `vxvol/vxrecover/vxva`. A volume may not start if no plexes are in CLEAN or ACTIVE state, in which case you can use `vxmend` to change the state of the selected plex to CLEAN/ACTIVE before the volume can be started.

A volume may enter the NEEDSYNC state if one or more nodes leave the cluster abruptly. In this case `vxrecover` is started by the cluster framework to perform the necessary synchronization. When a volume is being synchronized, it will be in the SYNC state, and it will move to the ACTIVE state once complete. If a process doing recovery is killed, the volume may not transition from SYNC to the ACTIVE state. In this case, it must be recovered using `vxvol -f resync`.

Plexes that are associated with a volume but are detached, are DISABLED (kstate). You can recover these plexes using `vxrecover`, which in turn calls `vxplex att`. The following procedures should enable you to recover from most common failures.

1. **Rectify the fault condition (hardware and/or software) and make sure the devices are accessible again.**

2. **Run** `vxdctl enable` **on all nodes of clusters.**

3. **Run** `vxreattach` **on the master node.**

4. **Run** `vxreattach` **on the other nodes that have non-shared disk groups.**

5. **Verify (by running** `vxprint`**) that the devices have been reattached. (Under certain circumstances,** `vxreattach` **may not reattach a disk removed and/or**

replaced disks. These disks must be manually reattached using `vxdg/vxdiskadm/vxva`**.**

6. **Run** `vxrecover -sb` **on the master node.**

7. **Run** `vxrecover -g <dg> -sb` **on another node with a non-shared disk group.**

2.4.2 Recovery Examples

The following examples show some typical recovery situations. You start the recovery process by checking the operating mode of the cluster nodes; recovery must be performed on the master node (for a non-shared disk group, recovery must be performed on the node where the disk group was imported).

```
Root@capri:/# vxdctl -c mode
mode: enabled: cluster active - SLAVE
Root@palermo:/# vxdctl -c mode
mode: enabled: cluster active - MASTER
```

To check the available disk groups on both nodes, you can use `vxdg list`:

```
Root@capri:/# vxdg list
NAME   STATE          ID
rootdg enabled        885258939.1025.capri
test   enabled,shared 885331519.1233.palermo
Root@palermo:/# vxdg list
NAME   STATE          ID
rootdg enabled        885258917.1025.palermo
test   enabled,shared 885331519.1233.palermo
```

In this case there is one non-shared (*rootdg*) and one shared (*test*) disk group. The disk group ID of *rootdg* differs between the two hosts, even though the name is the same. Notice the state of the volume manager objects. For each object KSTATE and STATE, you can consider the object state and decide whether or not recovery is warranted.

```
vxprint -g test
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg test         test         -        -        -        -        -       -

dm disk1        c4t0d6s2     -        8379057  -        -        -       -
dm disk2        c5t0d6s2     -        8379057  -        -        -       -

v  test1        fsgen        ENABLED  131072   -        ACTIVE   -       -
pl test1-01     test1        ENABLED  132867   -        ACTIVE   -       -
sd c4t0d6s2-01  test1-01     ENABLED  132867   0        -        -       -
pl test1-02     test1        ENABLED  132867   -        ACTIVE   -       -
```

(continued)

```
sd c5t0d6s2-01  test1-02     ENABLED  132867   0        -        -       -

v  test2        fsgen        ENABLED  131072   -        ACTIVE   -       -
pl test2-01     test2        ENABLED  132867   -        ACTIVE   -       -
sd c4t0d6s2-02  test2-01     ENABLED  132867   0        -        -       -
pl test2-02     test2        ENABLED  132867   -        ACTIVE   -       -
sd c5t0d6s2-02  test2-02     ENABLED  132867   0        -        -       -
```

If device `c4t0d6s2` or all devices under controller c4 become unavailable, the device is detached from the disk group. The following example shows how the `vxprint` output looks after fault injection.

```
vxprint -g test
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg test         test         -        -        -        -        -       -

dm disk1        -            -        -        -        NODEVICE -       -
dm disk2        c5t0d6s2     -        8379057  -        -        -       -

v  test1        fsgen        ENABLED  131072   -        ACTIVE   -       -
pl test1-01     test1        DISABLED 132867   -        NODEVICE -       -
sd c4t0d6s2-01  test1-01     DISABLED 132867   0        NODEVICE -       -
pl test1-02     test1        ENABLED  132867   -        ACTIVE   -       -
sd c5t0d6s2-01  test1-02     ENABLED  132867   0        -        -       -

v  test2        fsgen        ENABLED  131072   -        ACTIVE   -       -
pl test2-01     test2        DISABLED 132867   -        NODEVICE -       -
sd c4t0d6s2-02  test2-01     DISABLED 132867   0        NODEVICE -       -
pl test2-02     test2        ENABLED  132867   -        ACTIVE   -       -
sd c5t0d6s2-02  test2-02     ENABLED  132867   0        -        -       -
```

Notice that the state of the DM entry, disk1, and the subdisk and plex using this disk is NODEVICE. In this case, the device should be reattached when it becomes accessible again. The `vxdisk` list output shows the state of the disk. If the device state has changed, you should run `vxdctl enable` before you run `vxdisk list`.

```
vxdctl enable vxdisk list | grep c[45]t0d6s2
c4t0d6s2     sliced    -            -            error  shared
c5t0d6s2     sliced    disk2        test         online shared
-            -         disk1        test        failed was:c4t0d6s2
```

Notice that `c5t0d6s2` is online, but `c4t0d6s2` is in an error state. If the device was accessible to some nodes but not others, the `vxdisk` list output might differ between nodes (nodes that can still access the device will show it online). Now we can rectify the fault condition (in this case *palermo* lost connectivity to one of the SSAs;

connection was restored later). At this point, running `vxreattach` is enough to reattach the devices.

Next, you can run `vxdctl enable` and verify that the devices are now accessible (device state is online). The following examples show the use of the `vxdisk` and `vxdg` commands.

```
vxdctl enable
vxdisk -a online
vxdisk list | grep c[45]t0d6s2
c4t0d6s2     sliced    -            -            error  shared
c5t0d6s2     sliced    disk2        test         online shared
-            -         disk1        test         failed was:c4t0d6s2
```

The preceding listing shows that `c4t0d6s2`, which is no longer associated with any disk group, was associated with disk group *test* as DM *disk1*. You can reattach it with the command `vxdg -g test -k adddisk disk1=c4t0d6s2`, after you verify that *disk1* is still disassociated, and `c4t0d6s2` is the right disk (that is, it has not been swapped).

```
vxprint -d -g test -F ‘‘%name %nodarec %diskid’’
disk1 on 882484157.1163.palermo
disk2 off 884294145.1517.palermo
```

The preceding listing shows the DM name to disk ID association. Since the `nodarec` attribute of `disk1` is on, it is still disassociated. Disk `882484157.1163.palermo` used to be associated with it. If you did not physically replace or move the disk, this disk ID should correspond to `c4t0d6s2`. If it was replaced by a new initialized disk, you may not find a matching disk ID. To verify the disk ID, you can run the command `vxdisk -s list`.

```
vxdisk -s list c4t0d6s2 c5t0d6s2
Disk:   c4t0d6s2
type:   sliced
flags:  online ready private autoconfig shared autoimport
diskid: 882484157.1163.palermo
dgname: test
dgid:   885331519.1233.palermo
clusterid: italia
Disk:   c5t0d6s2
type:   sliced
flags:  online ready private autoconfig shared autoimport imported
diskid: 884294145.1517.palermo
dgname: test
dgid:   885331519.1233.palermo
clusterid: italia
```

The preceding listing shows that disk `c4t0d6s2` ID is `882484157.1163.palermo`. Verifying the association this way is rather tedious. Fortunately, `vxreattach` (with the `-c` option) can show you the disk group and DM with which a disk should be reattached:

```
vxreattach -c  c4t0d6s2
test disk1
```

You can now associate the disk using the command `vxdg -g test -k adddisk disk1=c4t0d6s2`. Under most circumstances, running `vxreattach` takes care of all the preceding steps (running `vxdctl enable` and reattaching devices with respective disk groups). However, if a disk was removed administratively using the `vxdiskadm` command, or was physically replaced, it must be replaced using the `vxdg` command (option 5) or the `VxVM` GUI.

```
vxreattach -bv
! vxdg -g 885331519.1233.palermo -k adddisk disk1=c4t0d6s2
```

You can verify state change of the DM and plex using the `vxprint` command.

```
vxprint -g test
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg test         test         -        -        -        -        -       -

dm disk1        c4t0d6s2     -        8379057  -        -        -       -
dm disk2        c5t0d6s2     -        8379057  -        -        -       -

v  test1        fsgen        ENABLED  131072   -        ACTIVE   -       -
pl test1-01     test1        DISABLED 132867   -        IOFAIL   -       -
sd c4t0d6s2-01  test1-01     ENABLED  132867   0        -        -       -
pl test1-02     test1        ENABLED  132867   -        ACTIVE   -       -
sd c5t0d6s2-01  test1-02     ENABLED  132867   0        -        -       -

v  test2        fsgen        ENABLED  131072   -        ACTIVE   -       -
pl test2-01     test2        DISABLED 132867   -        RECOVER  -       -
sd c4t0d6s2-02  test2-01     ENABLED  132867   0        -        -       -
pl test2-02     test2        ENABLED  132867   -        ACTIVE   -       -
sd c5t0d6s2-02  test2-02     ENABLED  132867   0        -        -       -
```

You can recover the volume and plex now by specifying the `-rb` option of `vxreattach` to start `vxrecover`.

```
vxrecover -g test -vb
job 026404 dg test volume test1: reattach plex test1-01
ps -ef | grep plex
    root 26404 26403  1 13:58:04 ?        0:01 /usr/lib/vxvm/type/fsgen/
vxplex -U fsgen -g 885331519.1233.palermo -- att test1
    root 26406   916  0 13:58:10 console  0:00 grep plex
```

Running in the background, `vxrecover` started `vxplex` to attach the plex to the volume (note the STALE state and ATT in `tutil0`).

```
vxprint -g test
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg test         test         -        -        -        -        -       -
dm disk1        c4t0d6s2     -        8379057  -        -        -       -
dm disk2        c5t0d6s2     -        8379057  -        -        -       -
v  test1        fsgen        ENABLED  131072   -        ACTIVE   ATT1    -
pl test1-01     test1        ENABLED  132867   -        STALE    ATT     -
sd c4t0d6s2-01  test1-01     ENABLED  132867   0        -        -       -
pl test1-02     test1        ENABLED  132867   -        ACTIVE   -       -
sd c5t0d6s2-01  test1-02     ENABLED  132867   0        -        -       -
v  test2        fsgen        ENABLED  131072   -        ACTIVE   -       -
pl test2-01     test2        DISABLED 132867   -        RECOVER  -       -
sd c4t0d6s2-02  test2-01     ENABLED  132867   0        -        -       -
pl test2-02     test2        ENABLED  132867   -        ACTIVE   -       -
sd c5t0d6s2-02  test2-02     ENABLED  132867   0        -        -       -
Root@palermo:/ # job 026404 done status=0
job 026408 dg test volume test2: reattach plex test2-01
job 026408 done status=0
```

After the volumes have been recovered, check the state of the devices again (KSTATE should be ENABLED and STATE should be ACTIVE):

```
vxprint -g test
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg test         test         -        -        -        -        -       -
dm disk1        c4t0d6s2     -        8379057  -        -        -       -
dm disk2        c5t0d6s2     -        8379057  -        -        -       -
v  test1        fsgen        ENABLED  131072   -        ACTIVE   -       -
pl test1-01     test1        ENABLED  132867   -        ACTIVE   -       -
sd c4t0d6s2-01  test1-01     ENABLED  132867   0        -        -       -
pl test1-02     test1        ENABLED  132867   -        ACTIVE   -       -
sd c5t0d6s2-01  test1-02     ENABLED  132867   0        -        -       -
v  test2        fsgen        ENABLED  131072   -        ACTIVE   -       -
pl test2-01     test2        ENABLED  132867   -        ACTIVE   -       -
sd c4t0d6s2-02  test2-01     ENABLED  132867   0        -        -       -
pl test2-02     test2        ENABLED  132867   -        ACTIVE   -       -
sd c5t0d6s2-02  test2-02     ENABLED  132867   0        -        -       -
```

Now the recovery is complete. If the fault had occurred on the slave node rather than the master node, the behavior might vary slightly. Following fault injection the `vxprint` output is similar to the following listing:

```
vxprint -g test
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg test         test         -        -        -        -        -       -

dm disk1        c4t0d6s2     -        8379057  -        -        -       -
dm disk2        c5t0d6s2     -        8379057  -        -        -       -

v  test1        fsgen        ENABLED  131072   -        ACTIVE   -       -
pl test1-01     test1        DETACHED 132867   -        IOFAIL   -       -
sd c4t0d6s2-01  test1-01     ENABLED  132867   0        -        -       -
pl test1-02     test1        ENABLED  132867   -        ACTIVE   -       -
sd c5t0d6s2-01  test1-02     ENABLED  132867   0        -        -       -

v  test2        fsgen        ENABLED  131072   -        ACTIVE   -       -
pl test2-01     test2        ENABLED  132867   -        ACTIVE   -       -
sd c4t0d6s2-02  test2-01     ENABLED  132867   0        -        -       -
pl test2-02     test2        ENABLED  132867   -        ACTIVE   -       -
sd c5t0d6s2-02  test2-02     ENABLED  132867   0        -        -       -
```

Since the devices are not detached, running vxrecover on the master after the slave can access the disk again will be enough. However, if the disk is removed administratively, it must be added using vxdg/vxdiskadm/vxva (vxreattach does not work) and then recovered by using vxrecover.

```
vxdg -k rmdisk disk1
vxprint -g test
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg test         test         -        -        -        -        -       -
dm disk1        -            -        -        -        REMOVED  -       -
dm disk2        c5t0d6s2     -        8379057  -        -        -       -
v  test1        fsgen        ENABLED  131072   -        ACTIVE   -       -
pl test1-01     test1        DISABLED 132867   -        REMOVED  -       -
sd c4t0d6s2-01  test1-01     DISABLED 132867   0        REMOVED  -       -
pl test1-02     test1        ENABLED  132867   -        ACTIVE   -       -
sd c5t0d6s2-01  test1-02     ENABLED  132867   0        -        -       -
v  test2        fsgen        ENABLED  131072   -        ACTIVE   -       -
pl test2-01     test2        DISABLED 132867   -        REMOVED  -       -
sd c4t0d6s2-02  test2-01     DISABLED 132867   0        REMOVED  -       -
pl test2-02     test2        ENABLED  132867   -        ACTIVE   -       -
sd c5t0d6s2-02  test2-02     ENABLED  132867   0        -        -       -
```

Note that vxreattach reports any disk that it can reattach. However, you can reattach the disk manually as follows:

```
vxdg -g test -k adddisk disk1=c4t0d6s2
vxprint -g test
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg test         test         -        -        -        -        -       -

dm disk1        c4t0d6s2     -        8379057  -        -        -       -
dm disk2        c5t0d6s2     -        8379057  -        -        -       -

v  test1        fsgen        ENABLED  131072   -        ACTIVE   -       -
```

```
pl test1-01     test1        DISABLED 132867   -        RECOVER  -       -
sd c4t0d6s2-01  test1-01     ENABLED  132867   0        -        -       -
pl test1-02     test1        ENABLED  132867   -        ACTIVE   -       -
sd c5t0d6s2-01  test1-02     ENABLED  132867   0        -        -       -
v  test2        fsgen        ENABLED  131072   -        ACTIVE   -       -
pl test2-01     test2        DISABLED 132867   -        RECOVER  -       -
sd c4t0d6s2-02  test2-01     ENABLED  132867   0        -        -       -
pl test2-02     test2        ENABLED  132867   -        ACTIVE   -       -
sd c5t0d6s2-02  test2-02     ENABLED  132867   0        -        -       -
vxrecover -v -g test
job 026416 dg test volume test1: reattach plex test1-01
waiting...
job 026416 done status=0
job 026417 dg test volume test2: reattach plex test2-01
waiting...
job 026417 done status=0
```

The following example shows how you can reattach and recover:

```
# ps -ef | grep vx
    root 21935     1  1 20:10:31 ?        5:36 vxconfigd
    root 29295     1  0 14:29:11 ?        0:00 /usr/sbin/vxrecover -c -v -s
    root 29349     1  0 14:29:13 ?        0:00 /usr/sbin/vxrecover -c -v -s
    root 29399 29295  0 14:29:14 ?        0:00 /usr/lib/vxvm/type/fsgen/
vxvol -U fsgen -g 885331519.1233.palermo -- resync tes
    root 29507 29399  0 14:29:16 ?        0:00 /usr/lib/vxvm/type/fsgen/
vxvol -U fsgen -g 885331519.1233.palermo -- resync tes
    root 29508 29349  0 14:29:17 ?        0:00 /usr/lib/vxvm/type/fsgen/
vxvol -U fsgen -g 885331519.1233.palermo -- resync tes
    root 29509 29508  0 14:29:17 ?        0:00 /usr/lib/vxvm/type/fsgen/
vxvol -U fsgen -g 885331519.1233.palermo -- resync tes
    root 29511   916  0 14:29:21 console  0:00 grep vx
vxprint -g test
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg test         test         -        -        -        -        -       -
dm disk1        c4t0d6s2     -        8379057  -        -        -       -
dm disk2        c5t0d6s2     -        8379057  -        -        -       -

v  test1        fsgen        ENABLED  131072   -        SYNC     -       -
pl test1-01     test1        ENABLED  132867   -        ACTIVE   -       -
sd c4t0d6s2-01  test1-01     ENABLED  132867   0        -        -       -
pl test1-02     test1        ENABLED  132867   -        ACTIVE   -       -
sd c5t0d6s2-01  test1-02     ENABLED  132867   0        -        -       -

v  test2        fsgen        ENABLED  131072   -        SYNC     -       -
pl test2-01     test2        ENABLED  132867   -        ACTIVE   -       -
sd c4t0d6s2-02  test2-01     ENABLED  132867   0        -        -       -
pl test2-02     test2        ENABLED  132867   -        ACTIVE   -       -
sd c5t0d6s2-02  test2-02     ENABLED  132867   0        -        -       -
```

Notice that the state of the volumes is now SYNC. Their state will be ACTIVE after `vxplex` completes.

CHAPTER **3**

Tutorial

Here are step-by-step examples of the initialization of the disks in two or more SPARCstorage Array units. By following these examples you can create shared volumes for running Oracle7 or Oracle8 Parallel Server (OPS) software on a Sun Cluster system.

3.1 Assumptions

For this tutorial, make the following assumptions. The actual conditions for your system may vary.

1. The system configuration consists of:
   - Two Enterprise server systems
2. Two SPARCstorage Array units, each containing 30 1-Gbyte disk drives
3. You want to set up areas for the database in the following sizes. Each area is mirrored, so there are two of each type.

TABLE 3–1 Example Database Specifications

Area	**Disks**
Account table	2
History table	2
Administration and system areas	2

TABLE 3–1 Example Database Specifications *(continued)*

Area	Disks
log_node1	2
log_node2	2

1. The root disk group, `rootdg`, has already been created during the system installation. Therefore, CVM recognizes `rootdg`.
2. None of the disks in the SPARCstorage Arrays is currently under CVM control. Therefore, CVM does not currently recognize these disks.
3. The 60 SPARCstorage Array disks are arranged by controller, target, and disk numbers (`c#`, `t#`, and `d#`, respectively) as shown in Figure 3–1.

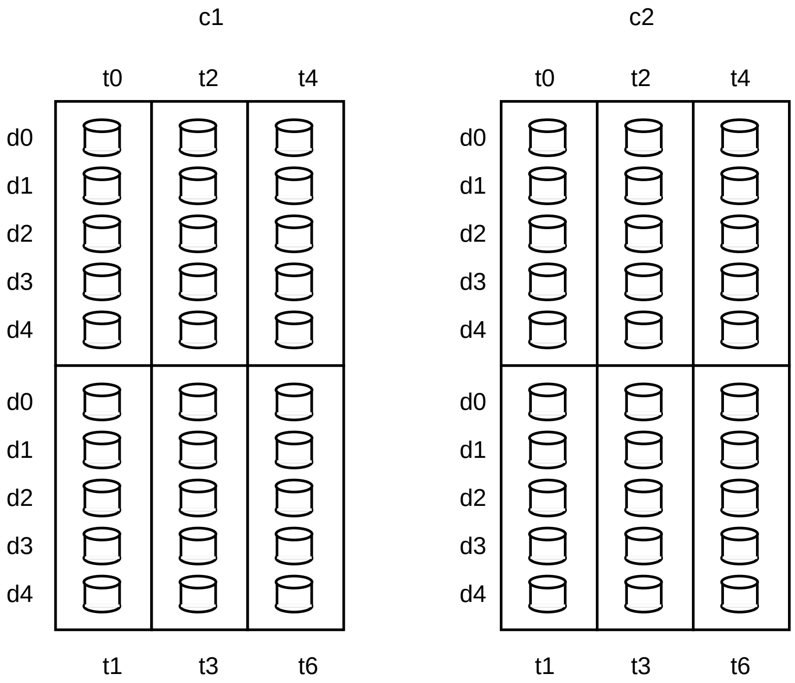

Figure 3–1 Example Disks

4. The operating environment recognizes the 60 disks by both the disk (*dsk*) and raw disk (*rdsk*) names. Table 3–2 lists examples of the disk and raw disk names used in this tutorial. Note that these names include the slice number (where `s2` signifies the entire disk), but the CVM names do not. Both Node 0 and Node 1 use the same name for each disk, because the two array units are connected to the same physical locations on both systems.

TABLE 3–2 Disk Names

	Node 0	Node 0	Node 1	Node 1
Disk #	**Disk name**	**Raw disk name**	**Disk name**	**Raw disk name**
1	/dev/dsk/c1t0d0s2	/dev/rdsk/c1t0d0s2	/dev/dsk/c1t0d0s2	/dev/rdsk/c1t0d0s2
2	/dev/dsk/c1t0d1s2	/dev/rdsk/c1t0d1s2	/dev/dsk/c1t0d1s2	/dev/rdsk/c1t0d1s2
3	/dev/dsk/c1t0d2s2	/dev/rdsk/c1t0d2s2	/dev/dsk/c1t0d2s2	/dev/rdsk/c1t0d2s2
	...	...	...	...
60	/dev/dsk/c2t5d4s2	/dev/rdsk/c2t5d4s2	/dev/dsk/c2t5d4s2	/dev/rdsk/c2t5d47s2

3.2 Initializing Disks Through CVM

Exercise 1: Initialize all of the disks that are to be managed by CVM.

You may use either the `vxva` tool (the graphical user interface or GUI) or the command line interface (CLI).

For the CLI, the command takes the general form:

```
# /etc/vx/bin/vxdisksetup -i devname
```

The term *devname* consists of the controller, target, and disk numbers. For example, *devname* is `c1t0d0` for the first disk.

Repeat the command for each disk that you would like to place under CVM control.

3.3 Starting the Sun Cluster Software

Exercise 2: Bring up the Sun Cluster software.

Note - Run the Sun Cluster software on one node only.

Use the command:

```
# scadmin startcluster nodename cluster_name
```

If you do not specify a cluster name, the default name will be used. After you execute this command, there should be one node (the master node) in the cluster.

3.4 Creating Shared Disk Groups

Exercise 3: Create a shared disk group.

The command syntax is:

```
# vxdg -s init group_name device_name(s)
```

You can specify multiple device names in a single command. For example, you can add the two disks (two mirrored sets of one) in the group `acct` at one time:

```
# vxdg -s init acct c1t0d0 c2t0d0
```

The disks listed here were selected for convenience, because they occupy the same relative locations in the two SPARCstorage Array units.

Note - If you select some other combination of locations, consider how this may complicate disk organization and maintenance.

You can check the results:

```
# vxdisk list
```

The `acct` disk group is a shared disk group. The term shared means that after another node joins the cluster, it will automatically import the `acct` disk group. The volumes can be accessed from both nodes.

The second set of disks on Controller c2 will be used as the mirror disks.

As an alternative to listing all devices in one command, you can start with a single device in a shareable disk group, then add more disks to that disk group. For example, to initialize a single device in a shareable disk group named `acct`:

```
# vxdg -s init acct c1t0d0
```

To add a second disk to the `acct` disk group:

```
# vxdg -g acct adddisk c2t0d0
```

Exercise 4: Continue by creating disk groups named `history`, `admin_system`, `log_node1`, and `log_node2`. Table 3–3 lists the *total* number of disks that you might use for each group. For simplicity in your exercises, use full disks (not partial disks).

TABLE 3–3 Disk Totals

Group Name	First Group	Mirrored Group	Total
acct	1 disk	1 disk	2 disks
history	1 disk	1 disk	2 disks
admin_system	1 disk	1 disk	2 disks
log_node1	1 disk	1 disk	2 disks
log_node2	1 disk	1 disk	2 disks

To list the disk groups on your system, use `vxdg list`.

```
# vxdg list
rootdg  enabled
acct  enabled  shared  nnnnnnnnn.nnnn.node
admin_system  enabled  shared  nnnnnnnnn.nnnn.node
history enabled  shared  nnnnnnnnn.nnnn.node
log_node1 enabled  shared  nnnnnnnnn.nnnn.node
log_node2 enabled shared  nnnnnnnnn.nnnn.node
```

When you have created the disk groups, display the CVM configuration records.

```
# vxprint
```

This command lists all of the volume records on the system.

To examine a specific group, use the `-g` option.

```
# vxprint -g group_name
```

3.5 Creating Volumes

3.5.1 Creating a Volume

Exercise 5: Create volumes in each disk group.

The syntax for the command is:

```
# vxassist -g group_name -U gen make volume_name volume_size disk_name
```

For example, the command for `vol01`, the first volume in the `acct` disk group is:

```
# vxassist -g acct -U gen make vol01 500m c1t0d0
```

3.5.2 Adding a Mirror to a Volume

Exercise 6: Create an associated mirror for each volume.

The syntax for the command is:

```
# vxassist -g group_name mirror volume_name disk_name
```

For example, the command for `vol1` is:

```
# vxassist -g acct mirror vol01 c2t0d0
```

The examples above use a volume size of 500 Mbytes (500m) for simplicity of calculations. The actual size of a CVM device is slightly less than the full disk drive size. CVM reserves a small amount of space for private use, called the private region.

Exercise 7: Repeat the procedure for each volume in the disk group.

Create one volume in each disk group. When you have finished, you should have these mirrored volumes:

TABLE 3–4 Volume Totals

Group	Volume Name	Total
acct	vol01	1
history	vol01	1
admin_system	vol01	1

TABLE 3–4 Volume Totals *(continued)*

Group	Volume Name	Total
log_node1	vol01	1
log_node2	vol01	1

Note - In the example, the five volumes all use the same name, `vol01`. The use of the same name is allowed if the volumes belong to different groups.

3.5.3 Creating a Log File for an Existing Volume

Exercise 8: Create a log for an existing volume.

The syntax for the command is:

```
# vxassist -g group_name addlog volume_name disk_name
```

For example, the command for `vol1` is:

```
# vxassist -g acct addlog vol01 c2t0d0
```

3.6 Bringing Up the Second Node

Exercise 9: Bring up the Sun Cluster software on the other node(s) by entering:

```
# scadmin startnode
```

Alternately, you can specify the cluster name.

```
# scadmin startnode cluster_name
```

After the Sun Cluster software completes the cluster configuration process, you should be able to see the same volume configuration from any node in the cluster.

To review information about the volumes, use `vxprint`.

CVM allows the use of private disk groups. These are limited to individual systems, and are not discussed in this tutorial.

3.7 Reorganizing Disks and Disk Groups

Note - There are several mirrored configurations that are not recommended for the cluster Sun Cluster environment. For example, mirroring across the same disk or within the same tray is unsafe.

3.7.1 Disk Group Expansion

The following procedure is one method of adding disks to a disk group.

Note - You may find that the CVM software offers many ways to produce the same result. Choose the method with which you feel the most comfortable.

There are now five disk groups in your cluster system, with two disks in each of the disk groups.

To inventory your disk resources, enter:

```
# vxdisk list
```

For example, presume that the command shows there are two free disks, `c1t0d1` and `c2t0d1`.

1. **Initialize the disks.**

```
# /etc/vx/bin/vxdisksetup -i c1t0d1
# /etc/vx/bin/vxdisksetup -i c2t0d1
```

2. **Add disks into an existing disk group:**
   The command syntax is:

```
# vxdg -g disk_group_name adddisk devices ...
```

   For the present example, the command is:

```
# vxdg -g acct adddisk c1t0d1 c2t0d1
```

3. **View the new expanded disk group.**

The command syntax is:

```
# vxprint -g disk_group_name
```

For the present example, the command would be:

```
# vxprint -g acct
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg acct         acct         -        -        -        -        -       -

dm c1t0d0       c1t0d0s2     -        2050272  -        -        -       -
dm c1t0d1       c1t0d1s2     -        2050272  -        -        -       -
dm c2t0d0       c2t0d0s2     -        2050272  -        -        -       -
dm c2t0d1       c2t0d1s2     -        2050272  -        -        -       -

v  vol01        gen          ENABLED  1024000  -        ACTIVE   -       -
pl vol01-01     vol01        ENABLED  1024128  -        ACTIVE   -       -
sd c1t0d0-01    vol01-01     ENABLED  1024128  0        -        -       -
pl vol01-02     vol01        ENABLED  1024128  -        ACTIVE   -       -
sd c2t0d0-01    vol01-02     ENABLED  1024128  0        -        -       -
```

4. **Create mirrored volumes.**

The command syntaxes are:

```
# vxassist make vol_name length disk_name
# vxassist mirror vol_name disk_name
```

For the present example:

```
# vxassist make newvol 100m c1t0d1
# vxassist mirror newvol c2t0d1
```

3.7.2 Moving a VM Disk to a Different Disk Group

To move a disk between disk groups, remove the disk from one disk group and add it to the other.

For example, to move the physical disk c1t0d1 from disk group acct to disk group log_node1:

1. **Determine if the disk is in use:**

```
# vxprint -g acct
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg acct         acct         -        -        -        -        -       -

dm c1t0d0       c1t0d0s2     -        2050272  -        -        -       -
dm c1t0d1       c1t0d1s2     -        2050272  -        -        -       -
dm c2t0d0       c2t0d0s2     -        2050272  -        -        -       -
dm c2t0d1       c2t0d1s2     -        2050272  -        -        -       -

v  newvol       gen          ENABLED  204800   -        ACTIVE   -       -
pl newvol-01    newvol       ENABLED  205632   -        ACTIVE   -       -
sd c1t0d1-01    newvol-01    ENABLED  205632   0        -        -       -
pl newvol-02    newvol       ENABLED  205632   -        ACTIVE   -       -
sd c2t0d1-01    newvol-02    ENABLED  205632   0        -        -       -

v  vol01        gen          ENABLED  1024000  -        ACTIVE   -       -
pl vol01-01     vol01        ENABLED  1024128  -        ACTIVE   -       -
sd c1t0d0-01    vol01-01     ENABLED  1024128  0        -        -       -
pl vol01-02     vol01        ENABLED  1024128  -        ACTIVE   -       -
sd c2t0d0-01    vol01-02     ENABLED  1024128  0        -        -       -
```

2. **Remove the volume to free up the** c1t0d1 **disk:**

```
# vxedit -g acct -fr rm newvol
```

The -f option forces an operation. The -r option makes the operation recursive.

3. **Remove the** c1t0d1 **disk from the** acct **disk group:**

```
# vxdg -g acct rmdisk c1t0d1
```

4. **Add the** c1t0d1 **disk to the** log_node1 **disk group:**

```
# vxdg -g log_node1 adddisk c1t0d1
```

Caution - This procedure does NOT save the configuration or data on the disk.

This is the acct disk group after c1t0d1 is removed.

```
# vxprint -g acct
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg acct         acct         -        -        -        -        -       -

dm c1t0d0       c1t0d0s2     -        2050272  -        -        -       -
dm c2t0d0       c2t0d0s2     -        2050272  -        -        -       -
dm c2t0d1       c2t0d1s2     -        2050272  -        -        -       -

v  vol01        gen          ENABLED  1024000  -        ACTIVE   -       -
pl vol01-01     vol01        ENABLED  1024128  -        ACTIVE   -       -
sd c1t0d0-01    vol01-01     ENABLED  1024128  0        -        -       -
pl vol01-02     vol01        ENABLED  1024128  -        ACTIVE   -       -
sd c2t0d0-01    vol01-02     ENABLED  1024128  0        -        -       -
```

This is the log_node1 disk group after c1t0d1 is added.

```
# vxprint -g log_node1
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg log_node1    log_node1    -        -        -        -        -       -

dm c1t0d1       c1t0d1s2     -        2050272  -        -        -       -
dm c1t3d0       c1t3d0s2     -        2050272  -        -        -       -
dm c2t3d0       c2t3d0s2     -        2050272  -        -        -       -
#
```

To change permissions or ownership of volumes, you must use the vxedit command.

Caution - Do not use chmod or chgrp. The permissions and/or ownership set by chmod or chgrp are automatically reset to root during a reboot.

Here is an example of the permissions and ownership of the volumes vol01 and vol02 in the directory /dev/vx/rdsk before a change.

```
# ls -l
crw------- 1 root  root  nnn,nnnnn  date  time  vol01
crw------- 1 root  root  nnn,nnnnn  date  time  vol02
...
```

This an example for changing the permissions and ownership for vol01.

```
# vxedit -g group_name set mode=755 user=oracle vol01
```

After the edit, note how the permissions and ownership have changed.

```
# ls -l
crwxr-xr-x 1  oracle  root  nnn,nnnnn   date  time  vol01
crw------- 1  root  root  nnn,nnnnn   date  time  vol02
...
```

3.8 Mirroring the System Boot Disk

This section describes the procedure for converting an existing Sun Cluster system that has a defined `rootdg` and that must be converted to full encapsulation.

This procedure has two parts. The first part (Procedure A) uses a single disk slice for temporary CVM data while doing the encapsulation. The second part (Procedure B) uses an entire disk for the temporary CVM data.

3.8.1 Mirroring the System Boot Disk on an Existing Node

Use one of the following procedures on an existing Sun Cluster node to mirror the boot disk:

- Section 3.8.1.1 "Procedure A: Using a Disk Slice" on page 3-13" uses a simple partition for the boot disk group.
- Section 3.8.1.2 "Procedure B: Using an Entire Disk" on page 3-16" uses a full disk for the boot disk group.

Both procedures use these assumptions:

1. The boot disk has at least two free partitions.
2. The beginning or end of the disk has two cylinders of free space available.
3. `rootdg` was created from a simple partition.
4. An extra disk or partition can be used as temporary spare space. This space should be on a disk other than the boot disk you are encapsulating.

3.8.1.1 Procedure A: Using a Disk Slice

Use this procedure to encapsulate and mirror the boot disk if only one partition is available for temporary storage. If an entire disk is available for temporary storage, see Section 3.8.1.2 “Procedure B: Using an Entire Disk” on page 3-16.”

Configuration

- Boot disk (example: boot disk is `c0t0d0`)
- `rootdg` is a simple partition that is part of the boot disk (example: `c0t0d0s5` is a simple disk for `rootdg`)
- One spare partition for temporary usage (example: `c0t3d0s7`)
- Mirror disk (example: `c0t2d0`)

Procedure A

1. **Stop the Sun Cluster software on the current node:**

```
# scadmin stopnode
```

2. **If the simple partition for `rootdg` is on the boot disk, find a separate partition (two cylinders in size) that is not on the boot disk. If the simple partition for `rootdg` is not on the boot disk, proceed directly to Step 10 on page @–14.**

3. **Use the `format` command to reserve and label the new partition:**

```
# format c0t3d0
```

Make `c0t3d0s7` the new partition, two cylinders in size.

4. **Define the new partition to CVM:**

```
# vxdisk -f init c0t3d0s7
```

5. **Add the new partition to `rootdg`:**

```
# vxdg adddisk c0t3d0s7
```

6. **Add the new `rootdg` partition to the `volboot` file:**

```
# vxdctl add disk c0t3d0s7
```

7. **Use the `format` command to free up the old disk partition:**

```
# format c0t0d0
```

This frees up the space for partition `c0t0d0s5`.

8. **Remove the original disk from `rootdg`:**

```
# vxdg rmdisk c0t0d0s5
# vxdisk rm c0t0d0s5
```

9. **Clean up the old partition of `rootdg` in the `volboot` file:**

```
# vxdctl rm disk c0t0d0s5
```

10. **Enter `vxdiskadm` to encapsulate the boot disk (`c0t0d0` in this example):**

```
# vxdiskadm -Select an operation to perform: 2
-Select disk devices to encapsulate:
[,all,list,q,?] c0t0d0
-Continue operation? [y,n,q,?] (default: y) y
-Which disk group [,list,q,?] (default: rootdg) rootdg
-Use a default disk name for the disk? [y,n,q,?] (default: y) n
-Continue with operation? [y,n,q,?] (default: y) y
-Continue with encapsulation? [y,n,q,?] (default: y) y
-Enter disk name for  [,q,?] (default: disk01) disk01
-Encapsulate other disks? [y,n,q,?] (default: n) n
-Select an operation to perform: q
```

Verify each step carefully.

11. **Reboot the system:**

```
# shutdown -g0 -y -i6
```

The system will reboot one more time to complete the process.

Your boot disk is now encapsulated and managed by CVM.

12. **To verify the encapsulation:**

a. **Invoke `vxva` and open up the icon of the `rootdg` disk group.**

   You should see the encapsulated boot disk and the four volumes that have been created from it.

b. **Verify that the `/etc/vfstab` file now refers to device files in the `/dev/vx/dsk` directory rather than the `/dev/dsk` directory.**

13. **To mirror the encapsulated disk, choose a disk (for example, `c0t2d0`) and a media name (for example, `mirrorroot`), and enter:**

```
# /etc/vx/bin/vxdisksetup -i c0t2d0
# /usr/sbin/vxdg adddisk mirrorroot=c0t2d0
```

You should now see the new disk in the `rootdg` disk group of `vxva`.

14. **Mirror the encapsulated boot disk on `mirrorroot`:**

```
# /etc/vx/bin/vxmirror disk01 mirrorroot
```

The `vxmirror` command prints a series of commands corresponding to the mirroring of volumes in the encapsulated boot disk.

15. **After the `vxmirror` command is completed, verify that the process was successful by looking for the mirrored volumes in the `rootdg` disk group of `vxva`.**

It is now possible to boot from the mirror of the boot disk.

16. **Remove the two-cylinder simple disk in the `rootdg` disk group:**

```
# vxdg rmdisk c0t3d0s7
# vxdisk rm c0t3d0s7
# vxdctl rm disk c0t3d0s7
```

17. **Start the Sun Cluster software:**

```
# scadmin startnode
```

18. If you wish to mirror the other system, repeat Procedure A.

19. Un-encapsulate the boot disk.

When you upgrade your system or the volume manager you need to un-encapsulate the boot disk first. Use the script provided under /*CD_path*/`CVM/scripts` called `upgrade_start` to automatically convert the file systems on volumes back to regular disk partitions. Reboot the system to complete the conversion to regular disk partitions.

3.8.1.2 Procedure B: Using an Entire Disk

This procedure contains step-by-step instructions for encapsulating and mirroring the boot disk when you are using an entire disk for temporary storage. If an entire disk is not available, see Section 3.8.1.1 "Procedure A: Using a Disk Slice" on page 3-13."

Configuration

- Boot disk (for example, boot disk is `c0t0d0`)
- `rootdg` is made with a simple partition, which is part of boot disk (example: `c0t0d0s5` is a simple disk for `rootdg`)
- One spare disk for temporary usage (for example, `c1t0d0`)
- Mirror disk (for example, `c0t2d0`)

Procedure B

1. **Stop the Sun Cluster software on the current node:**

```
# scadmin stopnode
```

2. **Initial the spare disk for use with CVM:**

```
# /etc/vx/bin/vxdisksetup -i c1t0d0
```

3. **Add the new disk to** `rootdg`**:**

```
# vxdg adddisk c1t0d0
```

4. **Use the** `format` **command to release the old disk partition, which is a part of boot disk for** `rootdg`**:**

```
# format c0t0d0
```

5. **Remove the original disk from** `rootdg`**:**

```
# vxdg rmdisk c0t0d0s5 # vxdisk rm c0t0d0s5
```

6. **Clean up the old partition of** `rootdg` **in the** `volboot` **file:**

```
# vxdctl rm disk c0t0d0s5
```

7. **Use the** `vxdiskadm` **command to encapsulate the boot disk (for example,** `c0t0d0`**):**

```
# vxdiskadm
-Select an operation to perform: 2
-Select disk devices to encapsulate:
[,all,list,q,?] c0t0d0
-Continue operation? [y,n,q,?] (default: y) y
-Which disk group [,list,q,?] (default: rootdg) rootdg
-Use a default disk name for the disk? [y,n,q,?] (default: y) n
-Continue with operation? [y,n,q,?] (default: y) y
-Continue with encapsulation? [y,n,q,?] (default: y) y
-Enter disk name for [,q,?] (default: disk01) disk01
-Encapsulate other disks? [y,n,q,?] (default: n) n
-Select an operation to perform: q
```

8. **Reboot the system:**

```
# shutdown -g0 -y -i6
```

The system will reboot a second time to complete the process.

The boot disk is now encapsulated and managed by CVM.

9. **To verify the encapsulation:**

   a. **Invoke** `vxva` **and open up the icon of the** `rootdg` **disk group.**

      You should see the encapsulated boot disk and the four volumes that have been created from it.

   b. **Verify that the** `/etc/vfstab` **file now refers to device files in the** `/dev/vx/dsk` **directory rather than the** `/dev/dsk` **directory.**

10. **To mirror the encapsulated disk, choose a disk (for example, `c0t2d0`) and a media name (for example, `mirrorroot`), and enter:**

```
# /etc/vx/bin/vxdisksetup -i c0t2d0
# /usr/sbin/vxdg adddisk mirrorroot=c0t2d0
```

    You should see the new disk in the `rootdg` disk group of `vxva`.

11. **Mirror the encapsulated boot disk on `mirrorroot`:**

```
# /etc/vx/bin/vxmirror disk01 mirrorroot
```

    `vxmirror` displays a series of commands corresponding to the mirroring of volumes in the encapsulated boot disk.

12. **After `vxmirror` is completed, verify that the process was successful by looking for the mirrored volumes in the `rootdg` disk group of `vxva`.**

    It is now possible to boot from the mirror of the boot disk.

13. **Remove the temporary disk from the `rootdg` disk group:**

```
# vxdg rmdisk c1t0d0
# vxdisk rm c1t0d0
```

14. **Start the Sun Cluster software:**

```
# scadmin startnode
```

15. **If you would like to mirror the other system, repeat Procedure B.**

3.8.2 Description of `vxconfigd`

The `vxconfigd` command takes requests from other utilities for volume/disk configuration changes, and communicates those changes to the kernel and modifies configuration information stored on disk. `vxconfigd` is also responsible for initializing CVM when the system is booted.

3.8.3 Error Handling

The vxconfigd command may exit if serious errors occur. By default, vxconfigd issues errors to the console. However, vxconfigd can be configured to log errors to a log file with various parameters for debugging. (See the man page for further information.)

This is an example which enables debugging parameters and logs messages to a file:

```
# vxdctl stop
# vxconfigd -x 1 -x logfile=filename -x mstimestamp > /dev/null 2>&1 &
```

The options are:

- `-x 1` Set the trace output level to 1.
- `-x logfile=`*filename* Log the message to a specified file.
- `-x mstimestamp` Attach a date, time-of-day and millisecond time stamp to all messages written by vxconfigd onto the log file.

3.9 Moving the Quorum Controller

When a quorum controller fails, you must select a new quorum controller.

Caution - Select a new quorum controller before taking a node down. If you fail to do this first, the entire cluster will shut down.

1. **Obtain the current SPARCstorage Array Sun Cluster configuration:**

```
# scconf cluster_name -p
```

   Look for the quorum controller serial numbers. The first controller should be the quorum controller (check the `Quorum Controller:` line in the display).

2. **On both nodes, change the quorum controller by selecting a new quorum controller serial number (validate against the SPARCstorage Array LCD display) and enter:**

```
# scconf cluster_name -q -m host1 host2 ctlr
```

Caution - If you do not make the change on *both* nodes, the database may be destroyed.

3. **Verify that the *ctrl2* SPARCstorage Array is now the new quorum controller:**

```
# scconf cluster_name -p
```

3.10 Replacing a Bad Disk in a SPARCstorage Array Tray

It is possible to replace a SPARCstorage Array disk without halting system operations.

Note - The following procedure is used to replace a failed disk in a SPARCstorage Array 100 Series. When removable storage media (RSM) disks are used, follow the procedure in the *Sun Cluster 2.2 System Administration Guide* and the applicable hardware service manual.

1. **Identify all the volumes and corresponding plexes on the disks in the tray which contains the faulty disk.**

   a. **From the physical device address `c`*N*`t`*N*`d`*N*, obtain the controller number and the target number.**

   For example, if the device address is c3t2d0, the controller number is 3 and the target is 2.

   b. **Identify devices from a `vxdisk list` output:**

   If the target is 0 or 1, identify all devices with physical addresses beginning with c*N*t0 and c*N*t1. If the target is 2 or 3, identify all devices with physical addresses beginning with c*N*t2 and c*N*t3. If the target is 4 or 5, identify all devices with physical addresses beginning with c*N*t4 and c*N*t5.

   For example:

```
# vxdisk -g diskgroup -q list | egrep c3t2\|c3t3 | nawk '{print $3}'
```

Record the volume media name for the faulty disk from the output of the command. The volume media name is the variable *device_media_name* used in Step 8 on page @–22.

c. **Identify all plexes on the above devices by using the appropriate version (csh, ksh, or Bourne shell) of the following command:**

```
PLLIST=‘vxprint -ptq -g diskgroup -e ’(aslist.sd_dm_name in
(‘‘c3t2d0’’,’’c3t3d0’’,’’c3t3d1’’)) && (pl_kstate=ENABLED)’ | nawk ’{print $2}’‘
```

For csh, the syntax is `set PLLIST ....` For ksh, the syntax is `export PLLIST= ....` The Bourne shell requires the command `export PLLIST` after the variable is set.

2. **After you have set the variable, detach all plexes in the tray:**

```
# vxplex det ${PLLIST}
```

An alternate command for detaching each plex in a tray is:

```
# vxplex -g diskgroup -v volume det plex
```

Note - The volumes will still be active because the other mirror is still available.

---

3. **Spin down the disks in the tray:**

```
# ssaadm stop -t tray controller
```

4. **Replace the faulty disk.**

5. **Spin up the drives:**

```
# ssaadm start -t tray controller
```

6. **Initialize the replacement disk:**

```
# vxdisksetup -i devicename
```

7. **Scan the current disk configuration again:**

   Enter the following commands on both nodes in the cluster

```
# vxdctl enable
# vxdisk -a online
```

8. **Add the new disk to the disk group:**

```
# vxdg -g diskgroup -k adddisk device_media_name=device_name
```

9. **Resynchronize the volumes:**

```
# vxrecover -b -o iosize=192K
```

3.11 Reattaching a Disk Drive

Disk drives sometimes become detached due to power failures, cable or controller problems, and so on. Detached disk drives result in plexes becoming detached and thus, unavailable. The remaining plex(es) in a mirrored volume are still available and therefore the volume is still active. It is possible to reattach the disk drives and recover from this condition without halting either node.

The following example configuration consists of five mirrored volumes. SPARCstorage Array c1 has been powered off. The mirrors are broken, but the volume remains active, as the remaining plex is still working.

```
# vxprint -g toi
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg toi          toi          -        -        -        -        -       -

dm c1t5d0       -            -        -        -        NODEVICE -       -
dm c1t5d1       -            -        -        -        NODEVICE -       -
```

(continued)

```
dm c1t5d2       -            -        -        -        NODEVICE -       -
dm c1t5d3       -            -        -        -        NODEVICE -       -
dm c1t5d4       -            -        -        -        NODEVICE -       -
dm c2t5d0       c2t5d0s2     -        2050272  -        -        -       -
dm c2t5d1       c2t5d1s2     -        2050272  -        -        -       -
dm c2t5d2       c2t5d2s2     -        2050272  -        -        -       -
dm c2t5d3       c2t5d3s2     -        2050272  -        -        -       -
dm c2t5d4       c2t5d4s2     -        2050272  -        -        -       -

v  toi-1        gen          ENABLED  61440    -        ACTIVE   -       -
pl toi-1-01     toi-1        DISABLED 65840    -        NODEVICE -       -
sd c1t5d0-01    toi-1-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d1-01    toi-1-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d2-01    toi-1-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d3-01    toi-1-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d4-01    toi-1-01     DISABLED 13104    0        NODEVICE -       -
pl toi-1-02     toi-1        ENABLED  65840    -        ACTIVE   -       -
sd c2t5d0-01    toi-1-02     ENABLED  13104    0        -        -       -
sd c2t5d1-01    toi-1-02     ENABLED  13104    0        -        -       -
sd c2t5d2-01    toi-1-02     ENABLED  13104    0        -        -       -
sd c2t5d3-01    toi-1-02     ENABLED  13104    0        -        -       -
sd c2t5d4-01    toi-1-02     ENABLED  13104    0        -        -       -

v  toi-2        gen          ENABLED  61440    -        ACTIVE   -       -
pl toi-2-01     toi-2        DISABLED 65840    -        NODEVICE -       -
sd c1t5d0-02    toi-2-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d1-02    toi-2-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d2-02    toi-2-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d3-02    toi-2-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d4-02    toi-2-01     DISABLED 13104    0        NODEVICE -       -
pl toi-2-02     toi-2        ENABLED  65840    -        ACTIVE   -       -
sd c2t5d0-02    toi-2-02     ENABLED  13104    0        -        -       -
sd c2t5d1-02    toi-2-02     ENABLED  13104    0        -        -       -
sd c2t5d2-02    toi-2-02     ENABLED  13104    0        -        -       -
sd c2t5d3-02    toi-2-02     ENABLED  13104    0        -        -       -
sd c2t5d4-02    toi-2-02     ENABLED  13104    0        -        -       -

v  toi-3        gen          ENABLED  61440    -        ACTIVE   -       -
pl toi-3-01     toi-3        DISABLED 65840    -        NODEVICE -       -
sd c1t5d0-03    toi-3-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d1-03    toi-3-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d2-03    toi-3-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d3-03    toi-3-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d4-03    toi-3-01     DISABLED 13104    0        NODEVICE -       -
pl toi-3-02     toi-3        ENABLED  65840    -        ACTIVE   -       -
sd c2t5d0-03    toi-3-02     ENABLED  13104    0        -        -       -
sd c2t5d1-03    toi-3-02     ENABLED  13104    0        -        -       -
sd c2t5d2-03    toi-3-02     ENABLED  13104    0        -        -       -
sd c2t5d3-03    toi-3-02     ENABLED  13104    0        -        -       -
sd c2t5d4-03    toi-3-02     ENABLED  13104    0        -        -       -

v  toi-4        gen          ENABLED  61440    -        ACTIVE   -       -
pl toi-4-01     toi-4        DISABLED 65840    -        NODEVICE -       -
sd c1t5d0-04    toi-4-01     DISABLED 13104    0        NODEVICE -       -
```

(continued)

```
sd c1t5d1-04    toi-4-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d2-04    toi-4-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d3-04    toi-4-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d4-04    toi-4-01     DISABLED 13104    0        NODEVICE -       -
pl toi-4-02     toi-4        ENABLED  65840    -        ACTIVE   -       -
sd c2t5d0-04    toi-4-02     ENABLED  13104    0        -        -       -
sd c2t5d1-04    toi-4-02     ENABLED  13104    0        -        -       -
sd c2t5d2-04    toi-4-02     ENABLED  13104    0        -        -       -
sd c2t5d3-04    toi-4-02     ENABLED  13104    0        -        -       -
sd c2t5d4-04    toi-4-02     ENABLED  13104    0        -        -       -

v  toi-5        gen          ENABLED  61440    -        ACTIVE   -       -
pl toi-5-01     toi-5        DISABLED 65840    -        NODEVICE -       -
sd c1t5d0-05    toi-5-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d1-05    toi-5-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d2-05    toi-5-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d3-05    toi-5-01     DISABLED 13104    0        NODEVICE -       -
sd c1t5d4-05    toi-5-01     DISABLED 13104    0        NODEVICE -       -
pl toi-5-02     toi-5        ENABLED  65840    -        ACTIVE   -       -
sd c2t5d0-05    toi-5-02     ENABLED  13104    0        -        -       -
sd c2t5d1-05    toi-5-02     ENABLED  13104    0        -        -       -
sd c2t5d2-05    toi-5-02     ENABLED  13104    0        -        -       -
sd c2t5d3-05    toi-5-02     ENABLED  13104    0        -        -       -
sd c2t5d4-05    toi-5-02     ENABLED  13104    0        -        -       -
# vxdisk list
DEVICE       TYPE      DISK         GROUP        STATUS
c0t0d0s4     simple    c0t0d0s4     rootdg       online
c1t5d0s2     sliced    -            -            online
c1t5d1s2     sliced    -            -            online
c1t5d2s2     sliced    -            -            online
c1t5d3s2     sliced    -            -            online
c1t5d4s2     sliced    -            -            online
c2t5d0s2     sliced    c2t5d0       toi          online shared
c2t5d1s2     sliced    c2t5d1       toi          online shared
c2t5d2s2     sliced    c2t5d2       toi          online shared
c2t5d3s2     sliced    c2t5d3       toi          online shared
c2t5d4s2     sliced    c2t5d4       toi          online shared
-            -         c1t5d0       toi          failed was:c1t5d0s2
-            -         c1t5d1       toi          failed was:c1t5d1s2
-            -         c1t5d2       toi          failed was:c1t5d2s2
-            -         c1t5d3       toi          failed was:c1t5d3s2
-            -         c1t5d4       toi          failed was:c1t5d4s2
```

To reattach a detached disk, complete the following steps.

1. **Fix the condition that resulted in the problem.**

   Be sure that the disks are spun up before proceeding further.

2. **Enter the following commands *on both nodes* in the cluster.**

   In some cases, the drive(s) must be rediscovered by the node(s).

```
# drvconfig
# disks
```

3. **Enter the following commands *on both nodes* in the cluster.**

   CVM must scan the current disk configuration again.

```
# vxdctl enable
# vxdisk -a online
```

4. **From the master node, repeat the following command for each disk that has been disconnected.**

   The physical disk and the CVM access name for that disk must be reconnected.

   ```
   # vxdg -g disk_group_name -k adddisk medianame=accessname
   ```

   Note - The values for `medianame` and `accessname` can be obtained from the end of the `vxdisk list` command output.

   For the example configuration:

```
# vxdg -g toi -k adddisk c1t5d0=c1t5d0s2
# vxdg -g toi -k adddisk c1t5d1=c1t5d1s2
# vxdg -g toi -k adddisk c1t5d2=c1t5d2s2
# vxdg -g toi -k adddisk c1t5d3=c1t5d3s2
# vxdg -g toi -k adddisk c1t5d4=c1t5d4s2
```

5. **From the master node, start volume recovery.**

   ```
   # vxrecover -svc
   ```

   For the example configuration:

```
# vxrecover -svc
job 028125 dg toi volume toi-1: reattach plex toi-1-01
job 028125 done status=0
job 028126 dg toi volume toi-2: reattach plex toi-2-01
job 028126 done status=0
job 028127 dg toi volume toi-3: reattach plex toi-3-01
job 028127 done status=0
job 028129 dg toi volume toi-4: reattach plex toi-4-01
job 028129 done status=0
job 028130 dg toi volume toi-5: reattach plex toi-5-01
job 028130 done status=0
```

6. **(Optional) Enter the** `vxprint -g` **command to see the changes.**

```
# vxprint -g toi
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg toi          toi          -        -        -        -        -       -

dm c1t5d0       c1t5d0s2     -        2050272  -        -        -       -
dm c1t5d1       c1t5d1s2     -        2050272  -        -        -       -
dm c1t5d2       c1t5d2s2     -        2050272  -        -        -       -
dm c1t5d3       c1t5d3s2     -        2050272  -        -        -       -
dm c1t5d4       c1t5d4s2     -        2050272  -        -        -       -
dm c2t5d0       c2t5d0s2     -        2050272  -        -        -       -
dm c2t5d1       c2t5d1s2     -        2050272  -        -        -       -
dm c2t5d2       c2t5d2s2     -        2050272  -        -        -       -
dm c2t5d3       c2t5d3s2     -        2050272  -        -        -       -
dm c2t5d4       c2t5d4s2     -        2050272  -        -        -       -

v  toi-1        gen          ENABLED  61440    -        ACTIVE   -       -
pl toi-1-01     toi-1        ENABLED  65840    -        ACTIVE   -       -
sd c1t5d0-01    toi-1-01     ENABLED  13104    0        -        -       -
sd c1t5d1-01    toi-1-01     ENABLED  13104    0        -        -       -
sd c1t5d2-01    toi-1-01     ENABLED  13104    0        -        -       -
sd c1t5d3-01    toi-1-01     ENABLED  13104    0        -        -       -
sd c1t5d4-01    toi-1-01     ENABLED  13104    0        -        -       -
pl toi-1-02     toi-1        ENABLED  65840    -        ACTIVE   -       -
sd c2t5d0-01    toi-1-02     ENABLED  13104    0        -        -       -
sd c2t5d1-01    toi-1-02     ENABLED  13104    0        -        -       -
sd c2t5d2-01    toi-1-02     ENABLED  13104    0        -        -       -
sd c2t5d3-01    toi-1-02     ENABLED  13104    0        -        -       -
sd c2t5d4-01    toi-1-02     ENABLED  13104    0        -        -       -

v  toi-2        gen          ENABLED  61440    -        ACTIVE   -       -
pl toi-2-01     toi-2        ENABLED  65840    -        ACTIVE   -       -
sd c1t5d0-02    toi-2-01     ENABLED  13104    0        -        -       -
sd c1t5d1-02    toi-2-01     ENABLED  13104    0        -        -       -
sd c1t5d2-02    toi-2-01     ENABLED  13104    0        -        -       -
sd c1t5d3-02    toi-2-01     ENABLED  13104    0        -        -       -
sd c1t5d4-02    toi-2-01     ENABLED  13104    0        -        -       -
pl toi-2-02     toi-2        ENABLED  65840    -        ACTIVE   -       -
```

(continued)

(Continuation)

```
sd c2t5d0-02    toi-2-02     ENABLED  13104    0        -        -       -
sd c2t5d1-02    toi-2-02     ENABLED  13104    0        -        -       -
sd c2t5d2-02    toi-2-02     ENABLED  13104    0        -        -       -
sd c2t5d3-02    toi-2-02     ENABLED  13104    0        -        -       -
sd c2t5d4-02    toi-2-02     ENABLED  13104    0        -        -       -

v  toi-3        gen          ENABLED  61440    -        ACTIVE   -       -
pl toi-3-01     toi-3        ENABLED  65840    -        ACTIVE   -       -
sd c1t5d0-03    toi-3-01     ENABLED  13104    0        -        -       -
sd c1t5d1-03    toi-3-01     ENABLED  13104    0        -        -       -
sd c1t5d2-03    toi-3-01     ENABLED  13104    0        -        -       -
sd c1t5d3-03    toi-3-01     ENABLED  13104    0        -        -       -
sd c1t5d4-03    toi-3-01     ENABLED  13104    0        -        -       -
pl toi-3-02     toi-3        ENABLED  65840    -        ACTIVE   -       -
sd c2t5d0-03    toi-3-02     ENABLED  13104    0        -        -       -
sd c2t5d1-03    toi-3-02     ENABLED  13104    0        -        -       -
sd c2t5d2-03    toi-3-02     ENABLED  13104    0        -        -       -
sd c2t5d3-03    toi-3-02     ENABLED  13104    0        -        -       -
sd c2t5d4-03    toi-3-02     ENABLED  13104    0        -        -       -

v  toi-4        gen          ENABLED  61440    -        ACTIVE   -       -
pl toi-4-01     toi-4        ENABLED  65840    -        ACTIVE   -       -
sd c1t5d0-04    toi-4-01     ENABLED  13104    0        -        -       -
sd c1t5d1-04    toi-4-01     ENABLED  13104    0        -        -       -
sd c1t5d2-04    toi-4-01     ENABLED  13104    0        -        -       -
sd c1t5d3-04    toi-4-01     ENABLED  13104    0        -        -       -
sd c1t5d4-04    toi-4-01     ENABLED  13104    0        -        -       -
pl toi-4-02     toi-4        ENABLED  65840    -        ACTIVE   -       -
sd c2t5d0-04    toi-4-02     ENABLED  13104    0        -        -       -
sd c2t5d1-04    toi-4-02     ENABLED  13104    0        -        -       -
sd c2t5d2-04    toi-4-02     ENABLED  13104    0        -        -       -
sd c2t5d3-04    toi-4-02     ENABLED  13104    0        -        -       -
sd c2t5d4-04    toi-4-02     ENABLED  13104    0        -        -       -

v  toi-5        gen          ENABLED  61440    -        ACTIVE   -       -
pl toi-5-01     toi-5        ENABLED  65840    -        ACTIVE   -       -
sd c1t5d0-05    toi-5-01     ENABLED  13104    0        -        -       -
sd c1t5d1-05    toi-5-01     ENABLED  13104    0        -        -       -
sd c1t5d2-05    toi-5-01     ENABLED  13104    0        -        -       -
sd c1t5d3-05    toi-5-01     ENABLED  13104    0        -        -       -
sd c1t5d4-05    toi-5-01     ENABLED  13104    0        -        -       -
pl toi-5-02     toi-5        ENABLED  65840    -        ACTIVE   -       -
sd c2t5d0-05    toi-5-02     ENABLED  13104    0        -        -       -
sd c2t5d1-05    toi-5-02     ENABLED  13104    0        -        -       -
sd c2t5d2-05    toi-5-02     ENABLED  13104    0        -        -       -
sd c2t5d3-05    toi-5-02     ENABLED  13104    0        -        -       -
sd c2t5d4-05    toi-5-02     ENABLED  13104    0        -        -       -
```

3.12 Changing Permission or Ownership

To run applications such as Oracle Parallel Server, it may be necessary to change read / write permissions and ownership of the volumes. To change the permissions or ownership, use the `vxedit` command. `vxedit` will set the necessary fields in a CVM record.

Index