VERITAS

VERITAS Volume Manager™

Administrator's Reference Guide

Release 3.0.1

Solaris
May, 1999
P/N 100-001125

TRADEMARKS

Contents

Preface . . . . . **xiii**

1. Recovery **1**

Introduction 1

The UNIX Boot Process 2

Booting After Failures 3

Possible Root (/), swap, and usr Configurations 4

Repairing Root (/) or /usr File Systems on Volumes 5

Recovering a Volume Manager Root Disk (and Root Mirror) From Backup Tape. 5

Backing Up and Restoring the Root File System 8

Failures and Recovery Procedures 8

Failures in UNIX Partitioning 9

Failures Accessing the Boot Device 9

Failures Due to Incorrect Entries in /etc/vfstab 10

Damaged / Entry in `/etc/vfstab` 10

Damaged /usr Entry in /etc/vfstab 11

Failures Due to Missing or Damaged /etc/system 11

`/etc/system` Copy Not Available 11

/etc/system Copy Available 12
/etc/system Not Available and /usr is Volume..... 13
Failures Due To Booting From Unusable or Stale Plexes..... 14
Hot-Relocation and Boot Disk Failures 15
Re-Adding and Replacing Boot Disks 16
Re-Adding a Failed Boot Disk 16
Replacing a Failed Boot Disk 18
Reattaching Disks 19
Reinstallation Recovery 19
General Reinstallation Information 20
Reinstallation and Reconfiguration Procedures..... 21
Preparing the System for Reinstallation..... 21
Reinstalling the Operating System 22
Reinstalling the Volume Manager..... 22
Recovering the Volume Manager Configuration..... 22
Configuration Cleanup 24
Rootability Cleanup 25
Plex and Volume States..... 31
Plex States 31
EMPTY Plex State 32
CLEAN Plex State 32
ACTIVE Plex State..... 32
STALE Plex State 33
OFFLINE Plex State..... 33
TEMP Plex State..... 33
TEMPRM Plex State 34
TEMPRMSD Plex State..... 34

IOFAIL Plex State 34
The Plex State Cycle 34
Plex Kernel State 35
Volume States 35
RAID-5 Volume States 36
Volume Kernel State 37
2. RAID-5 Volume Recovery 39
Introduction 39
RAID-5 Volume Layout 39
RAID-5 Plexes 39
RAID-5 Logs 40
Creating RAID-5 Volumes 41
`vxassist` and RAID-5 Volumes 41
`vxmake` and RAID-5 Volumes 42
Initializing RAID-5 Volumes 43
Failures and RAID-5 Volumes 44
System Failures 44
Disk Failures 45
RAID-5 Recovery 46
Parity Recovery 47
Subdisk Recovery 49
Recovering Logs After Failures 49
Miscellaneous RAID-5 Operations 49
Manipulating RAID-5 Logs 50
Manipulating RAID-5 Subdisks 50
RAID-5 Subdisk Moves 51
Starting RAID-5 Volumes 52

Unstartable RAID-5 Volumes 52
Forcibly Starting RAID-5 Volumes 54
Recovery When Starting RAID-5 Volumes 54
Changing RAID-5 Volume Attributes 56
Writing to RAID-5 Arrays 56
Read-Modify-Write 57
Full-Stripe Writes 57
Reconstruct-Writes 60
3. Disks and Disk Groups 63
Introduction 63
Standard Disk Devices 64
Disk Groups 66
Disk and Disk Group Utilities 67
Using Disks 68
Initializing and Adding Disks 68
Formatting the Disk Media 68
Volume Manager Disk Installation 68
Removing Disks 71
Moving Disks 72
Detecting and Replacing Failed Disks 73
Hot-Relocation 73
Modifying `vxrelocd` 75
Displaying Spare Disk Information 76
Moving Relocated Subdisks 76
Detecting Failed Disks 77
Partial Disk Failure 78
Complete Disk Failure 79

Replacing Disks 80
Using Disk Groups 82
Creating a Disk Group 82
Using Disk Groups 83
Removing a Disk Group 83
Moving Disk Groups Between Systems 83
Renaming Disk Groups 86
Reserving Minor Numbers for Disk Groups 87
Using Special Devices 88
Using `vxdisk` for Special Encapsulations 88
Using `vxdisk` for RAM Disks 90
Using `vxdisk` To Display Multipathing Information 91
4. VxVM Performance Monitoring 93
Introduction 93
Performance Guidelines 94
Data Assignment 94
Striping 94
Mirroring 95
Mirroring and Striping 97
Striping and Mirroring 97
Using RAID-5 98
Performance Monitoring 98
Performance Priorities 98
Getting Performance Data 99
Obtaining I/O Statistics (`vxstat`) 99
Tracing I/O (`vxtrace`) 100

Using Performance Data 100
Using I/O Statistics 100
Using I/O Tracing 104
Tuning the Volume Manager 104
General Tuning Guidelines 104
Tunables 105
vol_maxvol 105
vol_subdisk_num 105
vol_maxioctl 106
vol_maxspecialio 106
vol_maxio 106
vol_maxkiocount 107
vol_default_iodelay 107
voldrl_min_regionsz 107
voldrl_max_drtregs 108
vol_maxparallelio 108
vol_mvr_maxround 108
voliot_iobuf_limit 108
voliot_iobuf_max 109
voliot_iobuf_default 109
voliot_errbuf_default 109
voliot_max_open 110
vol_checkpt_default 110
volraid_rsrtransmax 110
voliomem_kvmap_size 111
voliomem_base_memory 111
voliomem_max_memory 111

voliomem_chunk_size 111
Tuning for Large Systems 112
The Number of Configuration Copies for a Disk Group 112

5. Volume Manager Cluster Functionality 115

Introduction 115
Cluster Functionality Overview 116
Shared Volume Manager Objects 116
How Cluster Volume Management Works 117
Configuration & Initialization 119
Cluster Reconfiguration 120
Volume Reconfiguration 121
Node Shutdown 122
Node Abort 123
Cluster Shutdown 123
Disks in VxVM Clusters 124
Dirty Region Logging and Cluster Environments 125
Log Format and Size 125
Compatibility 126
How DRL Works in a Cluster Environment 127
Cluster-related Volume Manager Utilities and Daemons 128
vxclust 129
vxconfigd 130
vxconfigd Recovery 131
vxdg 132
vxdisk 135
vxrecover 136

vxdctl 137
vxstat 138
Error Messages 139
Cluster Terminology 146
A. Volume Manager Error Messages 149
Introduction 149
Logging Error Messages 150
Volume Manager Configuration Daemon Error Messages 151
vxconfigd Usage Messages 151
vxconfigd Error Messages 153
vxconfigd Fatal Error Messages 178
vxconfigd Notice Messages 180
vxconfigd Warning Messages 183
Kernel Error Messages 198
Kernel Notice Messages 198
Kernel Warning Messages 202
Kernel Panic Messages 211
Utility Error Messages 212
B. Disk Array Overview 213
Introduction 213
Disk Array Overview 213
Redundant Arrays of Independent Disks (RAID) 214
RAID-0 215
RAID-1 215
RAID-2 215
RAID-3 215
RAID-4 218

RAID-5 218
Multipathed disk arrays 219
Active / Passive type disk arrays 220
Active / Active type disk arrays 220
Glossary 221
Index 233

Preface

The *VERITAS Volume Manager® Administrator's Reference Guide* provides information on the Volume Manager concepts and recovery procedures.

Audience

This guide is intended for system administrators responsible for installing, configuring, and maintaining systems under the control of the VERITAS Volume Manager.

This guide assumes that the user has a:

- working knowledge of the UNIX operating system
- basic understanding of system administration
- basic understanding of volume management

Scope

The purpose of this guide is to provide the system administrator with a thorough knowledge of the procedures and concepts involved with volume management and system administration using the Volume Manager. This guide includes guidelines on how to take advantage of various Volume Manager features, instructions on how to use Volume Manager commands to create and manipulate objects, and information on how to recover from disk failures.

Organization

This guide is organized as follows:

- Chapter 1, “Recovery**,”** describes how to preserve and recover data with the help of Volume Manager. It discusses ways to prevent data loss and / or loss of system availability due to disk failure.
- Chapter 2, “RAID-5 Volume Recovery,” discusses how to preserve and recover data when using RAID-5 devices.
- Chapter 3, “Disks and Disk Groups,” discusses disk and disk group administration using the Volume Manager.
- Chapter 4, “VxVM Performance Monitoring,” provides performance management and configuration guidelines for use with Volume Manager.
- Chapter 5, “Volume Manager Cluster Functionality,” describes the cluster functionality available with the Volume Manager.
- Appendix A, “Volume Manager Error Messages,” contains a list of error messages generated by Volume Manager, accompanied by a description of each message and a suggested user action.
- Appendix B, “Disk Array Overview,” describes how Volume Manager works with physical disks to store data.
- The “Glossary" defines various terms associated with the Volume Manager.

Using This Guide

This guide contains instructions for performing Volume Manager system administration functions. Volume Manager administration functions can be performed through one or more of the following interfaces:

- a set of complex commands
- a single automated command (`vxassist`)
- a menu-driven interface (`vxdiskadm`)
- the Storage Administrator (graphical user interface)

This guide describes how to use the various Volume Manager command line interfaces for Volume Manager administration. Details on how to use the Storage Administrator graphical user interface can be found in the *VERITAS*

Volume Manager Storage Administrator Administrator's Guide.". Detailed descriptions of the Volume Manager utilities, the options for each utility, and details on how to use them are located in the Volume Manager manual pages.

Note: Most of the Volume Manager commands require superuser or other appropriate privileges.

Related Documents

The following documents provide information related to the Volume Manager:

- The *VERITAS Volume Manager Getting Started Guide*, provides administrators with an overview of the Volume Manager and its features. It also provides general information about Volume Manager installation and setup.
- The *VERITAS Volume Manager Storage Administrator Administrator's Guide*, provides administrators with information on how to perform various Volume Manager operations through the graphical user interface (SA).
- The *VERITAS Volume Manager Command Line Interface Administrator's Guide* provides instructions for performing Volume Manager command line administration tasks. Volume Manager administration tasks can be performed by using the available interfaces.

Conventions

The following table describes the typographic conventions used in this guide.

Typeface	Usage	Examples

`courier`	Computer output; user input; names of commands, files, and directories	`$You have mail.`
		The `cat` command displays files.
		`$ls -a`

italics	New terms; document titles; words to be emphasized; variables to be substituted with a real name or value	`$cat` *filename* Refer to the *User's Guide* for details.
bold	Glossary terms	

Recovery

Introduction

The VERITAS Volume Manager protects systems from disk failures and helps you to recover from disk failures. This chapter describes recovery procedures and information to help you prevent loss of data or system access due to disk failures. It also describes possible plex and volume states.

For information about protecting your system, refer to Chapter 3, “Volume Manager Initialization and Setup,” in the *VERITAS Volume Manager Getting Started Guide*.

The following topics are covered in this chapter:

- The UNIX Boot Process
  - Booting After Failures
- Possible Root (/), swap, and usr Configurations
  - Repairing Root (/) or /usr File Systems on Volumes
  - Backing Up and Restoring the Root File System
- Failures and Recovery Procedures
  - Failures in UNIX Partitioning
  - Failures Accessing the Boot Device
  - Failures Due to Incorrect Entries in /etc/vfstab
  - Failures Due to Missing or Damaged /etc/system
  - Failures Due To Booting From Unusable or Stale Plexes

- Hot-Relocation and Boot Disk Failures
- Re-Adding and Replacing Boot Disks
- Reattaching Disks
- Reinstallation Recovery
  - General Reinstallation Information
  - Reinstallation and Reconfiguration Procedures
- Plex and Volume States
  - Plex States
  - The Plex State Cycle
  - Plex Kernel State
  - Volume States
  - Volume Kernel State

The UNIX Boot Process

A Sun SPARC system prompts for the `boot` command unless the `autoboot` flag has been set in the nonvolatile storage area used by the firmware. Machines with older PROMs have different prompts than the prompt for the newer V2 and V3 versions of PROM. These newer versions of PROM are also known as OpenBoot PROMs (OBP). The `boot` command has a different syntax for these two types of PROMs:

`ok boot [`*OBP names*`] [`*filename*`] [`*boot-flags*`]`

OBP names specify the open boot PROM designations. For example, on Desktop SPARC systems, the designation

`/sbus/esp@0,800000/sd@3,0:a`

indicates a SCSI disk (`sd`) at target 3, lun 0 on the SCSI bus, with the `esp` host adapter plugged into slot 0.

Note: With Volume Manager, you can use boot disk alias names. These aliases can take the form of Volume Manager provided names (for example, `vx-rootdisk` or `vx-disk01`) or operating system provided names (for example, `disk1`). You can view a list of possible bootable devices by using this command at the OpenBoot `OK` prompt: `devalias`

The *filename* is the name of a standalone program to the `boot` program. The default is to boot `/kernel/unix` from the `root` partition. You can specify another program (such as `/stand/diag`) on the command line. Some versions of the firmware allow the default filename to be saved in the nonvolatile storage area of the system.

The `boot` program interprets the `-a` flag to mean “ask me” and prompts you for the name of the standalone program to boot. The `-a` flag is then passed to the standalone program.

Note: A system running Volume Manager with rootability does not boot with the defaults presented by the `-a` flag. See “`/etc/system` Copy Available” for the correct responses to `boot -a`.

Flags are not interpreted by the `boot` program. The `boot` program passes all boot-flags to the file identified by *filename*. Boot. See the `kernel`(1) and `kadb`(1M) manual pages for information on the options available with the default standalone program, `/kernel/unix`.

Booting After Failures

If the root disk is mirrored, you can use the alternate boot disk to boot the system if the primary boot disk fails. To boot the system after failure of the primary boot disk, follow these steps:

1. Check for aliased VM disks using the `devalias` command at the OpenBoot command prompt.

   Disks that are suitable mirrors of the root disk are listed with the name `vx-`*medianame*, where *medianame* is the disk media name for the disk with the candidate root file system.

2. Enter this command:

```
ok boot alias_name
```

where *alias_name* is the aliased name of the selected disk.

If a selected disk contains a root mirror that is stale, `vxconfigd` displays an error stating that the mirror is unusable and lists any nonstale alternate bootable disks.

Possible Root (/), swap, and usr Configurations

During installation, different configurations are possible for `root`, `swap`, and `usr` file systems. These cases are possible:

- `usr` is a directory under the `root` and no separate partition is allocated for it. In this case, `usr` becomes part of the `rootvol` volume when the root disk is encapsulated and put under Volume Manager control.
- `usr` is on a separate partition that is on the `root` disk. In this case, a separate volume is created for the `usr` partition. `vxmirror` mirrors the `usr` volume on the destination disk.
- `usr` is on a separate partition that is not on the `root` disk. In this case, a volume is created for the `usr` partition only if that disk is encapsulated by Volume Manager. Note that in this case, encapsulating the `root` disk and having mirrors of the root volume does not help if the `usr` partition becomes inaccessible for any reason. It is recommended that you encapsulate both the disk containing the `usr` partition and the `root` disk, and have mirrors for the `usr`, `rootvol`, and `swapvol` volumes for maximum availability of the system.

The `rootvol` volume must exist in the `rootdg` disk group. Refer to "Boot-time Volume Restrictions," in Chapter 2, "Volume Manager Operations," of the *VERITAS Volume Manager Getting Started Guide*, for information on `rootvol` and `usr` volume restrictions.

Solaris 2.x allows you to put `swap` partitions on any disk; it does not need an initial `swap` area during early phases of the boot process. By default, the Solaris 2.x installation chooses partition 0 on the selected root disk as the `root` partition, and partition 1 as the `swap` partition. However, it is possible to have the `swap` partition on a partition not located on the root disk. In such cases, you are advised to encapsulate that disk and create mirrors for the `swap`

volume. If you do not do this, damage to the `swap` partition eventually causes the system to crash. It may be possible to boot the system, but having mirrors for the `swapvol` volume prevents system failures.

Repairing Root (/) or `/usr` File Systems on Volumes

If the root (/) or `/usr` file system becomes unusable, it is desirable to boot from a network-mounted `root` file system or from a valid backup. The backup must have all relevant filesystem partitions on the root disk. Also, you need a printout of the root disk partition table before the root disk was encapsulated.

This task is made more difficult when the `root` or `/usr` file system is defined on a mirrored volume. Changes to the partition that underlies one of the mirrors can result in corruption when the Volume Manager later boots and presumes that the mirrors are reasonably synchronized.

There are two workarounds for this:

- The simplest workaround is to mount one plex of the `root` or `/usr` file system, repair it, unmount it, and use `dd` to copy the fixed plex to all other plexes. However, this can be error prone.
- Another workaround is to restore the system from a valid backup tape. The procedure is described below. This procedure does not require the operating system to be installed from the base CD-ROM.

The procedure described below provides a simple, efficient, and reliable means of recovery when both the root disk and its mirror are damaged.

Recovering a Volume Manager Root Disk (and Root Mirror) From Backup Tape.

This procedure assumes that you have:

- A current full backup of all the file systems on the original Volume Manager root disk.
- A new boot disk installed to replace the original failed boot disk if the original boot disk was physically damaged.

This procedure requires the reinstallation of the Volume Manager root disk. To prevent the loss of data on disks not involved in the reinstallation, you should only involve the Volume Manager root disk in the reinstallation procedure.

Several of the automatic options for installation access disks other than the root disk without requiring confirmation from the administrator. Therefore, it is advisable that you disconnect all other disks (containing volumes) from the system prior to starting this procedure. Disconnecting the other disks ensures that they are unaffected by the reinstallation. Reconnect these disks after the procedure has been completed.

Procedure

To illustrate the procedure in the listing below, assume that the (new) boot disk is `c0t0d0` and that you need to recover both the `/` and `/usr` file systems, `s0` and `s6` respectively.

1. Boot the operating system from the CD ROM.
2. Using `format` to create identical partitions on the (new) boot disk (`c0t0d0`) to contain the file systems previously on the original boot disk.

Note: Since you need to restore the file systems, this means that you can have a maximum of seven partitions. Additionally, as you need to re-encapsulate this disk momentarily, you can only have a maximum of five partitions as two are required for the private and public regions on the disk.

---

3. Mount `/dev/rdsk/c0t0d0s0` on `/a/root` and restore the root file system from tape and install a bootblock device on `/a/root` using `installboot.`
4. Mount `/dev/rdsk/c0t0d0s6` on `/a/usr` and restore the `/usr` file system from tape.
5. Modify the restored root file system as follows to:
   - touch `/a/root/etc/vx/reconfig.d/state.d/install-db`
   - modify the `/a/root/etc/system` by removing these two lines:

     ```
     rootdev:/pseudo/vxio@0:0
     set vxio:vol_rootdev_is_volume=1
     ```
   - modify `/a/root/etc/vfstab` by replacing the Volume Manager volume device entries with the standard disk devices `/dev/dsk/c0t0d0s0` and `/dev/dsk/c0t0d0s6`.
6. Reboot the system from the (new) boot disk. This allows the boot disk to come up thinking that Volume Manager is NOT installed.

The next step in the procedure depends on whether there are:

- Other disks in the old `rootdg` that are NOT used as root disk mirrors. Go to step 7.
- Only root disk mirrors in the old `rootdg`. Go to step 8.

7. Other disks in the old `rootdg` that are NOT used as root disk mirrors.

   a. Remove files involved with the installation that are no longer needed:

   ```
   rm -r /etc/vx/reconfig.d/state.d/installdb
   ```

   b. Start the VXVM I/O daemons:

   ```
   vxiod set 10
   ```

   c. Start the VXVM Configuration daemon in disabled mode:

   ```
   vxconfigd -m disable
   ```

   d. Initialize the `vxconfigd` daemon:

   ```
   vxdctl init
   ```

   e. Enable `vxconfigd`:

   ```
   vxdctl enable
   ```

The above steps should bring in the old `rootdg` minus the boot disk which Volume Manager will think has failed

Use the `vxedit` command (or the Volume Manager GUI) to remove the old root disk volumes and the root disk itself.

Then use the `vxdiskadm` command to encapsulate the (new) boot disk and initialize any disks that are to serve as root disk mirrors. After the required reboot, mirror the root disk to the root disk mirrors.

8. Only root disk mirrors in the old rootdg

Run the `vxinstall` command to encapsulate the (new) boot disk, and initialize the root disk mirror(s). After the required reboot, mirror the root disk to the root disks mirrors.

The procedure is completed.

Backing Up and Restoring the Root File System

It is a good idea to back up your `root` file system so that it can be restored if it is ever damaged.

If you are using the `ufs` file system, you can back up your `root` file system by using this command:

```
/usr/lib/fs/ufs/ufsdump [dump-options] /dev/vx/rdsk/rootvol
```

You can then restore your `root` file system after a failure as follows:

1. Boot from a CD-ROM or network-mounted root file system, then get the Volume Manager running (see “Repairing Root (/) or /usr File Systems on Volumes”).
2. Mount and restore the root file system by using these commands:

```
newfs /dev/vx/rdsk/rootvol
mount /dev/vx/dsk/rootvol /mnt
cd /mnt
/usr/lib/fs/ufs/ufsrestore [restore-options]
```

Failures and Recovery Procedures

While there are many types of failures that can prevent a system from booting, the same basic procedure can be taken to bring the system up. When a system fails to boot, you should first try to identify the failure by the evidence left behind on the screen and then attempt to repair the problem (for example, by turning on a drive that was accidentally powered off). If the problem is one that cannot be repaired (such as data errors on the boot disk), boot the system from an alternate boot disk (containing a mirror of the `root` volume) so that the damage can be repaired or the failing disk can be replaced.

This section outlines some possible failures and provides instructions on the corrective actions.

Failures in UNIX Partitioning

Once the boot program has loaded, it attempts to access the boot disk through the normal UNIX partition information. If this information is damaged, the boot program fails with an error:

```
File just loaded does not appear to be executable
```

If this message appears during the boot attempt, the system should be booted from an alternate boot disk. While booting, most disk drivers display errors on the console about the invalid UNIX partition information on the failing disk. The messages are similar to this:

```
WARNING: unable to read label
WARNING: corrupt label_sdo
```

This indicates that the failure was due to an invalid disk partition. You can attempt to re-add the disk as described in “Re-Adding a Failed Boot Disk.” However, if the reattach fails, then the disk needs to be replaced as described in “Replacing a Failed Boot Disk.”

Failures Accessing the Boot Device

Early in the boot process, immediately following system initialization, the messages are similar to this:

```
SCSI device 0,0 is not responding
Can’t open boot device
```

This means that the system PROM was unable to read the boot program from the boot drive. Common causes for this problem are:

- The boot disk is not powered on.
- The SCSI bus is not terminated.
- There is a controller failure of some sort.
- A disk is failing and locking the bus, preventing any disks from identifying themselves to the controller, and making the controller assume that there are no disks attached.

The first step in diagnosing this problem is to check carefully that everything on the SCSI bus is in order. If disks are powered off or the bus is unterminated, correct the problem and reboot the system. If one of the disks has failed, remove the disk from the bus and replace it.

If no hardware problems are found, the error is probably due to data errors on the boot disk. In order to repair this problem, attempt to boot the system from an alternate boot disk (containing a mirror of the root volume). If you are unable to boot from an alternate boot disk, there is still some type of hardware problem. Similarly, if switching the failed boot disk with an alternate boot disk fails to allow the system to boot, this also indicates hardware problems.

Failures Due to Incorrect Entries in `/etc/vfstab`

When the root disk is encapsulated and put under Volume Manager control, as part of the normal encapsulation process, volumes are created for all of the partitions on the disk. Volume Manager modifies the `/etc/vfstab` to use the corresponding volumes instead of the disk partitions. Care should be taken while editing the `/etc/vfstab` file manually. The most important entries are those corresponding to `/` and `/usr`. The `vfstab` that existed prior to Volume Manager installation is saved in `/etc/vfstab.prevm`.

Damaged / Entry in `/etc/vfstab`

If the entry in `/etc/vfstab` for `/` is lost or is incorrect, the system boots in single-user mode. Messages similar to the following are displayed:

```
File just loaded does not appear to be executable
```

It is recommended that you run `fsck` at this point:

```
fsck /dev/vx/rdsk/rootvol
```

At this point in the boot process, `/` is not yet mounted `read/write`. Since the entry in `/etc/vfstab` was either incorrect or deleted, mount `/` as `read/write` manually, by using this command:

```
mount -o remount /dev/vx/dsk/rootvol
```

After mounting `/` as `read/write`, exit the shell. The system prompts for the run level. For multi-user mode, enter run level `3`:

```
ENTER RUN LEVEL (0-6,s or S): 3
```

Restore the entry in /etc/vfstab for / after the system boots.

Damaged /usr Entry in /etc/vfstab

/etc/vfstab has an entry for /usr only if /usr is located on a separate disk partition. After encapsulation of the disk containing the /usr partition, Volume Manager changes the entry in /etc/vfstab to use the corresponding volume.

In the event of loss of the entry for /usr from /etc/vfstab, the system cannot be booted (even if you have mirrors of the /usr volume).

In this case, boot the system from the CD-ROM and restore the /etc/vfstab. (Refer to the steps in "Repairing Root (/) or /usr File Systems on Volumes.")

Failures Due to Missing or Damaged /etc/system

Note: Do not edit any entries in /etc/system that are added by Volume Manager. All Volume Manager entries are enclosed with *vxvm_START and *vxvm_END.

It is advisable to make a copy of /etc/system in the root file system before making any changes to it. The saved system file can then be specified to the boot program if changes to the new /etc/system file are incorrect. To specify the saved system file to the boot program, boot the system with the command: boot -a. When the system prompts for the name of the system file, enter the path of the saved system file.

/etc/system Copy Not Available

If the /etc/system file is damaged and the saved copy of the system file is not available, the system cannot be booted with the Volume Manager rootability feature on. The system can be booted without Volume Manager rootability (i.e., without the rootvol being the /) if /usr is not a volume.

To boot the system without Volume Manager rootability, follow the steps outlined in "Repairing Root (/) or /usr File Systems on Volumes" to:

1. Start the Volume Manager from the CD-ROM. See the *VERITAS Volume Manager Installation Guide* for more information.
2. Run the `fixmountroot` command.
3. Make and mount `/tmp/rootvol`.

You can then edit the file `/tmp/rootvol/etc/system` and do any other necessary repairs.

After booting the system, make the following entries in `/etc/system`:

```
* vxvm_START
rootdev:/pseudo/vxio@0:0
set vxio:vol_rootdev_is_volume=1
* vxvm_END
```

You should also forceload all of the drivers required for the root mirror disks. To do this, edit the file `/etc/system` so that it contains a line of the following form for each driver:

`forceload: drv/`*`driver_name`*

The driver names for these disks can be obtained by doing a long listing on `/dev/dsk/`*`root_device`*. An example of a driver name is `io-unit`.

`/etc/system` Copy Available

If the `/etc/system` file is damaged and a saved copy of the `etc/system` file is available, the system can be booted with Volume Manager rootability.

To boot the system with Volume Manager rootability, boot the system with the following command and responses (pressing Return to accept defaults for all prompts *except* the root device name):

```
ok boot -a
 .
 .
Rebooting with command: -a
Boot device: /iommu/sbus/espdma/esp/sd@5,0   File and args: -a
Enter filename [/kernel/unix]:
Name of system file [/etc/system.sav]:
Name of default directory for modules [/kernel /usr/kernel]:
Enter name of device instance number file [/etc/path_to_inst]:
root file system type [ufs]:
Enter physical name of root device
[/iommu.................................]:/pseudo/vxio@0:0
```

/etc/system Not Available and /usr is Volume

If the etc/system file is damaged or lost and a backup copy is not available, and /usr is a volume, the system must be booted from the CD-ROM (using the steps outlined in “Repairing Root (/) or /usr File Systems on Volumes”). Once this is done, mount the root volume and edit the etc/system file on it. Create the following entries in the etc/system file:

```
* vxvm_START
rootdev:/pseudo/vxio@0:0
set vxio:vol_rootdev_is_volume=1
set vxio:vol_swapdev_is_volume=1
* vxvm_END
```

You should also forceload all of the drivers required for the root mirror disks (as described previously). After these changes, reboot the system from the same root partition on which the system file was restored.

Failures Due To Booting From Unusable or Stale Plexes

If a disk is unavailable when the system is running, any mirrors of volumes that reside on that disk become stale. This means that the data on that disk is inconsistent relative to the other mirrors of that volume. During the boot process, the system accesses only one copy of the `root` volume (the copy on the boot disk) until a complete configuration for this volume can be obtained.

If it turns out that the plex of this volume that was used for booting is stale, the system must be rebooted from an alternate boot disk that contains nonstale plexes. This problem can occur, for example, if the system was booted from one of the disks made bootable by Volume Manager with the original boot disk turned off. The system boots normally, but the plexes that reside on the unpowered disk are stale. If the system reboots from the original boot disk with the disk turned back on, the system boots using that stale plex.

Another possible problem can occur if errors in the Volume Manager headers on the boot disk prevent Volume Manager from properly identifying the disk. In this case, Volume Manager does not know the name of that disk. This is a problem because plexes are associated with disk names, so any plexes on the unidentified disk are unusable.

A problem can also occur if the root disk has a failure that affects the root volume plex. At the next boot attempt, the system still expects to use the failed root plex for booting. If the root disk was mirrored at the time of the failure, an alternate root disk (with a valid root plex) can be specified for booting.

If any of these situations occur, the Volume Manager utility `vxconfigd` notes it when it is configuring the system as part of the `init` processing of the boot sequence. `vxconfigd` displays a message describing the error and what can be done about it, then it halts the system. For example, if the plex `rootvol-01` of the root volume `rootvol` on disk `rootdisk` is stale, `vxconfigd` can display this message:

```
vxvm:vxconfigd: Warning Plex rootvol-01 for root volume is stale or
unusable.
vxvm:vxconfigd: Error: System boot disk does not have a valid root
plex
Please boot from one of the following disks:
Disk: disk01              Device: c0t1d0s2
vxvm:vxconfigd:    Error: System startup failed
The system is down.
```

This informs the administrator that the alternate boot disk named `disk01` contains a usable copy of the root plex and should be used for booting. When this message is displayed, you should reboot the system from the alternate boot disk.

Once the system has booted, the exact problem needs to be determined. If the plexes on the boot disk were simply stale, they are caught up automatically as the system comes up. If, on the other hand, there was a problem with the private area on the disk or the disk failed, you need to re-add or replace the disk.

If the plexes on the boot disk are unavailable, you should receive mail from Volume Manager utilities describing the problem. Another way to determine the problem is by listing the disks with the `vxdisk` utility. In the above example, if the problem is a failure in the private area of `rootdisk` (such as due to media failures or accidentally overwriting the Volume Manager private region on the disk), `vxdisk list` shows this display:

```
DEVICE      TYPE       DISK       GROUP    STATUS
  -           -        rootdisk   rootdg   failed was: c0t3d0s2
c0t1d0s2    sliced     disk01     rootdg   ONLINE
```

Hot-Relocation and Boot Disk Failures

If the boot (root) disk fails and it is mirrored, hot-relocation automatically attempts to replace the failed root disk mirror with a new mirror. To do this, hot-relocation uses a surviving mirror of the root disk to create a new mirror on either a spare disk or a disk with sufficient free space. This ensures that there are always at least two mirrors of the root disk that can be used for booting. The hot-relocation daemon also calls the `vxbootsetup` utility, which configures the disk with the new mirror as a bootable disk.

Hot-relocation can fail for a root disk if the `rootdg` disk group does not contain sufficient spare or free space to fit the volumes from the failed root disk. The `rootvol` and `swapvol` volumes require contiguous disk space. If the root volume and other volumes on the failed root disk cannot be relocated to the same new disk, each of these volumes can be relocated to a different disk.

Mirrors of `rootvol` and `swapvol` volumes must be cylinder-aligned, so they can only be created on disks with enough space to allow their subdisks to begin and end on cylinder boundaries. Hot-relocation fails if these disks are not available.

Re-Adding and Replacing Boot Disks

Data that is not critical for booting the system is only accessed by the Volume Manager after the system is fully operational, so it does not have to be located in specific areas. The Volume Manager can find it. However, boot-critical data must be placed in specific areas on the bootable disks for the boot process to find it.

On some systems, the controller-specific actions done by the disk controller in the process and the system BIOS constrain the location of this critical data.

When a disk fails, there are two paths that can be taken to correct the problem:

1. If the error(s) are transient or correctable, the same disk can be re-used; this is known as *re-adding* a disk. In some cases, reformatting a failed disk or doing a surface analysis to rebuild the alternate-sector mappings are sufficient to make a disk re-usable and a candidate for re-addition.
2. If the disk has truly failed, it should be replaced.

The following sections describe how to re-add or replace a failed boot disk.

Re-Adding a Failed Boot Disk

Re-adding a disk is the same procedure as replacing the disk, except that the same physical disk is used. Normally, a disk that needs to be re-added has been *detached*. This means that Volume Manager has detected the disk failure and has ceased to access the disk.

Note: Your system may use a *device name* or *path* that differs from the examples. See Chapter 1, “Understanding Volume Manager,” in the VERITAS *Volume Manager Getting Started Guide*, for more information on device names.

For example, consider a system that has two disks, `disk01` and `disk02`, which are normally mapped into the system configuration during boot as disks `c0t0d0s2` and `c0t1d0s2`, respectively. A failure has caused `disk01` to become detached. This can be confirmed by listing the disks with the `vxdisk` utility with this command:

```
vxdisk list
```

`vxdisk` displays this (example) list:

```
DEVICE       TYPE      DISK      GROUP      STATUS
c0t0d0s2     sliced    -         -          error
c0t1d0s2     sliced    disk02    rootdg     online
 -            -        disk01    rootdg     failed was:c0b0t0d0s0
```

Note that the disk `disk01` has no device associated with it, and has a status of `failed` with an indication of the device that it was detached from. It is also possible that the device `c0b0t0d0s0` would not be listed at all. This occurs if the disk fails totally and the disk controller does not detect it on the bus (for systems that use a bus).

In some cases, the `vxdisk list` output can differ. For example, if the boot disk has uncorrectable failures associated with the UNIX partition table. There can be a missing root partition that cannot be corrected but there are no errors in the Volume Manager private area. The output of the `vxdisk list` command displays this (example) list:

```
DEVICE       TYPE      DISK      GROUP      STATUS
c0t0d0s2     sliced    disk01    rootdg     online
c0t1d0s2     sliced    disk02    rootdg     online
```

However, because the error was not correctable by the described procedures, the disk is viewed as failed. In this case, it is necessary to detach the failing disk from its device manually. This is done using the “Remove a disk for replacement” function of the `vxdiskadm` utility (see the `vxdiskadm`(1M) manual page or the *VERITAS Volume Manager Command Line Interface Administrator’s Guide*, for more information about `vxdiskadm`). Once the disk is detached from the device, any special procedures for correcting the problem can be followed (such as reformatting the device).

To re-add the disk, use the “Replace a failed or removed disk” function of the `vxdiskadm` utility to replace the disk, and select the *same* device as the replacement. Using the previous examples, you replace `disk01` with the device `c0t0d0s2` or `c0b0t0d0s2` (for systems that use a bus).

If hot-relocation is enabled when a mirrored boot disk fails, it attempts to create a new mirror and remove the failed subdisks from the failing boot disk. If a re-add succeeds after a successful hot-relocation, the root and/or other volumes affected by the disk failure no longer exist on the re-added disk. However, the re-added disk can still be used for other purposes.

If a re-add of the disk fails, the disk should be replaced.

Replacing a Failed Boot Disk

When a boot disk is to be replaced, the system should first be booted off an alternate boot disk. If the failing disk is not detached from its device, it should be manually detached using the “Remove a disk for replacement” function of `vxdiskadm`. See the `vxdiskadm`(1M) manual page or the *VERITAS Volume Manager Command Line Interface Administrator’s Guide*, for more information about `vxdiskadm`. Once the disk is detached, shut down the system and replace the hardware.

The replacement disk must have at least as much storage capacity as was in use on the disk being replaced. The replacement disk must be large enough so that the region of the disk for storing subdisks can accommodate all subdisks of the original disk at their current disk offsets. To determine the minimum size of a replacement disk, you need to determine how much space was in use on the disk that failed.

To approximate the size of the replacement disk, use the command:

```
vxprint -st -e 'sd_disk="diskname"'
```

From the resulting output, add the values under the `DISKOFFS` and `LENGTH` columns for the last subdisk listed. The total is in 512-byte multiples. Divide the sum by 2 for the total in kilobytes.

Note: Disk sizes reported by manufacturers do not usually represent usable capacity. Also, some manufacturers report millions of bytes rather than megabytes, which are not equivalent.

Once a replacement disk has been found, shut down the system cleanly and replace the necessary hardware. When the hardware replacement is complete, boot the system. You use the `vxdiskadm` “Replace a failed or removed disk” function to replace the failing disk with the new device that was added.

Reattaching Disks

You can do a disk reattach operation if a disk has a full failure and hot-relocation is not possible, or if the Volume Manager is started with some disk drivers unloaded and unloadable (causing disks to enter the failed state). If the problem is fixed, you can use the `vxreattach` command to reattach the disks without plexes being flagged as stale. However, the reattach must occur before any volumes on the disk are started.

The `vxreattach` command is called as part of disk recovery from the `vxdiskadm` menus and during the boot process. If possible, `vxreattach` reattaches the failed disk media record to the disk with the same device name. The reattach occurs in the same disk group it was located in before and retains its original disk media name.

After a reattach takes place, recovery may not be necessary. The reattach can fail if the original (or another) cause for the disk failure still exists.

The command `vxreattach -c` checks whether a reattach is possible, but does not do the operation. Instead, it displays the disk group and disk media name where the disk can be reattached.

Refer to the `vxreattach`(1M) manual page for more information on the `vxreattach` command.

Reinstallation Recovery

Reinstallation is necessary if all copies of your root (boot) disk are damaged, or if certain critical files are lost due to file system damage. When a failure of either of these types occurs, you must reinstall the entire system, because there is currently no method of restoring the root file system from backup.

If these types of failures occur, attempt to preserve as much of the original Volume Manager configuration as possible. Any volumes not directly involved in the failure may be saved. You do not have to reconfigure any volumes that are preserved.

General Reinstallation Information

This section describes procedures used to reinstall Volume Manager and preserve as much of the original configuration as possible after a failure.

System reinstallation destroys the contents of any disks that are used for reinstallation.

All Volume Manager related information is removed during reinstallation. Data removed includes data in private areas on removed disks that contain the disk identifier and copies of the Volume Manager configuration, The removal of this information makes the disk unusable as a Volume Manager disk.

The system root disk is always involved in reinstallation. Other disks can also be involved. If the root disk was placed under Volume Manager control, either during Volume Manager installation or by later encapsulation, that disk and any volumes or mirrors on it are lost during reinstallation. Any other disks that are involved in the reinstallation, or that are removed and replaced, can lose Volume Manager configuration data (including volumes and mirrors).

If a disk, including the root disk, is not under Volume Manager control prior to the failure, no Volume Manager configuration data is lost at reinstallation. Any other disks can be replaced by following the procedures in Chapter 3, “Disks and Disk Groups.”

Although it simplifies the recovery process after reinstallation, not having the root disk under Volume Manager control increases the possibility of a reinstallation being necessary. By having the root disk under Volume Manager control and creating mirrors of the root disk contents, you can eliminate many of the problems that require system reinstallation.

When reinstallation is necessary, the only volumes saved are those that reside on, or have copies on, disks that are not directly involved with the failure and reinstallation. Any volumes on the root disk and other disks involved with the failure and/or reinstallation are lost during reinstallation. If backup copies of these volumes are available, the volumes can be restored after reinstallation. On some systems, the exceptions are the `root`, `stand`, and `usr` file systems, which cannot be restored from backup.

Reinstallation and Reconfiguration Procedures

To reinstall the system and recover the Volume Manager configuration, follow this procedure. These steps are described in detail in the sections that follow:

1. Prepare the system for installation.

   This includes replacing any failed disks or other hardware, and detaching any disks not involved in the reinstallation.

2. Install the operating system.

   Do this by reinstalling the base system and any other nonrelated Volume Manager packages.

3. Install Volume Manager.

   Add the Volume Manager package, but *do not* execute the `vxinstall` command.

4. Recover the Volume Manager configuration.

5. Clean up the Volume Manager configuration.

   This includes restoring any information in volumes affected by the failure or reinstallation, and recreating system volumes (`rootvol`, `swapvol`, `usr`, and other system volumes.).

Preparing the System for Reinstallation

To prevent the loss of data on disks not involved in the reinstallation, you should only involve the root disk in the reinstallation procedure.

Note: Several of the *automatic* options for installation access disks other than the root disk without requiring confirmation from the administrator. It is advised that you disconnect all other disks containing volumes from the system prior to reinstalling the operating system.

Disconnecting the other disks ensures that they are unaffected by the reinstallation. For example, if the operating system was originally installed with a `home` file system on the second disk, it can still be recoverable. Removing the second disk ensures that the home file system remains intact.

Reinstalling the Operating System

Once any failed or failing disks have been replaced and disks not involved with the reinstallation have been detached, reinstall the operating system as described in your operating system documentation. Install the operating system prior to installing Volume Manager.

You must be sure that no disks other than the root disk are accessed in any way while the operating system installation is in progress, If anything is written on a disk other than the root disk, the Volume Manager configuration on that disk can be destroyed.

Note: During reinstallation, you can change the host ID (or name). It is recommended that you keep the existing host ID (name), as the sections that follow assume that you have not changed your host ID (name).

Reinstalling the Volume Manager

The installation of the Volume Manager has two parts:

- loading Volume Manager from CD-ROM
- initializing the Volume Manager

To reinstall the Volume Manager, follow the instructions for loading the Volume Manager (from CD-ROM) in the *VERITAS Volume Manager Installation Guide*.

Note: To reconstruct the Volume Manager configuration left on the nonroot disks, *do not* initialize the Volume Manager (using `vxinstall`) after the reinstallation.

On some systems, you can use `vxserial` to install the Volume Manager license key (see the `vxserial`(1M) manual page).

Recovering the Volume Manager Configuration

Once the Volume Manager package has been loaded, recover the Volume Manager configuration by following these steps:

1. Shut down the system.

2. Reattach the disks that were removed from the system.
3. Reboot the system.
4. When the system comes up, bring the system to single-user mode by using this command:

   ```
   shutdown -g0 -iS -y
   ```

5. When prompted, enter the password and press Return to continue.
6. Remove files involved with installation that were created when you loaded Volume Manager but are no longer needed. Use this command:

   ```
   rm -rf /etc/vx/reconfig.d/state.d/install-db
   ```

7. After the files are removed, start some Volume Manager I/O daemons. Start the daemons by entering the command:

   ```
   vxiod set 10
   ```

8. Start the Volume Manager configuration daemon, `vxconfigd`, in disabled mode by using this command:

   ```
   vxconfigd -m disable
   ```

9. Initialize the `vxconfigd` daemon with this command:

   ```
   vxdctl init
   ```

10. Initialize the DMP subsystem with this command:

    ```
    vxdctl initdmp
    ```

11. Enable `vxconfigd` with this command:

    ```
    vxdctl enable
    ```

The configuration preserved on the disks not involved with the reinstallation has now been recovered. However, because the root disk has been reinstalled, it does not appear to the Volume Manager as a Volume Manager disk. The configuration of the preserved disks does not include the root disk as part of the Volume Manager configuration.

If the root disk of your system and any other disks involved in the reinstallation were not under Volume Manager control at the time of failure and reinstallation, then the reconfiguration is complete at this point. If any other disks containing volumes or mirrors are to be replaced, follow the replacement procedures in Chapter 3, “Disks and Disk Groups.” There are several methods available to replace a disk; choose the method that you prefer.

If the root disk (or another disk) was involved with the reinstallation, any volumes or mirrors on that disk (or other disks no longer attached to the system) are now inaccessible. If a volume had only one plex contained on a disk that was reinstalled, removed, or replaced, then the data in that volume is lost and must be restored from backup.

In addition, the system `root` file system, `swap` area, (and on some systems `stand` area), and `/usr` file system are *no longer* located on volumes. To correct these problems, follow the instructions in “Configuration Cleanup.”

The hot-relocation facility can be started after `vxdctl enable` succeeds, but should actually be started only when the Administrator is sure that its services, when enabled and operating, will not interfere with other reconfiguration procedures. It is recommended that hot-relocation be started after completion of the “Final Reconfiguration” steps. See the “Starting Hot-Relocation” section for more information on starting hot-relocation.

Configuration Cleanup

The following sections describe how to clean up the configuration of your system after reinstallation of the Volume Manager.

The following types of cleanup are described:

- rootability cleanup
- volume cleanup
- disk cleanup

These sections are followed by reconfiguration information:

- rootability reconfiguration
- final reconfiguration

Rootability Cleanup

To begin the cleanup of the Volume Manager configuration, remove any volumes associated with rootability. This must be done if the root disk (and any other disk involved in the system boot process) was under Volume Manager control. The volumes to remove are:

- `rootvol`, that contains the `root` file system
- `swapvol`, that contains the `swap` area
- (on some systems) `standvol`, that contains the `stand` file system
- `usr`, that contains the `/usr` file system

To remove the root volume, use the `vxedit` command:

```
vxedit -fr rm rootvol
```

Repeat this command, using `swapvol` and `usr (standvol)` in place of `rootvol`, to remove the `swap, stand,` and `usr` volumes.

Volume Cleanup

After completing the rootability cleanup, you must determine which volumes need to be restored from backup. The volumes to be restored include those with all mirrors (all copies of the volume) residing on disks that have been reinstalled or removed. These volumes are invalid and must be removed, recreated, and restored from backup. If only some mirrors of a volume exist on reinitialized or removed disks, these mirrors must be removed. The mirrors can be re-added later.

To restore the volumes, do these steps:

1. Establish which VM disks have been removed or reinstalled by using this command:

   ```
   vxdisk list
   ```

   The Volume Manager displays a list of system disk devices and the status of these devices. For example, for a reinstalled system with three disks and a reinstalled root disk, the output of the `vxdisk list` command is similar to this:

```
DEVICE          TYPE      DISK      GROUP     STATUS
c0t0d0s2      sliced    -         -         error
c0t1d0s2      sliced    disk02    rootdg    online
c0t2d0s2      sliced    disk03    rootdg    online
-             -         disk01    rootdg failed was: c0t0d0s2
```

Note: Your system may use a *device name* that differs from the examples. See Chapter 1, "Understanding Volume Manager," in the *VERITAS Volume Manager Getting Started Guide*, for more information on device names.

The display shows that the reinstalled root device, `c0t0d0s2`, is not associated with a VM disk and is marked with a status of `error`. `disk02` and `disk03` were not involved in the reinstallation and are recognized by the Volume Manager and associated with their devices (`c0t1d0`s2 and `c0t2d0`s2). The former `disk01`, which was the VM disk associated with the replaced disk device, is no longer associated with the device (`c0t0d0`s2).

If other disks (with volumes or mirrors on them) had been removed or replaced during reinstallation, those disks would also have a disk device listed in `error` state and a VM disk listed as not associated with a device.

2. Once you know which disks have been removed or replaced, locate all the mirrors on failed disks. Use this command:

   ```
   vxprint -sF "%vname" -e'sd_disk = "disk"'
   ```

   where *disk* is the name of a disk with a `failed` status. Be sure to enclose the disk name in quotes in the command. Otherwise, the command returns an error message. The `vxprint` command returns a list of volumes that have mirrors on the failed disk. Repeat this command for every disk with a `failed` status.

3. Check the status of each volume. You use this command to print volume information:

   ```
   vxprint -th volume_name
   ```

where *volume_name* is the name of the volume to be examined. The `vxprint` command displays the status of the volume, its plexes, and the portions of disks that make up those plexes. For example, a volume named `v01` with only one plex resides on the reinstalled disk named `disk01`. The `vxprint -th v01` command outputs this display:

```
V  NAME         USETYPE  KSTATE   STATE    LENGTH   READPOL    PREFPLEX
PL NAME         VOLUME   KSTATE   STATE    LENGTH   LAYOUT     NCOL/WID  MODE
SD NAME         PLEX     DISK     DISKOFFS LENGTH   [COL/]OFF  DEVICE    MODE

v  v01          fsgen    DISABLED ACTIVE   24000    SELECT     -
pl v01-01       v01      DISABLED NODEVICE 24000    CONCAT     -         RW
sd disk01-06 v0101       disk01   245759   24000    0          c1t5d1    ENA
```

The only plex of the volume is shown in the line beginning with `pl`. The `STATE` field for the plex named `v01-01` is `NODEVICE`. The plex has space on a disk that has been replaced, removed, or reinstalled. The plex is no longer valid and must be removed.

Because `v01-01` was the only plex of the volume, the volume contents are irrecoverable except by restoring the volume from a backup. The volume must also be removed. If a backup copy of the volume exists, you can restore the volume later. Keep a record of the volume name and its length, as you will need it for the backup procedure.

4. To remove the volume `v01`, use the `vxedit` command:

   ```
   vxedit -r rm v01
   ```

   It is possible that only part of a plex is located on the failed disk. If the volume has a striped plex associated with it, the volume is divided between several disks. For example, the volume named `v02` has one striped plex striped across three disks, one of which is the reinstalled disk `disk01`. The `vxprint -th v02` command displays:

```
V   NAME        USETYPE   KSTATE   STATE    LENGTH   READPOL    PREFPLEX
PL  NAME        VOLUME    KSTATE   STATE    LENGTH   LAYOUT     NCOL/WID  MODE
SD  NAME        PLEX      DISK     DISKOFFS LENGTH   [COL/]OFF  DEVICE    MODE

v   v02         fsgen     DISABLED ACTIVE   30720    SELECT     v02-01
pl  v02-01      v02       DISABLED NODEVICE 30720     STRIPE     3/128     RW
```

```
sd  disk02-02  v02-01    disk02    424144    10240     0/0       c1t2d0    ENA
sd  disk01-05  v02-01    disk01    620544    10240     1/0       c1t2d1    DIS
sd  disk03-01  v02-01    disk03    620544    10240     2/0       c1t2d2    ENA
```

The display shows three disks, across which the plex `v02-01` is striped (the lines starting with `sd` represent the stripes). One of the stripe areas is located on a failed disk. This disk is no longer valid, so the plex named `v02-01` has a state of `NODEVICE`. Since this is the only plex of the volume, the volume is invalid and must be removed. If a copy of `v02` exists on the backup media, it can be restored later. Keep a record of the volume name and length of any volume you intend to restore from backup.

5. Use the `vxedit` command to remove the volume, as described earlier.

   A volume that has one mirror on a failed disk can also have other mirrors on disks that are still valid. In this case, the volume does not need to be restored from backup, since the data is still valid on the valid disks.

   The output of the `vxprint -th` command for a volume with one plex on a failed disk (`disk01`) and another plex on a valid disk (`disk02`) is similar to this example:

```
V   NAME        USETYPE   KSTATE    STATE     LENGTH   READPOL     PREFPLEX
PL  NAME        VOLUME    KSTATE    STATE     LENGTH   LAYOUT      NCOL/WID  MODE
SD  NAME        PLEX      DISK      DISKOFFS  LENGTH   [COL/]OFF   DEVICE    MODE

v   v03         fsgen     DISABLED  ACTIVE    30720     SELECT      -
pl  v03-01      v03       DISABLED  ACTIVE    30720     CONCAT      -         RW
sd  disk02-01   v03-01    disk01    620544    30720     0           c1t3d0    ENA
pl  v03-02      v03       DISABLED  NODEVICE  30720     CONCAT      -         RW
sd  disk01-04   v0302     disk03    262144    30720     0           c1t2d2    DIS
```

This volume has two plexes, `v03-01` and `v03-02`. The first plex (`v03-01`) does not use any space on the invalid disk, so it can still be used. The second plex (`v03-02`) uses space on invalid disk `disk01` and has a state of `NODEVICE`. Plex `v03-02` must be removed. However, the volume still has one valid plex containing valid data. If the volume needs to be mirrored, another plex can be added later. Note the name of the volume to create another plex later.

6. To remove an invalid plex, the plex must be dissociated from the volume and then removed. This is done with the `vxplex` command. To remove the plex `v03-02`, use this command:

   ```
   vxplex -o rm dis v03-02
   ```

7. Once all the volumes have been cleaned up, you must clean up the disk configuration as described in the section “Disk Cleanup.”

Disk Cleanup

Once all invalid volumes and plexes have been removed, the disk configuration can be cleaned up. Each disk that was removed, reinstalled, or replaced (as determined from the output of the `vxdisk list` command) must be removed from the configuration.

To remove the disk, use the `vxdg` command. To remove the failed disk `disk01`, use this command:

```
vxdg rmdisk disk01
```

If the `vxdg` command returns an error message, some invalid mirrors exist. Repeat the processes described in the section “Volume Cleanup” until all invalid volumes and mirrors are removed.

Rootability Reconfiguration

Once all the invalid disks have been removed, the replacement or reinstalled disks can be added to Volume Manager control. If the root disk was originally under Volume Manager control or you now wish to put the root disk under Volume Manager control, add this disk first.

To add the root disk to Volume Manager control, use the Volume Manager Support Operations (`vxdiskadm`). Use this command:

```
vxdiskadm
```

From the `vxdiskadm` main menu, select menu item `2` (`Encapsulate a disk`). Follow the instructions and encapsulate the root disk for the system. For more information, see Chapter 3, “Disks and Disk Groups.”.

When the encapsulation is complete, reboot the system to multi-user mode.

Final Reconfiguration

Once the root disk is encapsulated, any other disks that were replaced should be added using `vxdiskadm`. If the disks were reinstalled during the operating system reinstallation, they should be encapsulated; otherwise, they can be added.

Once all the disks have been added to the system, any volumes that were completely removed as part of the configuration cleanup can be recreated and their contents restored from backup. The volume recreation can be done by using `vxassist` or the graphical user interface.

You can recreate the volumes `v01` and `v02by` using this `vxassist` command:

```
vxassist make v01 24000
```

```
vxassist make v02 30720 layout=stripe nstripe=3
```

Once the volumes are created, they can be restored from backup using normal backup / restore procedures.

Any volumes that had plexes removed as part of the volume cleanup can have these mirrors recreated by following the instructions for mirroring a volume with `vxassist` as described in *VERITAS Volume Manager Command Line Interface Administrator's Guide*.

To replace the plex removed from volume `v03` using `vxassist`, use this command:

```
vxassist mirror v03
```

Once you have restored the volumes and plexes lost during reinstallation, the recovery is complete and your system should be configured as it was prior to the failure.

Starting Hot-Relocation

At this point, the Administrator should reboot the system or manually start up hot-relocation (if its service is desired). Either step causes the relocation daemon (and also its `vxnotify` process) to start.

To start hot-relocation, use the following commands.

Start the watch daemon. This sends E-mail to the administrator when any problems are found. To change the address used for sending problem reports, change the argument to `vxrelocd`:

```
nohup /usr/lib/vxvm/bin/vxrelocd root &
```

The following command is also valid:

```
nohup /usr/lib/vxvm/bin/vxrelocd root > /dev/null 2>&1 &
```

The following command can detect whether or not hot-relocation has been started:

```
ps -ef | grep vxrelocd | grep -v grep
```

Plex and Volume States

The following sections describe plex and volume states.

Plex States

Plex states reflect whether or not plexes are complete and consistent copies (mirrors) of the volume contents. Volume Manager utilities automatically maintain the plex state. However, you can modify the state of a plex if changes to the volume with which the plex is associated should not be written to it. For example, if a disk with a particular plex located on it begins to fail, you can temporarily disable that plex.

Note: A plex does not have to be associated with a volume. A plex can be created with the `vxmake plex` command. A plex created with this command can later be attached to a volume.

Volume Manager utilities use plex states to:

- indicate whether volume contents have been initialized to a known state
- determine if a plex contains a valid copy (mirror) of the volume contents
- track whether a plex was in active use at the time of a system failure
- monitor operations on plexes

This section explains plex states in detail and is intended for administrators who wish to have a detailed knowledge of plex states.

Plexes that are associated with a volume have one of the following states:

- EMPTY
- CLEAN
- ACTIVE
- STALE
- OFFLINE
- TEMP
- TEMPRM
- TEMPRMSD
- IOFAIL

A Dirty Region Logging or RAID-5 log plex is a special case, as its state is always set to LOG.

EMPTY Plex State

Volume creation sets all plexes associated with the volume to the EMPTY state to indicate that the plex is not yet initialized.

CLEAN Plex State

A plex is in a CLEAN state when it is known to contain a consistent copy (mirror) of the volume contents and an operation has disabled the volume. As a result, when all plexes of a volume are clean, no action is required to guarantee that the plexes are identical when that volume is started.

ACTIVE Plex State

A plex can be in the ACTIVE state in two ways:

- when the volume is started and the plex fully participates in normal volume I/O (the plex contents change as the contents of the volume change)

- when the volume was stopped as a result of a system crash and the plex was ACTIVE at the moment of the crash

In the latter case, a system failure can leave plex contents in an inconsistent state. When a volume is started, Volume Manager does the recovery action to guarantee that the contents of the plexes marked as ACTIVE are made identical.

Note: On a system running well, ACTIVE should be the most common state you see for any volume plexes.

STALE Plex State

If there is a possibility that a plex does not have the complete and current volume contents, that plex is placed in the STALE state. Also, if an I/O error occurs on a plex, the kernel stops using and updating the contents of that plex, and an operation sets the state of the plex to STALE.

A `vxplex att` operation recovers the contents of a STALE plex from an ACTIVE plex. Atomic copy operations copy the contents of the volume to the STALE plexes. The system administrator can force a plex to the STALE state with a `vxplex det` operation.

OFFLINE Plex State

The `vxmend off` task indefinitely detaches a plex from a volume by setting the plex state to OFFLINE. Although the detached plex maintains its association with the volume, changes to the volume do not update the OFFLINE plex. The plex is not updated until the plex is put online and reattached with the `vxplex att` task. When this occurs, the plex is placed in the STALE state, which causes its contents to be recovered at the next `vxvol start` operation.

TEMP Plex State

Setting a plex to the TEMP state eases some plex operations that cannot occur in a truly atomic fashion. For example, attaching a plex to an enabled volume requires copying volume contents to the plex before it can be considered fully attached.

A utility sets the plex state to TEMP at the start of such an operation and to an appropriate state at the end of the operation. If the system fails for any reason, a TEMP plex state indicates that the operation is incomplete. A later vxvol start dissociates plexes in the TEMP state.

TEMPRM Plex State

A TEMPRM plex state is similar to a TEMP state except that at the completion of the operation, the TEMPRM plex is removed. Some subdisk operations require a temporary plex. Associating a subdisk with a plex, for example, requires updating the subdisk with the volume contents before actually associating the subdisk. This update requires associating the subdisk with a temporary plex, marked TEMPRM, until the operation completes and removes the TEMPRM plex.

If the system fails for any reason, the TEMPRM state indicates that the operation did not complete successfully. A later operation dissociates and removes TEMPRM plexes.

TEMPRMSD Plex State

The TEMPRMSD plex state is used by vxassist when attaching new plexes. If the operation does not complete, the plex and its subdisks are removed.

IOFAIL Plex State

The IOFAIL plex state is associated with persistent state logging. On the detection of a failure of an ACTIVE plex, vxconfigd places that plex in the IOFAIL state so that it is excluded from the recovery selection process at volume start time.

The Plex State Cycle

The changing of plex states is part normal operations. Changes in plex state indicate abnormalities that Volume Manager must normalize. At system startup, volumes are automatically started and the vxvol start task makes all CLEAN plexes ACTIVE. If all goes well until shutdown, the volume-stopping operation marks all ACTIVE plexes CLEAN and the cycle continues. Having all plexes CLEAN at startup (before vxvol start makes them ACTIVE) indicates a normal shutdown and optimizes startup.

Plex Kernel State

The *plex kernel state* indicates the accessibility of the plex. The plex kernel state is monitored in the volume driver and allows a plex to have an offline (DISABLED), maintenance (DETACHED), or online (ENABLED) mode of operation.

The following are plex kernel states:

- DISABLED—The plex cannot be accessed.
- DETACHED—A write to the volume is not reflected to the plex. A read request from the volume is not reflected from the plex. Plex operations and ioctl functions are accepted.
- ENABLED—A write request to the volume is reflected to the plex. A read request from the volume is satisfied from the plex.

Note: No user intervention is required to set these states; they are maintained internally. On a system that is operating properly, all plexes are enabled.

Volume States

There are several volume states, some of which are similar to plex states:

- CLEAN—The volume is not started (kernel state is DISABLED) and its plexes are synchronized.
- ACTIVE—The volume has been started (kernel state is currently ENABLED) or was in use (kernel state was ENABLED) when the machine was rebooted. If the volume is currently ENABLED, the state of its plexes at any moment is not certain (since the volume is in use). If the volume is currently DISABLED, this means that the plexes cannot be guaranteed to be consistent, but are made consistent when the volume is started.
- EMPTY—The volume contents are not initialized. The kernel state is always DISABLED when the volume is EMPTY.
- SYNC—The volume is either in read-writeback recovery mode (kernel state is currently ENABLED) or was in read-writeback mode when the machine was rebooted (kernel state is DISABLED). With read-writeback recovery, plex consistency is recovered by reading data from blocks of one plex and

writing the data to all other writable plexes. If the volume is ENABLED, this means that the plexes are being resynchronized through the read-writeback recovery. If the volume is DISABLED, it means that the plexes were being resynchronized through read-writeback when the machine rebooted and therefore still need to be synchronized.

- NEEDSYNC—The volume requires a resynchronization operation the next time it is started.

The interpretation of these flags during volume startup is modified by the persistent state log for the volume (for example, the DIRTY/CLEAN flag). If the clean flag is set, an ACTIVE volume was not written to by any processes or was not even open at the time of the reboot; therefore, it can be considered CLEAN. The clean flag is always set in any case where the volume is marked CLEAN.

RAID-5 Volume States

RAID-5 volumes have their own set of volume states:

- CLEAN—The volume is not started (kernel state is DISABLED) and its parity is good. The RAID-5 plex stripes are consistent.
- ACTIVE—The volume has been started (kernel state is currently ENABLED) or was in use (kernel state was ENABLED) when the system was rebooted. If the volume is currently ENABLED, the state of its RAID-5 plex at any moment is not certain (since the volume is in use). If the volume is currently DISABLED, parity cannot be guaranteed to be synchronized.
- EMPTY—The volume contents are not initialized. The kernel state is always DISABLED when the volume is EMPTY.
- SYNC—The volume is either undergoing a parity resynchronization (kernel state is currently ENABLED) or was having its parity resynchronized when the machine was rebooted (kernel state is DISABLED).
- NEEDSYNC—The volume requires a parity resynchronization operation the next time it is started.
- REPLAY—The volume is in a transient state as part of a log replay. A log replay occurs when it becomes necessary to use logged parity and data.

Volume Kernel State

The *volume kernel state* indicates the accessibility of the volume. The volume kernel state allows a volume to have an offline (DISABLED), maintenance (DETACHED), or online (ENABLED) mode of operation.

The following are volume kernel states:

- DISABLED—The volume cannot be accessed.
- DETACHED—The volume cannot be read or written, but plex device operations and ioctl functions are accepted.
- ENABLED—The volumes can be read and written.

RAID-5 Volume Recovery

Introduction

This section describes the operation and recovery of RAID-5 volumes. For more information on RAID-5 volumes, refer to Chapter 1, “Understanding Volume Manager,” in the *VERITAS Volume Manager Getting Started Guide*.

RAID-5 Volume Layout

A RAID-5 volume consists of one or more plexes, each of which consists of one or more subdisks. Unlike mirrored volumes, not all plexes in a RAID-5 volume serve to keep a mirror copy of the volume data. A RAID-5 volume can have two types of plexes:

- the *RAID-5 plex* is used to keep both data and parity for the volume
- *Log plexes* keep logs of data written to the volume for faster and more efficient recovery

RAID-5 Plexes

RAID-5 volumes keep both the data and parity information in a single RAID-5 plex. A RAID-5 plex consists of subdisks arranged in columns, similar to the striping model. Figure 1 shows a RAID-5 plex and the output of `vxprint` associated with it:

Note: Your system may use a *device name* that differs from the examples. See Chapter 1, "Understanding Volume Manager," in the *VERITAS Volume Manager Getting Started Guide*, for more information on device names.

Figure 1 vxprint Output for a RAID-5 Plex

```
PL   NAME       VOLUME   KSTATE   STATE     LENGTH   LAYOUT      NCOL/WID    MODE
SD   NAME       PLEX     DISK     DISKOFFS  LENGTH   [COL/]OFF   DEVICE      MODE

pl   rvol-01    rvol     ENABLED  ACTIVE    20480    RAID        3/16        RW
sd   disk00-00  rvol-01  disk00   0         10240    0/0         c1t4d1      ENA
sd   disk01-00  rvol-01  disk01   0         10240    1/0         c1t2d1      ENA
sd   disk02-00  rvol-01  disk02   0         10240    2/0         c1t3d1      ENA
```

The plex line shows that the plex layout is `RAID` and that it has three columns and a stripe unit size of 16 sectors. Each subdisk line shows the column in the plex and offset in the column in which it is located.

RAID-5 Logs

Each RAID-5 volume has one RAID-5 plex where the data and parity are stored. Any other plexes associated with the volume are used to log information about data and parity being written to the volume. These plexes are referred to as *RAID-5 log plexes* or *RAID-5 logs.*

RAID-5 logs can be concatenated or striped plexes, and each RAID-5 log associated with a RAID-5 volume has a complete copy of the logging information for the volume. It is suggested that you have a minimum of two RAID-5 log plexes for each RAID-5 volume. These log plexes should be located on different disks. Having two RAID-5 log plexes for each RAID-5 volume protects against the loss of logging information due to the failure of a single disk.

To support concurrent access to the RAID-5 array, the log should be several times the stripe size of the RAID-5 plex.

You can tell the difference between a RAID-5 log plex and a RAID-5 plex of a RAID-5 volume by examining `vxprint` output. The `STATE` field for a log plex is marked as `LOG`. Figure 2 shows the `vxprint` output for a RAID-5 volume.

Figure 2 vxprint Output for a RAID-5 Volume

```
V      NAME         USETYPE    KSTATE     STATE      LENGTH   READPOL    PREFPLEX
PL     NAME         VOLUME     KSTATE     STATE      LENGTH   LAYOUT     NCOL/WIDMODE
SD     NAME         PLEX       DISK       DISKOFFS   LENGTH   [COL/]OFF  DEVICE MODE

v      r5vol        raid5      ENABLED    ACTIVE     20480    RAID       -
pl     r5vol-01     r5vol      ENABLED    ACTIVE     20480    RAID       3/16   RW
sd     disk00-00    r5vol-01   disk00     0          10240    0/0        c1t4d1 ENA
sd     disk01-00    r5vol-01   disk01     0          10240    1/0        c1t2d1 ENA
sd     disk02-00    r5vol-01   disk02     0          10240    2/0        c1t3d1 ENA
pl     r5vol-l1     r5vol      ENABLED    LOG        1024     CONCAT     -      RW
sd     disk03-01    r5vol-l1   disk00     0          1024     0          c1t3d0 ENA
pl     r5vol-l2     r5vol      ENABLED    LOG        1024     CONCAT     -      RW
sd     disk04-01    r5vol-l2   disk02     0          1024     0          c1t1d1 ENA
```

The RAID-5 volume (`r5vol`) can be identified as a RAID-5 volume by its read policy being `RAID`. It has one RAID-5 plex (`r5vol-01`), similar to the one described earlier. It has two RAID-5 logs in the plexes `r5vol-l1` and `r5vol-l2`. These are identified by the state field being `LOG` and they are associated with a RAID-5 volume and have a layout that is not `RAID`.

Creating RAID-5 Volumes

You can create RAID-5 volumes by using either `vxassist` (recommended) or `vxmake`. Both approaches are described in this section.

A RAID-5 volume contains a RAID-5 plex that consists of two or more subdisks located on two or more physical disks. Only one RAID-5 plex can exist per volume.

`vxassist` and RAID-5 Volumes

You can create a RAID-5 volume by using this `vxassist` command:

`vxassist make` *volume_name length* `layout=raid5`

For example, to create a 10M RAID-5 volume named `volraid`, enter:

```
vxassist make volraid 10m layout=raid5
```

This creates a RAID-5 volume with the default stripe unit size on the default number of disks.

vxmake and RAID-5 Volumes

You can create a RAID-5 volume by using the vxmake command which is similar to the creation of other volumes (see "Creating Volumes"). Also see the vxmake(1M) manual page for details on creating other volumes. The creation of subdisks for use in a RAID-5 volume is the same as creating any subdisk.

Creating a RAID-5 plex for a RAID-5 volume is similar to creating striped plexes, except that the layout attribute is set to raid5. Subdisks can be implicitly associated in the same way as with striped plexes. A four-column RAID-5 plex with a stripe unit size of 32 sectors can be created from four existing subdisks with this command:

```
vxmake plex raidplex layout=raid5 stwidth=32 \
sd=disk00-01,disk01-00,disk02-00,disk03-00
```

Note that because four subdisks are specified with no specification of columns, vxmake assumes a four-column RAID-5 plex and places one subdisk in each column. This is the same behavior for creating striped plexes. If the subdisks are to be created later, use this command to create the plex:

```
vxmake plex raidplex layout=raid5 ncolumn=4 stwidth=32
```

Note: If no subdisks are specified, the ncolumn attribute must be specified. Subdisks can later be filled in the plex by using vxsd assoc (see "Manipulating RAID-5 Subdisks").

To create a three-column RAID-5 plex using six subdisks, use this command:

```
vxmake plex raidplex layout=raid5 stwidth=32 \
sd=disk00-00:0,disk01-00:1,disk02-00:2,disk03-00:0, \
disk04-00:1,disk05-00:2
```

This command stacks subdisks `disk00-00` and `disk03-00` consecutively in column 0, subdisks `disk01-00` and `disk04-00` consecutively in column 1, and subdisks `disk02-00` and `disk05-00` in column 2. Offsets can also be specified to create sparse RAID-5 plexes, as for striped plexes.

Because log plexes are plexes without a RAID-5 layout, they can be created normally.

To create a RAID-5 volume with `vxmake`, specify the usage type to be `RAID-5` by using this command:

```
vxmake -Uraid5 vol raidvol
```

RAID-5 plexes and RAID-5 log plexes can be associated implicitly:

```
vxmake -Uraid5 vol raidvol plex=raidplex,raidlog1,
raidlog2
```

Initializing RAID-5 Volumes

A RAID-5 volume must be initialized if it was created by `vxmake` and has not yet been initialized or if it has been set to an uninitialized state.

A RAID-5 volume can be initialized by `vxvol` with either of these commands:

`vxvol init zero` *volume_name*

or

`vxvol start` *volume_name*

`vxvol init zero` writes zeroes to any RAID-5 log plexes and to the entire length of the volume. It then leaves the volume in the `ACTIVE` state.

`vxvol start` recovers parity by XORing corresponding data stripe units in all other columns. Although it is slower than a `vxvol init zero` operation, `vxvol start` makes the RAID-5 volume immediately available.

Failures and RAID-5 Volumes

Failures are seen in two varieties: *system failures* and *disk failures*. A system failure means that the system has abruptly ceased to operate due to an operating system panic or power failure. Disk failures imply that the data on some number of disks has become unavailable due to a system failure (such as a head crash, electronics failure on disk, or disk controller failure).

System Failures

RAID-5 volumes are designed to remain available with a minimum of disk space overhead, if there are disk failures. However, many forms of RAID-5 can have data loss after a system failure. Data loss occurs because a system failure causes the data and parity in the RAID-5 volume to become unsynchronized. Loss of sync occurs because the status of writes that were outstanding at the time of the failure cannot be determined.

If a loss of sync occurs while a RAID-5 volume is being accessed, the volume is described as having *stale parity*. The parity must then be reconstructed by reading all the nonparity columns within each stripe, recalculating the parity, and writing out the parity stripe unit in the stripe. This must be done for every stripe in the volume, so it can take a long time to complete.

CAUTION! While this resynchronization is going on, any failure of a disk within the array causes the data in the volume to be lost. This only applies to RAID-5 volumes *without* log plexes.

Besides the vulnerability to failure, the resynchronization process can tax the system resources and slow down system operation.

RAID-5 logs reduce the damage that can be caused by system failures, because they maintain a copy of the data being written at the time of the failure. The process of resynchronization consists of reading that data and parity from the logs and writing it to the appropriate areas of the RAID-5 volume. This greatly reduces the amount of time needed for a resynchronization of data and parity. It also means that the volume never becomes truly stale. The data and parity for all stripes in the volume are known at all times, so the failure of a single disk cannot result in the loss of the data within the volume.

Disk Failures

Disk failures can cause the data on a disk to become unavailable. In terms of a RAID-5 volume, this means that a subdisk becomes unavailable.

This can occur due to an uncorrectable I/O error during a write to the disk. The I/O error can cause the subdisk to be detached from the array or a disk being unavailable when the system is booted (for example, from a cabling problem or by having a drive powered down).

When this occurs, the subdisk cannot be used to hold data and is considered *stale* and *detached*. If the underlying disk becomes available or is replaced, the subdisk is still considered stale and is not used.

If an attempt is made to read data contained on a stale subdisk, the data is reconstructed from data on all other stripe units in the stripe. This operation is called a *reconstructing-read*. This is a more expensive operation than simply reading the data and can result in degraded read performance. When a RAID-5 volume has stale subdisks, it is considered to be in *degraded mode*.

A RAID-5 volume in degraded mode can be recognized from the output of `vxprint`, as shown in Figure 3.

Figure 3 vxprint Output for a Degraded RAID-5 Volume

```
V   NAME      USETYPE   KSTATE   STATE     LENGTH   READPOL   PREFPLEX
PL  NAME      VOLUME    KSTATE   STATE     LENGTH   LAYOUT    NCOL/WID    MODE
SD  NAME      PLEX      DISK     DISKOFFS  LENGTH   [COL/]OFF DEVICE      MODE
v   r5vol     RAID-5    ENABLED  DEGRADED  20480    RAID      -
pl  r5vol-01  r5vol     ENABLED  ACTIVE    20480    RAID      3/16        RW
sd  disk00-00 r5vol-01  disk00   0         10240    0/0       c1t4d1
sd  disk01-00 r5vol-01  disk01   0         10240    1/0       c1t2d1      dS
sd  disk02-00 r5vol-01  disk02   0         10240    2/0       c1t3d1      -
pl  r5vol-l1  r5vol     ENABLED  LOG       1024     CONCAT    -           RW
sd  disk03-01 r5vol-l1  disk00   10240     1024     0         c1t3d0      -
pl  r5vol-l2  r5vol     ENABLED  LOG       1024     CONCAT    -           RW
sd  disk04-01 r5vol-l2  disk02   10240     1024     0         c1t1d1      -
```

The volume r5vol is in degraded mode, as shown by the STATE, which is listed as DEGRADED. The failed subdisk is disk01-00, as shown by the flags in the last column—the d indicates that the subdisk is detached and the S indicates that the subdisk contents are stale.

A disk containing a RAID-5 log can have a failure. This has no direct effect on the operation of the volume. However, the loss of all RAID-5 logs on a volume makes the volume vulnerable to a complete failure. In the output of vxprint -ht, failure within a RAID-5 log plex is indicated by the plex state being BADLOG. This is shown in Figure 4, where the RAID-5 log plex r5vol-l1 has failed.

Figure 4 vxprint Output for a RAID-5 Volume with Failed Log Plex

```
V   NAME      USETYPE   KSTATE    STATE     LENGTH   READPOL   PREFPLEX
PL  NAME      VOLUME    KSTATE    STATE     LENGTH   LAYOUT    NCOL/WID  MODE
SD  NAME      PLEX      DISK      DISKOFFS  LENGTH   [COL/]OFF DEVICE    MODE
v   r5vol     RAID-5    ENABLED   ACTIVE    20480    RAID      -
pl  r5vol-01  r5vol     ENABLED   ACTIVE    20480    RAID      3/16      RW
sd  disk00-00 r5vol-01  disk00    0         10240    0/0       c1t4d1    ENA
sd  disk01-00 r5vol-01  disk01    0         10240    1/0       c1t2d1    dS
sd  disk02-00 r5vol-01  disk02    0         10240    2/0       c1t3d1    ENA
pl  r5vol-l1  r5vol     DISABLED  BADLOG    1024     CONCAT    -         RW
sd  disk03-01 r5vol-l1  disk00    10240     1024     0         c1t3d0    ENA
pl  r5vol-l2  r5vol     ENABLED   LOG       1024     CONCAT    -         RW
sd  disk04-01 r5vol-l2  disk02    10240     1024     0         c1t1d1    ENA
```

RAID-5 Recovery

Here are the types of recovery typically needed for RAID-5 volumes:

- parity resynchronization
- stale subdisk recovery
- log plex recovery

These types of recovery are described in the sections that follow. Parity resynchronization and stale subdisk recovery are typically performed when:

- the RAID-5 volume is started

- shortly after the system boots
- by calling the `vxrecover` command

For more information on starting RAID-5 volumes, see “Starting RAID-5 Volumes.”

If hot-relocation is enabled at the time of a disk failure, system administrator intervention is not required unless there is no suitable disk space available for relocation. Hot-relocation is triggered by the failure and the system administrator is notified of the failure by electronic mail.

Hot-relocation automatically attempts to relocate the subdisks of a failing RAID-5 plex. After any relocation takes place, the hot-relocation daemon (`vxrelocd`) also initiate a parity resynchronization.

In the case of a failing RAID-5 log plex, relocation only occurs if the log plex is mirrored; `vxrelocd` then initiates a mirror resynchronization to recreate the RAID-5 log plex. If hot-relocation is disabled at the time of a failure, the system administrator may need to initiate a resynchronization or recovery.

Parity Recovery

In most cases, a RAID-5 array does not have stale parity. Stale parity only occurs after all RAID-5 log plexes for the RAID-5 volume have failed, and then only if there is a system failure. Even if a RAID-5 volume has stale parity, it is usually repaired as part of the volume start process.

If a volume without valid RAID-5 logs is started and the process is killed before the volume is resynchronized, the result is an active volume with stale parity. This can be confirmed by checking the volume state displayed in the output of the `vxprint -ht` command, as shown in Figure 5.

Figure 5 vxprint Output for a Stale RAID-5 Volume

```
V    NAME       USETYPE  KSTATE   STATE     LENGTH   READPOL    PREFPLEX
PL   NAME       VOLUME   KSTATE   STATE     LENGTH   LAYOUT     NCOL/WID  MODE
SD   NAME       PLEX     DISK     DISKOFFS  LENGTH   [COL/]OFF  DEVICE    MODE
v    r5vol      RAID-5   ENABLED  NEEDSYNC  20480    RAID       -
pl   r5vol-01   r5vol    ENABLED  ACTIVE    20480    RAID       3/16      RW
sd   disk00-00  r5vol-01 disk00   0         10240    0/0        c1t4d1    ENA
sd   disk01-00  r5vol-01 disk01   0         10240    1/0        c1t2d1    ENA
sd   disk02-00  r5vol-01 disk02   0         10240    2/0        c1t3d1    ENA
```

This output lists the volume state as NEEDSYNC, indicating that the parity needs to be resynchronized. The state could also have been SYNC, indicating that a synchronization was attempted at start time and that a synchronization process should be doing the synchronization. If no such process exists or if the volume is in the NEEDSYNC state, a synchronization can be manually started by using the `resync` keyword for the `vxvol` command. For example, to resynchronize the RAID-5 volume in Figure 5, use this command:

```
vxvol resync r5vol
```

Parity is regenerated by issuing VOL_R5_RESYNC ioctls to the RAID-5 volume. The resynchronization process starts at the beginning of the RAID-5 volume and resynchronizes a region equal to the number of sectors specified by the `-o` *iosize* option. If `-o` *iosize* is not specified, the default maximum I/O size is used. The `resync` operation then moves onto the next region until the entire length of the RAID-5 volume has been resynchronized.

For larger volumes, parity regeneration can take a long time. It is possible that the system could be shut down or crash before the operation is completed. In case of a system shutdown, the progress of parity regeneration must be kept across reboots. Otherwise, the process has to start all over again.

To avoid the restart process, parity regeneration is *checkpointed*. This means that the offset up to which the parity has been regenerated is saved in the configuration database. The `-o checkpt=`*size* option controls how often the checkpoint is saved. If the option is not specified, it uses the default checkpoint size.

Because saving the checkpoint offset requires a transaction, making the checkpoint size too small can extend the time required to regenerate parity. After a system reboot, a RAID-5 volume that has a checkpoint offset smaller than the volume length starts a parity resynchronization at the checkpoint offset.

Subdisk Recovery

Stale subdisk recovery is usually done at volume start time. However, it is possible that the process doing the recovery crashes, or that the volume started with an option to prevent subdisk recovery. It's also possible that the disk on which the subdisk resides was replaced without recovery operations being performed. In any case, a subdisk recovery can be done by using the `recover` keyword of the `vxvol` command. For example, to recover the stale subdisk in the RAID-5 volume shown in Figure 3, use this command:

```
vxvol recover r5vol disk01-00
```

You can have a RAID-5 volume that has multiple stale subdisks to be caught up all at once. Calling `vxvol recover` with only the name of the volume catches multiple stale subdisks:

```
vxvol recover r5vol
```

Recovering Logs After Failures

RAID-5 log plexes can become detached due to disk failures, as shown in Figure 4. These RAID-5 logs can be reattached by using the `att` keyword for the `vxplex` command. To reattach the failed RAID-5 log plex shown in Figure 4, use this command:

```
vxplex att r5vol r5vol-l1
```

Miscellaneous RAID-5 Operations

Many operations exist for manipulating RAID-5 volumes and associated objects. These operations are usually performed by other commands such as `vxassist` and `vxrecover` as part of larger operations, such as evacuating disks. These command line operations should not be necessary for light usage of the Volume Manager.

Manipulating RAID-5 Logs

RAID-5 logs are represented as plexes of RAID-5 volumes and are manipulated using the `vxplex` command. A RAID-5 log can be added using `vxplex att`:

```
vxplex att r5vol r5log
```

The attach operation can only proceed if the size of the new log is large enough to hold all of the data on the stripe. If the RAID-5 volume already has logs, the new log length is the minimum of each individual log length. This is because the new log is a mirror of the old logs.

If the RAID-5 volume is not enabled, the new log is marked as `BADLOG` and is enabled when the volume is started. However, the contents of the log is ignored.

If the RAID-5 volume is enabled and has other enabled RAID-5 logs, the new log has its contents synchronized with the other logs through `ATOMIC_COPY` ioctls.

If the RAID-5 volume currently has no enabled logs, the new log is zeroed before it is enabled.

Log plexes can be removed from a volume by using the `vxplex dis` command:

```
vxplex dis r5log3
```

If removing the log leaves the volume with less than two valid logs, a warning is printed and the operation is not allowed to continue. The operation must be forced by using the `-o force` option.

Manipulating RAID-5 Subdisks

As with other subdisks, subdisks of the RAID-5 plex of a RAID-5 volume are manipulated using the `vxsd` command. Association is done by using the `assoc` keyword in the same manner as for striped plexes. For example, to add subdisks at the end of each column of the RAID-5 volume in Figure 2, use this command:

```
vxsd assoc r5vol-01 disk10-01:0 disk11-01:1 disk12-01:2
```

If a subdisk is filling a “hole” in the plex (that is, some portion of the volume logical address space is mapped by the subdisk), the subdisk is considered stale. If the RAID-5 volume is enabled, the association operation regenerates the data that belongs on the subdisk by using VOL_R5_RECOVER ioctls. Otherwise, it is marked as stale and is recovered when the volume is started.

Subdisks can be removed from the RAID-5 plex by using vxsd dis:

```
vxsd dis disk10-01
```

CAUTION! If the subdisk maps a portion of the RAID-5 volume address space, this places the volume in DEGRADED mode. In this case, the dis operation prints a warning and must be forced using the -o force option. Also, if removing the subdisk makes the RAID-5 volume unusable, because another subdisk in the same stripe is unusable or missing and the volume is not DISABLED and empty, this operation is not allowed.

Subdisks can be moved to change the disks which a RAID-5 volume occupies by using vxsd mv. For example, if disk03 is to be evacuated and disk22 has enough room by using two portions of its space, you can use this command:

```
vxsd mv disk03-01 disk22-01 disk22-02
```

While this command is similar to that for striped plexes, the actual mechanics of the operation are not similar.

RAID-5 Subdisk Moves

To do RAID-5 subdisk moves, the current subdisk is removed from the RAID-5 plex and replaced by the new subdisks. The new subdisks are marked as stale and then recovered using VOL_R5_RECOVER operations. Recovery is done either by vxsd or (if the volume is not active) when the volume is started. *This means that the RAID-5 volume is degraded for the duration of the operation*.

Another failure in the stripes involved in the move makes the volume unusable. The RAID-5 volume can also become invalid if the parity of the volume becomes stale.

To avoid these situations, the vxsd utility does not allow a subdisk move if:

- a stale subdisk occupies any of the same stripes as the subdisk being moved

- the RAID-5 volume is stopped but was not shut down cleanly (parity is considered stale)
- the RAID-5 volume is active and has no valid log areas

Only the third case can be overridden by using the `-o force` option.

Subdisks of RAID-5 volumes can also be split and joined by using `vxsd split` and `vxsd join`. These operations work the same as for mirrored volumes.

Note: RAID-5 subdisk moves are performed the same as other subdisk moves without the penalty of degraded redundancy.

Starting RAID-5 Volumes

When a RAID-5 volume is started, it can be in one of many states. After a normal system shutdown, the volume should be clean and require no recovery. However, if the volume was not closed, or was not unmounted before a crash, it can require recovery when it is started, before it can be made available. This section describes actions that can be taken under certain conditions.

Under normal conditions, volumes are started automatically after a reboot and any recovery takes place automatically or is done through the `vxrecover` command.

Unstartable RAID-5 Volumes

A RAID-5 volume is unusable if some part of the RAID-5 plex does not map the volume length:

- the RAID-5 plex cannot be sparse in relation to the RAID-5 volume length
- the RAID-5 plex does not map a region where two subdisks have failed within a stripe, either because they are stale or because they are built on a failed disk

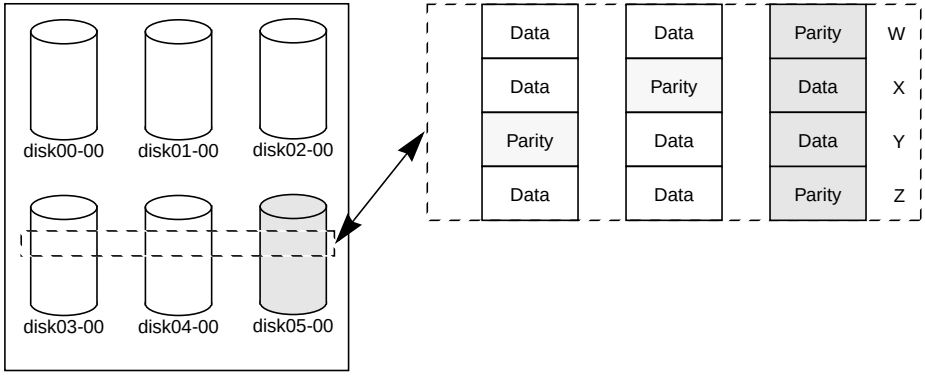
When this occurs, the `vxvol start` command returns this error message:

```
vxvm:vxvol: ERROR: Volume r5vol is not startable; RAID-5 plex does
not map entire volume length.
```

At this point, the contents of the RAID-5 volume are unusable.

Another possible way that a RAID-5 volume can become unstartable is if the parity is stale and a subdisk becomes detached or stale. This occurs because within the stripes that contain the failed subdisk, the parity stripe unit is invalid (because the parity is stale) *and* the stripe unit on the bad subdisk is also invalid. This situation is shown in Figure 6, which shows a RAID-5 volume that has become invalid due to stale parity and a failed subdisk.

Figure 6 Invalid RAID-5 Volume



RAID-5 Plex

This example shows four stripes in the RAID-5 array. All parity is stale and subdisk `disk05-00` has failed. This makes stripes X and Y unusable because two failures have occurred within those stripes.

This qualifies as two failures within a stripe and prevents the use of the volume. In this case, the output display from `vxvol start` is:

```
vxvm:vxvol: ERROR: Volume r5vol is not startable; some subdisks  are
unusable and the parity is stale.
```

This situation can be avoided by *always* using two or more RAID-5 log plexes in RAID-5 volumes. RAID-5 log plexes prevent the parity within the volume from becoming stale which prevents this situation (see “System Failures” for details).

Forcibly Starting RAID-5 Volumes

You can start a volume even if subdisks are marked as stale. For example, if a stopped volume has stale parity and no RAID-5 logs and a disk becomes detached and then reattached.

The subdisk is considered stale even though the data is not out of date (because the volume was in use when the subdisk was unavailable) and the RAID-5 volume is considered invalid. To prevent this case, always have multiple valid RAID-5 logs associated with the array. However, this may not always be possible.

To start a RAID-5 volume with stale subdisks, you can use the `-f` option with the `vxvol start` command. This causes all stale subdisks to be marked as nonstale. Marking takes place before the `start` operation evaluates the validity of the RAID-5 volume and what is needed to start it. Also, you can mark individual subdisks as nonstale by using the command `vxmend fix unstale` *subdisk*.

Recovery When Starting RAID-5 Volumes

Several operations can be necessary to fully restore the contents of a RAID-5 volume and make it usable. Whenever a volume is started, any RAID-5 log plexes are zeroed before the volume is started. This is done to prevent random data from being interpreted as a log entry and corrupting the volume contents. Also, some subdisks may need to be recovered, or the parity may need to be resynchronized (if RAID-5 logs have failed).

The following steps are taken when a RAID-5 volume is started:

1. If the RAID-5 volume was not cleanly shut down, it is checked for valid RAID-5 log plexes.
   - If valid log plexes exist, they are replayed. This is done by placing the volume in the `DETACHED` kernel state and setting the volume state to `REPLAY`, and enabling the RAID-5 log plexes. If the logs can be successfully read and the replay is successful, move on to Step 2.
   - If no valid logs exist, the parity must be resynchronized. Resynchronization is done by placing the volume in the `DETACHED` kernel state and setting the volume state to `SYNC`. Any log plexes are left `DISABLED`

The volume is not made available while the parity is resynchronized because any subdisk failures during this period makes the volume unusable. This can be overridden by using the `-o unsafe` start option with `vxvol`. If any stale subdisks exist, the RAID-5 volume is unusable.

CAUTION! The `-o unsafe` start option is considered dangerous, as it can make the contents of the volume unusable. It is therefore not recommended.

2. Any existing logging plexes are zeroed and enabled. If all logs fail during this process, the start process is aborted.
3. If no stale subdisks exist or those that exist are recoverable, the volume is put in the `ENABLED` kernel state and the volume state is set to `ACTIVE`. The volume is now started.
4. If some subdisks are stale and need recovery, and if valid logs exist, the volume is enabled by placing it in the `ENABLED` kernel state and the volume is available for use during the subdisk recovery. Otherwise, the volume kernel state is set to `DETACHED` and it is not available during subdisk recovery.

   This is done because if the system were to crash or the volume was ungracefully stopped while it was active, the parity becomes stale, making the volume unusable. If this is undesirable, the volume can be started with the `-o unsafe` start option.

CAUTION! The `-o unsafe` start option is considered dangerous, as it can make the contents of the volume unusable. It is therefore not recommended.

5. The volume state is set to `RECOVER` and stale subdisks are restored. As the data on each subdisk becomes valid, the subdisk is marked as no longer stale.

   If any subdisk recovery fails and there are no valid logs, the volume start is aborted because the subdisk remains stale and a system crash makes the RAID-5 volume unusable. This can also be overridden by using the `-o unsafe` start option.

CAUTION! The `-o unsafe` start option is considered dangerous, as it can make the contents of the volume unusable. It is therefore not recommended.

If the volume has valid logs, subdisk recovery failures are noted but do not stop the start procedure.

6. When all subdisks have been recovered, the volume is placed in the ENABLED kernel state and marked as ACTIVE. It is now started.

Changing RAID-5 Volume Attributes

You can change several attributes of RAID-5 volumes. For RAID-5 volumes, the volume length and RAID-5 log length can be changed by using the `vxvol set` command. To change the length of a RAID-5 volume, use this command:

```
vxvol set len=10240 r5vol
```

The length of a volume cannot exceed the mapped region (called the *contiguous length*, or *contiglen*) of the RAID-5 plex. The length cannot be extended so as to make the volume unusable. If the RAID-5 volume is active and the length is being shortened, the operation must be forced by using the `-o force` usage type option. This is done to prevent removal of space from applications using the volume.

The length of the RAID-5 logs can also be changed by using `vxvol set` with this command:

```
vxvol set loglen=2M r5vol
```

Remember that RAID-5 log plexes are only valid if they map the entire length of the RAID-5 volume log length. If increasing the log length makes any of the RAID-5 logs invalid, the operation not allowed. Also, if the volume is not active and is dirty (not shut down cleanly), the log length cannot be changed. This avoids the loss of any of the log contents (if the log length is decreased) or the introduction of random data into the logs (if the log length is being increased).

Writing to RAID-5 Arrays

This section describes the write process for RAID-5 arrays.

Read-Modify-Write

When you write to a RAID-5 array, the following steps can be followed for each stripe involved in the I/O:

1. The data stripe units to be updated with new write data are accessed and read into internal buffers. The parity stripe unit is read into internal buffers.
2. Parity is updated to reflect the contents of the new data region. First, the contents of the old data undergo an exclusive OR (XOR) with the parity (logically removing the old data). The new data is then XORed into the parity (logically adding the new data). The new data and new parity are written to a log.
3. The new parity is written to the parity stripe unit. The new data is written to the data stripe units. All stripe units are written in a single write.

This process is known as a *read-modify-write* cycle, which is the default type of write for RAID-5. If a disk fails, both data and parity stripe units on that disk become unavailable. The disk array is then said to be operating in a *degraded* mode.

The read-modify-write sequence is shown in Figure 7.

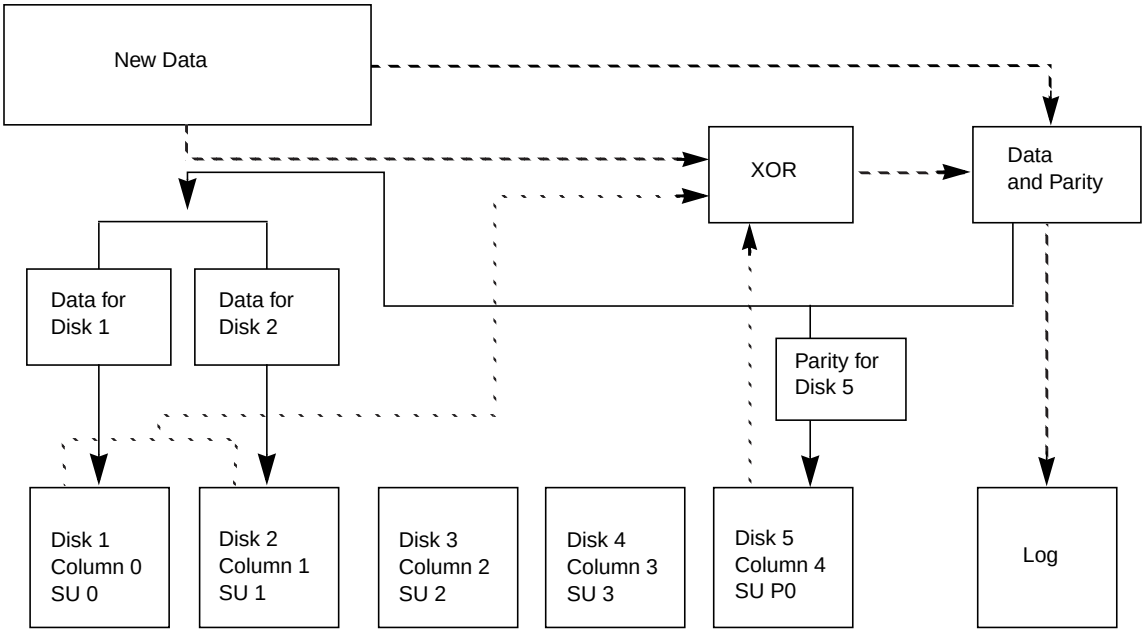
Full-Stripe Writes

When large writes (writes that cover an entire data stripe) are issued, the read-modify-write procedure can be bypassed in favor of a *full-stripe write.* A full-stripe write is faster than a read-modify-write because it does not require the read process to take place. Eliminating the read cycle reduces the I/O time necessary to write to the disk. A full-stripe write procedure consists of the following steps:

1. All the new data stripe units are XORed together, generating a new parity value. The new data and new parity is written to a log.
2. The new parity is written to the parity stripe unit. The new data is written to the data stripe units. The entire stripe is written in a single write.

Figure 8 shows a full-stripe write.

Figure 7 Read-Modify-Write



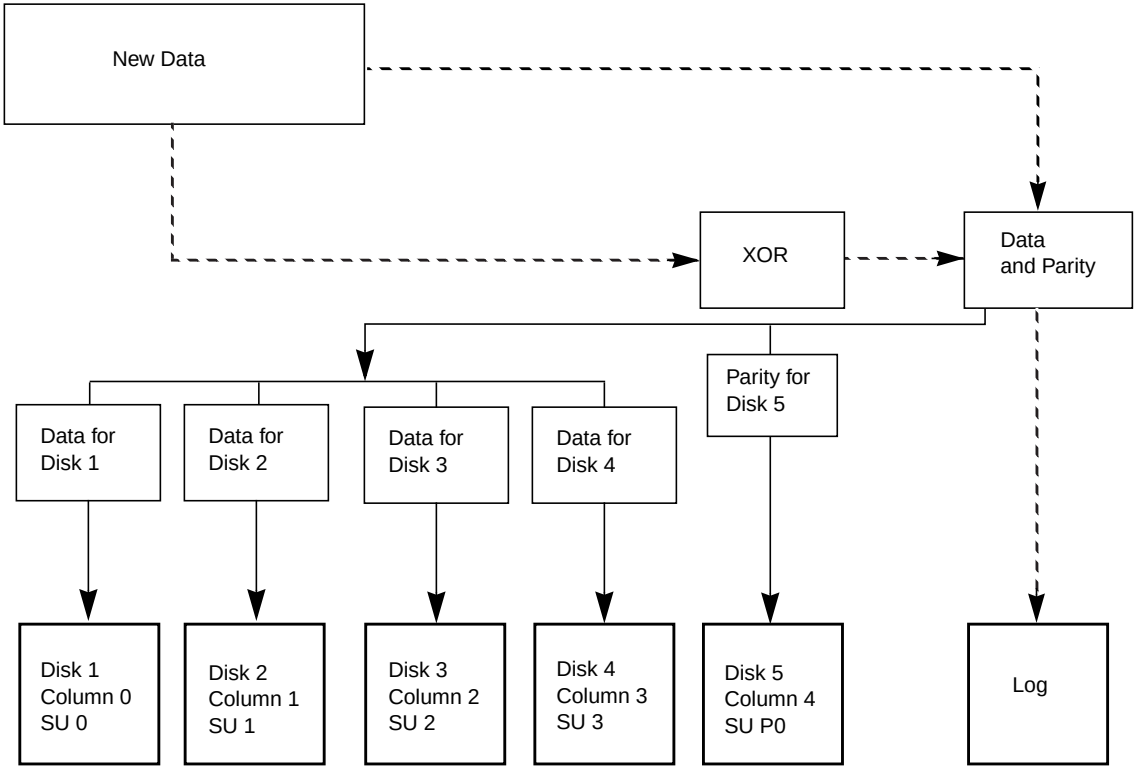
SU = Stripe Unit

= Step 1: Reads data (from parity stripe unit P0 and data stripe units 0 & 1).

= Step 2: Performs XORs between data and parity to calculate new parity. Logs new data and new parity.

= Step 3: Writes new parity (resulting from XOR) to parity stripe unit P0 and new data to data stripe units 0 & 1.

Figure 8 Full-Stripe Write



SU = Stripe Unit

- - - - - = Step 1: Performs XORs between data and parity to calculate new parity. Logs new data and new parity.

——— = Step 2: Writes new parity (resulting from XOR) to parity stripe unit P0 and new data to data stripe units 0, 1, 2, & 3.

Reconstruct-Writes

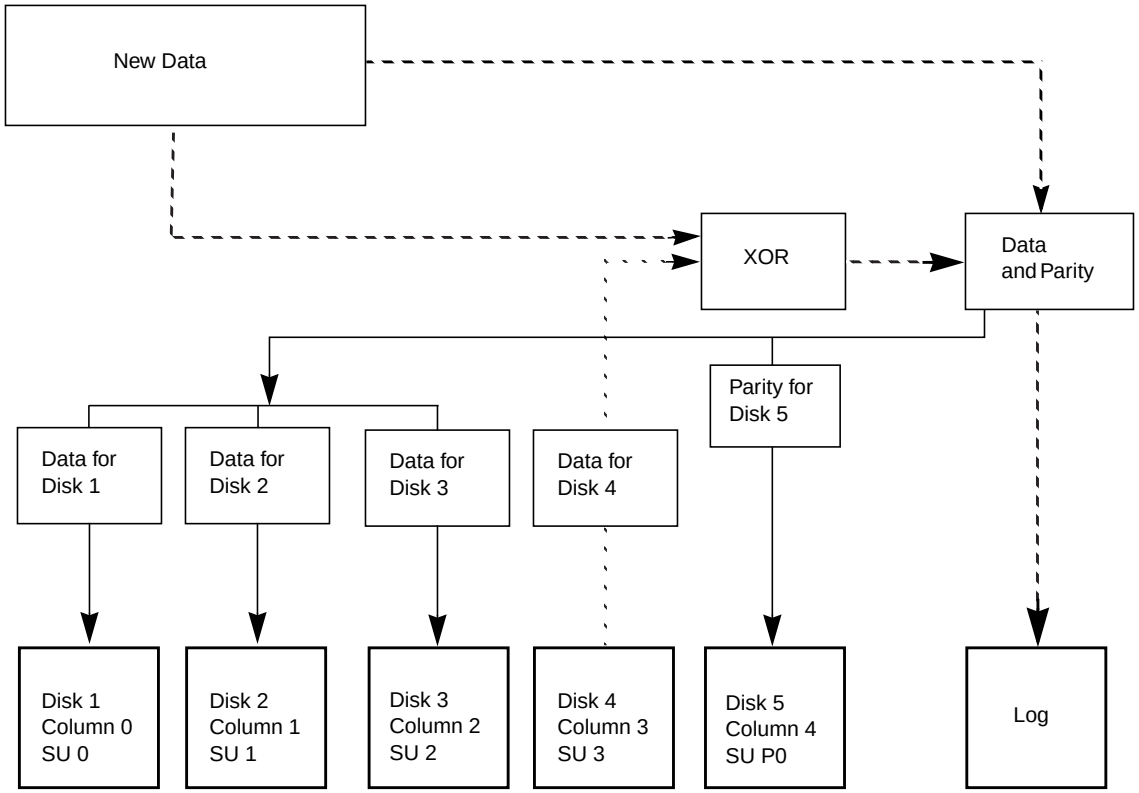
When 50 percent or more of the data disks are undergoing writes in a single I/O, a *reconstruct-write* can be used. A reconstruct-write saves I/O time by XORing. XORing does not require a read of the parity region and only requires a read of the unaffected data. Unaffected data amounts to less than 50 percent of the stripe units in the stripe.

A reconstruct-write procedure consists of the following steps:

1. Unaffected data is read from the unchanged data stripe unit(s).
2. The new data is XORed with the old, unaffected data to generate a new parity stripe unit. The new data and resulting parity are logged.
3. The new parity is written to the parity stripe unit. The new data is written to the data stripe units. All stripe units are written in a single write.

Figure 9 shows a reconstruct-write. A reconstruct-write is preferable to a read-modify-write in this case because it reads only the necessary data disks, rather than reading the disks and the parity disk.

Figure 9 Reconstruct-Write



SU = Stripe Unit

= Step 1: Reads data from unaffected data stripe unit 3.

= Step 2: Performs XORs between old, unaffected data and new data. Logs new data and new parity.

= Step 3: Writes new parity (resulting from XOR) to parity stripe unit P0 and new data to data stripe units 0, 1, & 2.

Disks and Disk Groups

Introduction

This chapter describes the operations for managing disks used by the Volume Manager. It also provides information on disk group operations.

Note: Most Volume Manager commands require superuser or other appropriate privileges.

The following topics are covered in this chapter:

- Standard Disk Devices
- Disk Groups
- Disk and Disk Group Utilities
- Using Disks
  - Initializing and Adding Disks
  - Removing Disks
  - Moving Disks
- Detecting and Replacing Failed Disks
  - Hot-Relocation
  - Detecting Failed Disks
  - Replacing Disks
- Using Disk Groups

- Creating a Disk Group
- Using Disk Groups
- Removing a Disk Group
- Moving Disk Groups Between Systems

- Using Special Devices
  - Using `vxdisk` for Special Encapsulations
  - Using `vxdisk` for RAM Disks

Standard Disk Devices

There are two classes of disk devices that can be used with the Volume Manager: standard devices and special devices. Special devices are described later in this chapter.

The Volume Manager supports up to eight partitions (also called *slices*) on a physical disk. These partitions are named, in order, 0 through 7. Partition 2 is reserved to indicate the entire disk.

Note: On some systems, Volume Manager supports up to sixteen partitions. On these systems, the partitions are named 0 through 15 and partition 0 is reserved to indicate the entire disk.

When a partition is placed under Volume Manager control, a VM disk is assigned to that partition. You can use a symbolic name (the *disk name* or *disk media name*), for example, `disk0,` to refer to a VM disk.

Note: Your system may use a *device name* that differs from the examples. See Chapter 1, “Understanding Volume Manager,” in the *VERITAS Volume Manager Getting Started Guide*, for more information on device names.

A partition is addressed through a physical address (generally referred to as the *device name)* of the form `c#b#t#d#s#`, which has the following elements:

- `c#` —The number of the controller to which the disk drive is attached
- `b#`—The corresponding bus (if used on your system)

- t# and d# —The target ID and device number that constitute the address of the disk drive on the controller
- s# —The partition number on the disk drive

An example of a device name is c0t0d0s2. By convention, s2 refers to the standard partitioning method used by Volume Manager. On some systems, Volume Manager uses s0 as the standard partitioning method. The physical disk is identified to the Volume Manager as c#b#t#d#s# (b# is for systems that use a bus).

For many commands, the suffix s# is optional, though display commands report devices with an s# suffix. The Volume Manager vxdiskadm and vxdiskadd utilities take device names without the s2 (or s0) suffix. For example, to specify the second disk attached to the first controller to vxdiskadd, use the name c0t1d0.

The boot disk (which contains the root file system and is used when booting the system) is often identified to the Volume Manager by the device name c0t0d0.

A VM disk has two regions:

- *private region*—a small area where configuration information is stored. A disk label and configuration records are stored here.
- *public region*—an area that covers the remainder of the disk and is used to store subdisks (and allocate storage space).

Three basic disk types are used by Volume Manager:

- *sliced*—the public and private regions are on different disk partitions.
- *simple*—the public and private regions are on the same disk partition (with the public area following the private area).
- *nopriv*—there is no private region (only a public region for allocating subdisks).

The Volume Manager initializes each new disk with the fewest number of partitions possible (typically two partitions per physical disk). For disk access names ending in s2 (or s0), the default type is sliced.

Disk Groups

Disks are organized by the Volume Manager into disk groups. A *disk group* is a named collection of disks that share a common configuration. Volumes are created within a disk group and are restricted to using disks within that disk group.

A system with the Volume Manager installed has the default disk group, `rootdg`. By default, operations are directed to the `rootdg` disk group. The system administrator can create additional disk groups as necessary. Many systems do not use more than one disk group, unless they have a large number of disks. Disks are not added to disk groups until the disks are needed to create Volume Manager objects. Disks can be initialized, reserved, and added to disk groups later. However, at least one disk (partition) must be added to `rootdg` for you to do the Volume Manager installation procedures.

When a disk is added to a disk group, it is given a name (for example, `disk02`). This name identifies a disk for volume operations: volume creation or mirroring. This name relates directly to the physical disk. If a physical disk is moved to a different target address or to a different controller, the name `disk02` continues to refer to it. Disks can be replaced by first associating a different physical disk with the name of the disk to be replaced and then recovering any volume data that was stored on the original disk (from mirrors or backup copies).

Having large disk groups can cause the private region to fill. In the case of larger disk groups, disks should be set up with larger private areas to log in. A major portion of a private region is space for a disk group configuration database containing records for each Volume Manager object in that disk group. Because each configuration record takes up 256 bytes (or half a block), the number of records that can be created in a disk group is twice the configuration database copy size. The copy size can be obtained from the output of the command `vxdg list` *diskgroupname*.

Disk and Disk Group Utilities

The Volume Manager provides three interfaces that you can use to manage disks:

- the graphical user interface
- a set of command-line utilities
- the `vxdiskadm` menu-based interface

Utilities discussed in this chapter include:

- `vxdiskadm`—this is the Volume Manager Support Operations menu interface. This utility provides a menu of disk operations. Each entry in the main menu leads you through a particular task by providing you with information and prompts. Default answers are provided for many questions so you can easily select common answers.

  This chapter briefly describes the `vxdiskadm` utility, but does not describe its use in detail (refer to the *VERITAS Volume Manager Command Line Interface Administrator's Guide*, for information on how to use `vxdiskadm`).

- `vxdiskadd`—this utility is used to add standard disks to the Volume Manager. `vxdiskadd` leads you through the process of initializing a new disk by displaying information and prompts. See the `vxdiskadd`(1M) manual page for information on how to use `vxdiskadd`.

- `vxdisk`—this is the command-line utility for administering disk devices. `vxdisk` defines special disk devices, initializes information stored on disks (that the Volume Manager uses to identify and manage disks), and performs additional special operations. See the `vxdisk`(1M) manual page for information on how to use `vxdisk`.

- `vxdg`—this is the command-line utility for operating on disk groups. `vxdg` creates new disk groups, adds and removes disks from disk groups, and enables (imports) or disables (deports) access to disk groups. See the `vxdg`(1M) manual page for information on how to use `vxdg`.

Note: The `vxdiskadd` utility and most `vxdiskadm` operations can be used only with standard disk devices.

Using Disks

This section describes the basic disk administration options that are available with the Volume Manager.

Initializing and Adding Disks

There are two levels of initialization for disks in the Volume Manager:

1. Formatting of the disk media itself. This must be done outside of the Volume Manager.
2. Storing identification and configuration information on the disk for use by the Volume Manager. Volume Manager interfaces are provided to step through this level of disk initialization.

A fully initialized disk can be added to a disk group, used to replace a previously failed disk, or to create a new disk group. These topics are discussed later in this chapter.

Formatting the Disk Media

To perform the first initialization phase, use the interactive `format` (on some systems, `diskadd`) command to do a media format of any disk.

Note: SCSI disks are usually preformatted. The `format` (or `diskadd`) command typically is needed only if the format becomes severely damaged.

Volume Manager Disk Installation

You use either the `vxdiskadm` menus or `vxdiskadd to` do the disk initialization phase. This section describes how to use `vxdiskadd`. For information on how to use `vxdiskadm` to initialize a single disk or all disks on a controller, refer to Chapter 5, "Menu Interface Operations," of the *VERITAS Volume Manager Command Line Interface Administrator's Guide.*

You can use `vxdiskadd` to initialize a specific disk. For example, to initialize the second disk on the first controller, use this command:

```
vxdiskadd c0t1d2
```

vxdiskadd examines your disk to determine whether it has been initialized and displays prompts based on what it finds. vxdiskadd checks for disks that can be encapsulated (described in "Using vxdisk for Special Encapsulations"), disks that have been added to the Volume Manager, and for other conditions.

Note: If you are adding an uninitialized disk, warning and error messages are displayed on the console during vxdiskadd. Ignore these messages. These messages should not appear after the disk has been fully initialized; vxdiskadd displays a success message when the initialization completes.

At the following prompt, enter y (or press Return) to continue:

```
Add or initialize disks
Menu: VolumeManager/Disk/AddDisks

 Here is the disk selected.  Output format: [Device_Name]

  c0t1d0

Continue operation? [y,n,q,?] (default: y) y
```

If the disk is uninitialized, or if you choose to reinitialize the disk, you are prompted with this display:

```
  You can choose to add this disk to an existing disk group, a
  new disk group, or leave the disk available for use by future
  add or replacement operations.  To create a new disk group,
  select a disk group name that does not yet exist.  To leave
  the disk available for future use, specify a disk group name
  of "none".

Which disk group [<group>,none,list,q,?] (default: rootdg)
```

To add this disk to the default group rootdg, press Return. To leave the disk free as a replacement disk (not yet added to any disk group), enter none. After this, you are prompted to select a name for the disk in the disk group:

```
Use a default disk name for the disk? [y,n,q,?] (default: y) y
```

Normally, you should accept the default disk name (unless you prefer to enter a special disk name).

At the following prompt, enter n to indicate that this disk should not be used as a hot-relocation spare:

```
Add disk as a spare disk for rootdg? [y,n,q,?] (default: n) n
```

Press Return to continue with the operation after this display:

```
The selected disks will be added to the disk group rootdg with
default disk names.

c0t1d0

Continue with operation? [y,n,q,?] (default: y) y
```

If you are certain that there is no data on this disk that needs to be saved, enter n at the following prompt:

```
The following disk device has a valid VTOC, but does not appear to
have been initialized for the Volume Manager.  If there is data
on the disk that should NOT be destroyed you should encapsulate
the existing disk partitions as volumes instead of adding the disk
as a new disk. Output format: [Device_Name]

c0t1d0

Encapsulate this device? [y,n,q,?] (default: y)
```

When vxdiskadm prompts you to initialize the disk instead, enter y:

```
Instead of encapsulating, initialize? [y,n,q,?] (default: n) y
```

Messages similar to the following should now confirm that disk c1t0d1 is being placed under Volume Manager control. You may also be given the option of performing surface analysis on some systems.

```
Initializing device c0t1d0.

Perform surface analysis (highly recommended)
[y,n,q,?] (default: y) n
Adding disk device c0t1d0 to disk group rootdg with disk
name disk33.
```

Removing Disks

A disk that contains no subdisks can be removed from its disk group with this command:

`vxdg [-g` *groupname*`] rmdisk` *diskname*

where the disk group name is only specified for a disk group other than the default, `rootdg`.

For example, to remove `disk02` from `rootdg`, use this command:

```
vxdg rmdisk disk02
```

If the disk has subdisks on it when you try to remove it, the following error message is displayed:

`vxdg:Disk` *diskname* `is used by one or more subdisks`

Use the `-k` option to `vxdg` to remove device assignment. Using the `-k` option allows you to remove the disk even if subdisks are present. For more information, see the `vxdg`(1M) manual page.

Note: Use of the `-k` option to `vxdg` can result in data loss.

Once the disk has been removed from its disk group, you can (optionally) remove it from Volume Manager control completely, as follows:

`vxdisk rm` *devicename*

For example, to remove `c1t0d0` (or `c1b0t0d0`) from Volume Manager control, use these commands:

```
vxdisk rm c1t0d0s2
```

or on systems with a bus:

```
vxdisk rm c1b0t0d0s0
```

You can remove a disk on which some subdisks are defined. For example, you can consolidate all the volumes onto one disk. If you use `vxdiskadm` to remove a disk, you can choose to move volumes off that disk. To do this, run `vxdiskadm` and select item `3` (`Remove a disk`) from the main menu.

If the disk is used by some subdisks, then this typical message is displayed:

```
The following subdisks currently use part of disk disk02:

     home usrvol

Subdisks must be moved from disk02 before it can be removed.

Move subdisks to other disks? [y,n,q,?] (default: n)
```

If you choose y, then all subdisks are moved off the disk, if possible. Some subdisks may not be movable. The most common reasons why a subdisk may not be movable are:

- There is not enough space on the remaining disks.
- Plexes or striped subdisks cannot be allocated on different disks from existing plexes or striped subdisks in the volume.

If `vxdiskadm` cannot move some subdisks, you may need to remove some plexes from some disks to free more space before proceeding with the disk removal operation. See *VERITAS Volume Manager Command Line Interface Administrator's Guide* for information on how to remove volumes and plexes.

Moving Disks

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 `c0t3d0` (attached with the disk name `disk04`) from disk group `rootdg` and add it to disk group `mktdg`, you use these commands:

```
vxdg rmdisk disk04

vxdg -g mktdg adddisk mktdg02=c0t3d0
```

Note: This procedure does not save the configurations or data on the disks.

You can also move a disk by using `vxdiskadm`. Select item `3` (`Remove a disk`) from the main menu, and then select item `1` (`Add or initialize a disk`).

Detecting and Replacing Failed Disks

This section describes how to detect disk failures and replace failed disks. It begins with the hot-relocation feature, which automatically attempts to restore redundant Volume Manager objects when a failure occurs.

Hot-Relocation

Hot-relocation automatically reacts to I/O failures on redundant (mirrored or RAID-5) Volume Manager objects and restores redundancy and access to those objects. The Volume Manager detects I/O failures on objects and relocates the affected subdisks to disks designated as spare disks and/or free space within the disk group. Volume Manager then reconstructs the objects that existed before the failure and makes them redundant and accessible again. Refer to Chapter 2, "Volume Manager Operations," in the *VERITAS Volume Manager Getting Started Guide*, for a description of hot-relocation.

Note: Hot-relocation is only performed for redundant (mirrored or RAID-5) subdisks on a failed disk. Non-redundant subdisks on a failed disk are not relocated, but the system administrator is notified of their failure.

Hot-relocation is enabled by default and goes into effect without system administrator intervention when a failure occurs. The hot-relocation daemon, `vxrelocd`, detects and reacts to Volume Manager events that signify the following types of failures:

- disk failure—this is normally detected as a result of an I/O failure from a Volume Manager object. Volume Manager attempts to correct the error. If the error cannot be corrected, Volume Manager tries to access configuration information in the private region of the disk. If it cannot access the private region, it considers the disk failed.
- plex failure—this is normally detected as a result of an uncorrectable I/O error in the plex (which affects subdisks within the plex). For mirrored volumes, the plex is detached.
- RAID-5 subdisk failure—this is normally detected as a result of an uncorrectable I/O error. The subdisk is detached.

When such a failure is detected, `vxrelocd` informs the system administrator by electronic mail of the failure and which Volume Manager objects are affected. `vxrelocd` then determines which subdisks (if any) can be relocated. If relocation is possible, `vxrelocd` finds suitable relocation space and relocates the subdisks.

Hot-relocation space is chosen from the disks reserved for hot-relocation in the disk group where the failure occurred. If no spare disks are available or additional space is needed, free space in the same disk group is used. Once the subdisks are relocated, each relocated subdisk is reattached to its plex.

Finally, `vxrelocd` initiates appropriate recovery procedures. For example, recovery includes mirror resynchronization for mirrored volumes or data recovery for RAID-5 volumes. The system administrator is notified of the hot-relocation and recovery actions taken.

If relocation is not possible, the system administrator is notified and no further action is taken. Relocation is not possible in the following cases:

- If subdisks are not redundant (that is, they do not belong to mirrored or RAID-5 volumes), they cannot be relocated.
- If enough space is not available (from spare disks and free space) in the disk group, failing subdisks cannot be relocated.
- If the only available space is on a disk that already contains a mirror of the failing plex, the subdisks in that plex cannot be relocated.
- If the only available space is on a disk that already contains the RAID-5 plex log plex or one of its healthy subdisks, the failing subdisk in the RAID-5 plex cannot be relocated.
- If a mirrored volume has a Dirty Region Logging log subdisk as part of its data plex, subdisks belonging to that plex cannot be relocated.
- If a RAID-5 volume log plex or a mirrored volume DRL log plex fails, a new log plex is created elsewhere (so the log plex is not actually relocated).

You can prepare for hot-relocation by designating one or more disks per disk group as hot-relocation spares. For information on how to designate a disk as a spare, see Chapter 2, “Volume Manager Operations,” in the *VERITAS Volume Manager Getting Started Guide*. If no spares are available at the time of a failure or if there is not enough space on the spares, free space is automatically used.

By designating spare disks, you have control over which space is used for relocation in the event of a failure. If the combined free space and space on spare disks is not sufficient or does not meet the redundancy constraints, the subdisks are not relocated.

After a successful relocation occurs, you need to remove and replace the failed disk (see “Replacing Disks”). Depending on the locations of the relocated subdisks, you can choose to move the relocated subdisks elsewhere after hot-relocation occurs (see “Moving Relocated Subdisks”).

Modifying `vxrelocd`

Hot-relocation is turned on as long as `vxrelocd` is running. You leave hot-relocation turned on so that you can take advantage of this feature if a failure occurs. However, if you choose to disable this feature (you do not want the free space on some of your disks used for relocation), prevent `vxrelocd` from starting at system startup time.

Refer to the *VERITAS Volume Manager Installation Guide* for information on how to disable hot-relocation at system startup. You can stop hot-relocation at any time by killing the `vxrelocd` process (this should not be done while a hot-relocation attempt is in progress).

You can make some minor changes to the way `vxrelocd` behaves by either editing the `vxrelocd` line in the startup file that invokes `vxrelocd` (`/etc/rc2.d/S95vxvm-recover`) or killing the existing `vxrelocd` process and restarting it with different options. After making changes to the way `vxrelocd` is invoked in the startup file, you need to reboot the system so that the changes go into effect. If you choose to kill and restart the daemon instead, make sure that hot-relocation is not in progress when you kill the `vxrelocd` process. You should also restart the daemon immediately so that hot-relocation can take effect if a failure occurs.

You can alter `vxrelocd`’s behavior in the following ways:

- By default, `vxrelocd` sends electronic mail to `root` when failures are detected and relocation actions are performed. You can instruct `vxrelocd` to notify additional users by adding the appropriate user names and invoking `vxrelocd` as shown here:

  `vxrelocd root` *user_name1 user_name2* `&`

- To reduce the impact of recovery on system performance, you can instruct `vxrelocd` to increase the delay between the recovery of each region of the volume, as shown here:

  ```
  vxrelocd -o slow[=IOdelay] root &
  ```

  where the optional *IOdelay* indicates the desired delay (in milliseconds). The default value for the delay is 250 milliseconds. See the `vxrelocd`(1M) manual page for more information.

Displaying Spare Disk Information

You use the command `vxdg spare` to display information about all of the spare disks available for relocation. The output displays this information:

```
GROUP        DISK         DEVICE       TAG          OFFSET    LENGTH    FLAGS
rootdg       disk02       c0t2d0s2     c0t2d0        0         658007    s
```

In this example, `disk02` is the only disk designated as a spare. The `LENGTH` field indicates how much spare space is currently available on this disk for relocation.

The following commands can also be used to display information about disks that are currently designated as spares:

- `vxdisk list`—lists disk information and displays spare disks with a `spare` flag.
- `vxprint`—lists disk and other information and displays spare disks with a `SPARE` flag.

Moving Relocated Subdisks

When hot-relocation occurs, subdisks are relocated to spare disks and/or available free space within the disk group. The new subdisk locations may not provide the same performance or data layout that existed before hot-relocation took place. You can move the relocated subdisks (after hot-relocation is complete) to improve performance.

You can also move the relocated subdisks off the spare disk(s) to keep the spare disk space free for future hot-relocation needs. Another reason for moving subdisks is to recreate the configuration that existed before hot-relocation occurred.

During hot-relocation, one of the electronic mail messages sent to `root` is shown in the following example:

```
To: root
Subject: Volume Manager failures on host teal

Attempting to relocate subdisk disk02-03 from plex home-02.
Dev_offset 0 length 1164 dm_name disk02 da_name c0t5d0s2.
The available plex home-01 will be used to recover the data.
```

This message has information about the subdisk before relocation and can be used to decide where to move the subdisk after relocation.

Here is an example message that shows the new location for the relocated subdisk:

```
To: root
Subject: Attempting VxVM relocation on host teal

Volume home Subdisk disk02-03 relocated to disk05-01,
but not yet recovered.
```

Before you move any relocated subdisks, fix or replace the disk that failed (as described in previous sections). Once this is done, you can move a relocated subdisk back to the original disk. For example, you can move the relocated subdisk `disk05-01` back to `disk02` with this command:

```
vxassist -g rootdg move home !disk05 disk02
```

Note: During subdisk move operations, RAID-5 volumes are not redundant.

Detecting Failed Disks

Note: The Volume Manager hot-relocation feature automatically detects disk failures and notifies the system administrator of the failures by electronic mail. If hot-relocation is disabled or you miss the electronic mail, you can see disk failures through the output of the `vxprint` command or by using the graphical user interface to look at the status of the disks. You can also see driver error messages on the console or in the system messages file.

If a volume has a disk I/O failure (for example, because the disk has an uncorrectable error), the Volume Manager can detach the plex involved in the failure.

If a plex is detached, I/O stops on that plex but continues on the remaining plexes of the volume. If a disk fails completely, the Volume Manager can detach the disk from its disk group.

If a disk is detached, all plexes on the disk are disabled. If there are any unmirrored volumes on a disk when it is detached, those volumes are also disabled.

Partial Disk Failure

If hot-relocation is enabled when a plex or disk is detached by a failure, mail indicating the failed objects is sent to `root`. If a partial disk failure occurs, the mail identifies the failed plexes. For example, if a disk containing mirrored volumes fails, you can receive mail information as shown in this example:

```
To: root
Subject: Volume Manager failures on host teal

Failures have been detected by the VERITAS Volume Manager:

failed plexes:
  home-02
  src-02
```

See “Modifying `vxrelocd`,” for information on how to send the mail to users other than `root`.

You can determine which disk is causing the failures in the above example message by using this command:

```
vxstat -s -ff home-02 src-02
```

Here is a typical output display:

```
                          FAILED
TYP NAME                READS   WRITES
sd disk01-04              0        0
sd disk01-06              0        0
sd disk02-03              1        0
sd disk02-04              1        0
```

This display indicates that the failures are on `disk02` (and that subdisks `disk02-03` and `disk02-04` are affected).

Hot-relocation automatically relocates the affected subdisks and initiates any necessary recovery procedures. However, if relocation is not possible or the hot-relocation feature is disabled, you have to investigate the problem and attempt to recover the plexes. These errors can be caused by cabling failures, so check the cables connecting your disks to your system. If there are obvious problems, correct them and recover the plexes with this command:

```
vxrecover -b home src
```

This command starts recovery of the failed plexes in the background (the command returns before the operation is done). If an error message appears later, or if the plexes become detached again and there are no obvious cabling failures, replace the disk (see “Replacing Disks”).

Complete Disk Failure

If a disk fails completely and hot-relocation is enabled, the mail message lists the disk that failed and all plexes that use the disk. For example, you can receive mail as shown in this example display:

```
To: root
Subject: Volume Manager failures on host teal

Failures have been detected by the VERITAS Volume Manager:

failed disks:
  disk02
```

```
failed plexes:
  home-02
  src-02
  mkting-01

failing disks:
  disk02
```

This message shows that `disk02` was detached by a failure. When a disk is detached, I/O cannot get to that disk. The plexes `home-02`, `src-02`, and `mkting-01` were also detached (probably because of the failure of the disk).

Again, the problem can be a cabling error. If the problem is not a cabling error, replace the disk (see "Replacing Disks").

Replacing Disks

Disks that have failed completely (that have been detached by failure) can be replaced by running `vxdiskadm` and selecting item `5` (`Replace a failed or removed disk`) from the main menu. If there are any initialized but unadded disks, you can select one of those disks as a replacement.

Note: Do not choose the old disk drive as a replacement even though it appears in the selection list. If there are no suitable initialized disks, you can choose to initialize a new disk.

If a disk failure caused a volume to be disabled, the volume must be restored from backup after the disk is replaced. To identify volumes that wholly reside on disks that were disabled by a disk failure, use this command:

```
vxinfo
```

Any volumes that are listed as `Unstartable` must be restored from backup. Here is an example `vxinfo` display:

```
home         fsgen     Started
mkting       fsgen     Unstartable
src          fsgen     Started
standvol     gen       Started  (used on some systems)
rootvol      root      Started
swapvol      swap      Started
```

To restart volume `mkting` so that it can be restored from backup, use this command:

```
vxvol -o bg -f start mkting
```

The `-o bg` option combination resynchronizes plexes as a background task.

If failures are starting to occur on a disk, but the disk has not yet failed completely, replace the disk. This involves two steps:

1. Detach the disk from its disk group.
2. Replace the disk with a new one.

To detach the disk, run `vxdiskadm` and select item `4` (`Remove a disk for replacement`) from the main menu. If there are initialized disks available as replacements, you can specify the disk as part of this operation. Otherwise, you must specify the replacement disk later by selecting item `5` (`Replace a failed or removed disk`) from the main menu.

When you select a disk to remove for replacement, all volumes that can be affected by the operation are displayed. Here is an example display:

```
The following volumes will lose mirrors as a result of this
operation:

     home src

No data on these volumes will be lost.

The following volumes are in use, and will be disabled as a result
of this operation:

     mkting

Any applications using these volumes will fail future accesses.
These volumes will require restoration from backup.
```

```
Are you sure you want do this? [y,n,q,?] (default: n)
```

If any volumes are likely to be disabled, quit from `vxdiskadm` and save the volume. Either back up the volume or move the volume off of the disk. To move the volume `mkting` to a disk other than `disk02`, use this command:

```
vxassist move mkting !disk02
```

After the volume is backed up or moved, run `vxdiskadm` again and continue to remove the disk for replacement.

After the disk has been removed for replacement, a replacement disk can be specified by selecting item `5 (Replace a failed or removed disk)` from `vxdiskadm`'s main menu.

Using Disk Groups

This section describes some disk group administrative options available with the Volume Manager.

Creating a Disk Group

You can create a new disk group on a physical disk using `vxdiskadd`. If you use `vxdiskadd`, specify a new disk group name when prompted for a disk group. Disk groups can also be created by using the operation `vxdg init`. To create a disk group with the `vxdg` utility, use this command:

`vxdg init` *diskgroup diskname=devicename*

For example, to create a disk group named `mktdg` on device `c1t0d0`s2, use the command:

```
vxdg init mktdg mktdg01=c1t0d0
```

The disk device name given to `vxdg` must have been initialized already with `vxdiskadd`. The disk must not have been added to a disk group.

Using Disk Groups

Most Volume Manager commands allow you to specify a disk group using the `–g` option. For example, to create a volume in disk group `mktdg`, you can use the command:

```
vxassist -g mktdg make mktvol 50m
```

The (block) volume device for this volume is:

```
/dev/vx/dsk/mktdg/mktvol
```

The disk group does not have to be specified. Most Volume Manager commands use object names specified on the command line to determine the disk group for the operation. For example, a volume can be created on disk `mktdg01` without specifying the disk group name:

```
vxassist make mktvol 50m mktdg01
```

Many commands work this way as long as two disk groups do not have objects with the same name. For example, the Volume Manager allows you to create volumes named mktvol in both rootdg and in mktdg. If you do this, you must add –g mktdg to any command where you want to manipulate the volume in the mktdg disk group.

Removing a Disk Group

To remove a disk group, unmount and stop any volumes in the disk group and then use this command:

`vxdg deport` *diskgroup*

Deporting a disk group does not actually remove the disk group. It disables use of the disk group by the system. However, disks that are in a deported disk group can be reused, reinitialized, or added to other disk groups.

Moving Disk Groups Between Systems

An important feature of disk groups is that they can be moved between systems. If all disks in a disk group are moved from one system to another, then the disk group can be used by the second system. You do not have to specify the configuration again.

You use the following steps to move a disk group between systems:

1. On the first system, stop all volumes in the disk group, then deport (disable local access to) the disk group with this command:

   `vxdg deport` *diskgroup*

2. Move all the disks to the second system and perform the steps necessary (system-dependent) to make the second system and Volume Manager recognize the new disks.

   This may require a reboot, in which case the `vxconfigd` daemon is restarted and recognizes the new disks. If you do not reboot, use the command `vxdctl enable` to restart `vxconfigd` so Volume Manager also recognizes the disks.

3. Import (enable local access to) the disk group on the second system with this command:

   `vxdg import` *diskgroup*

4. After the disk group is imported, start all volumes in the disk group with this command:

   `vxrecover -g` *diskgroup* `-sb`

You can move disks from a system that has crashed. In this case, you cannot deport the disk group from the first system. When a disk group is created or imported on a system, that system writes a lock on all disks in the disk group.

Note: The purpose of the lock is to ensure that *dual-ported disks* (disks that can be accessed simultaneously by two systems) are not used by both systems at the same time. If two systems try to manage the same disks at the same time, configuration information stored on the disk is corrupted. The disk and its data become unusable.

If you move disks from a system that has crashed or failed to detect the group before the disk is moved, the locks stored on the disks remain and must be cleared. The system returns the this error message:

`vxdg:disk group` *groupname*`: import failed: Disk is in use by another host`

To clear locks on a specific set of devices, use this command:

```
vxdisk clearimport devicename ...
```

It is possible to clear the locks during import by using this command:

```
vxdg -C import diskgroup
```

Note: You must be careful when using the `vxdisk clearimport` or `vxdg -C import` command on systems that do have dual-ported disks. Clearing the locks allows those disks to be accessed at the same time from multiple hosts and could result in corrupted data.

In some cases, you may want to import a disk group when some disks are not available. The `import` operation normally fails if some disks for the disk group cannot be found among the disk drives attached to the system. If the `import` operation fails, one of these error messages is displayed:

```
vxdg: Disk group groupname: import failed: Disk group has no valid
configuration copies
```

This message indicates a fatal error that requires hardware repair or the creation of a new disk group.

```
vxdg: Disk group groupname: import failed: Disk for disk group not
found
```

This message indicates a recoverable error.

If some of the disks in the disk group have failed, you can force the disk group to be imported with the command:

```
vxdg -f import diskgroup
```

Note: You must be careful when using the `-f` option. It can cause the same disk group to be imported twice from different sets of disks, causing the disk group to become inconsistent.

These operations can be performed using `vxdiskadm`. To deport a disk group by using `vxdiskadm`, select menu item `9` (`Remove access to (deport) a disk group`). To import a disk group, select item `8` (`Enable access to`

(`import`) a `disk group`). The `vxdiskadm` import operation checks for host import locks and prompts to see if you want to clear any that are found. It also starts volumes in the disk group.

Renaming Disk Groups

Only one disk group of a given name can exist per system. It is not possible to import or deport a disk group when the target system already has a disk group of the same name. To avoid this problem, the Volume Manager allows you to rename a disk group during import or deport.

For example, because every system running the Volume Manager must have a single `rootdg` default disk group, importing or deporting `rootdg` across systems is a problem. There cannot be two `rootdg` disk groups on the same system. This problem can be avoided by renaming the `rootdg` disk group during the import or deport.

To give a new name to the disk group during import, use this command:

`vxdg [-t] -n` *newdg_name* `import` *diskgroup*

If the `-t` option is included, the import is temporary and does not persist across reboots. In this case, the stored name of the disk group remains unchanged on its original host, but the disk group is known as *newdg_name* to the importing host. If the `-t` option is not used, the name change is permanent.

A disk group can also be renamed on deport with this command:

`vxdg [-h` *hostname*`] -n` *newdg_name* `deport` *diskgroup*

When renaming on deport, you can specify the `-h` *hostname* option to assign a lock to an alternate host. This ensures that the disk group is automatically imported when the alternate host reboots.

You can use the following steps to temporarily move the `rootdg` disk group from one host to another (for example, for repair work on the root volume) and then move the disk group back:

1. On the original host, identify the disk group ID of the `rootdg` disk group to be imported to the other host by using this command:

   ```
   vxdisk -s list
   ```

The output display includes disk group information similar to this example:

```
dgname: rootdg
dgid:   774226267.1025.tweety
```

2. On the importing host, import and rename the `rootdg` disk group with this command:

   `vxdg -tC -n` *newdg_name* `import` *diskgroup*

   where `-t` indicates a temporary import name; `-C` clears import locks; `-n` specifies a temporary name for the `rootdg` to be imported (so it does not conflict with the existing `rootdg`); and *diskgroup* is the disk group ID of the disk group being imported (for example, `774226267.1025.tweety`).

   If a reboot or crash occurs at this point, the temporarily-imported disk group becomes unimported and requires a reimport.

3. After the necessary work has been done on the imported `rootdg`, deport it back to its original host with this command:

   `vxdg -h` *hostname* `deport` *diskgroup*

   where *hostname* is the name of the system whose `rootdg` is being returned (the system name can be confirmed with the command `uname -n`).

   This command removes the imported `rootdg` from the importing host and returns locks to its original host. The original host then autoimports its `rootdg` on the next reboot.

Reserving Minor Numbers for Disk Groups

Disk groups can be moved between systems. You allocate volume device numbers in separate ranges for each disk group. Then all disk groups in a group of machines can be moved without causing device number collisions.

Volume Manager allows you to select a range of minor numbers for a specified disk group. You use this range of numbers during the creation of a volume. This guarantees that each volume has the same minor number across reboots or reconfigurations. If two disk groups have overlapping ranges, an import collision is detected and an avoidance or renumbering mechanism is then needed.

You can set a base volume device minor number for a disk group with this command:

`vxdg init` *diskgroup* `minor=`*base_minor* *devicename*

Volume device numbers for a disk group are chosen to have minor numbers starting at this *base_minor* number. Minor numbers (on most systems) can range up to 131071. A reasonably sized range can be left at the end for temporary device number remappings (in the event that two device numbers still conflict).

If you do not specify a `minor` operand on the `vxdg init` command line, the Volume Manager chooses a random number. The number chosen is at least 1000 or is a multiple of 1000, and yields a usable range of 1000 device numbers. The chosen default number does not overlap within a range of 1000 of any currently imported disk groups. It also does not overlap any currently allocated volume device numbers.

Note: The default policy ensures that a small number of disk groups can be merged successfully between a set of machines. However, where disk groups are merged automatically using fail-over mechanisms, you should select ranges that avoid overlap.

For further information on minor number reservation, refer to the `vxdg`(1M) manual page.

Using Special Devices

This section describes special devices the Volume Manager uses to do administrative tasks.

Using `vxdisk` for Special Encapsulations

Encapsulation is a process that converts existing partitions on a specified disk to volumes. If any partitions contain file systems, `/etc/vfstab` entries are modified so the file systems are mounted on volumes instead.

Disk encapsulation requires that free space be available on the disk for storing Volume Manager identification and configuration information. This free space cannot be included in any other partitions. (See the *VERITAS Volume Manager Installation Guide* and the vxencap(1M) manual page for more information.)

You can encapsulate a disk that does not have space available for the Volume Manager private region partition. The vxdisk utility encapsulates disks that do not have available space. This is done by using special types of disk devices, called nopriv devices, that do not have private regions.

To use vxdisk, create a partition on the disk device that maps all parts of the disk that you want to access. Then add the partition device for that partition with this command:

```
vxdisk define partition-device type=nopriv
```

Where *partition-device* is the basename of the device in the /dev/dsk directory. For example, to use partition 3 of disk device c0t4d0, use the following command:

```
vxdisk define c0t4d0s3 type=nopriv
```

You create volumes for other partitions on the disk drive by:

- adding the device to a disk group
- determine where those partitions reside within the encapsulation partition
- use vxassist to create a volume with that offset and length

vxassist, by default, reinitializes the data area of a volume that it creates. If there is data to be preserved on the partition, do not use vxassist. Create the volume with vxmake and start the volume with vxvol init active.

A drawback with using the nopriv devices is that the Volume Manager cannot track changes in the address or controller of the disk. Normally, the Volume Manager uses identifying information stored in the private region on the physical disk to track changes in the location of a physical disk. Because nopriv devices do not have private regions and have no identifying information stored on the physical disk, tracking cannot occur.

The best use of special encapsulation partition devices is to encapsulate a disk so that the Volume Manager can be used to move space off the disk. When space is made available on the disk, the special partition device can be removed and the disk can then be encapsulated as a standard disk device.

A disk group cannot be formed entirely from `nopriv` devices. This is because `nopriv` devices do not provide space for storing disk group configuration information. Configuration information must be stored on at least one disk in the disk group.

Using `vxdisk` for RAM Disks

Note: This section only applies to systems with RAM disks.

Some systems support creation of RAM disks. A RAM disk is a device made from system RAM that looks like a small disk device. Often, the contents of a RAM disk are erased when the system is rebooted. RAM disks that are erased on reboot prevent the Volume Manager from identifying physical disks. This is because information stored on the physical disks (now erased on reboot) is used to identify the disk.

`nopriv` devices have a special feature to support RAM disks: a *volatile* option which indicates to the Volume Manager that the device contents do not survive reboots. Volatile devices receive special treatment on system startup. If a volume is mirrored, plexes made from volatile devices are always recovered by copying data from nonvolatile plexes.

To use a RAM disk, a device node must be created for the disk in the `/dev/dsk` and `/dev/rdsk` directories (for example, `/dev/dsk/ramd0` and `/dev/rdsk/ramd0`). To define the RAM disk device to the Volume Manager, use this command:

```
vxdisk define ramd0 type=nopriv volatile
```

Normally, the Volume Manager does not start volumes that are formed entirely from plexes with volatile subdisks. That is because there is no plex that is guaranteed to contain the most recent volume contents.

Some RAM disks are used in situations where all volume contents are recreated after reboot. In these situations, you can force volumes formed from RAM disks to be started at reboot by using this command:

```
vxvol set startopts=norecov volume_name
```

This option can be used only with gen-type volumes. See vxvol(1M) for more information on the vxvol set operation and the norecov option.

Using vxdisk To Display Multipathing Information

Note: This section applies only to systems with the Dynamic Multipathing (DMP) feature.

In Volume Manager, physical disks connected to the system are represented as metadevices with one or more physical access paths. The access paths depend on whether the disk is a single disk or part of a multiported disk array connected to the system. You use the vxdisk utility to display the paths of a metadevice, and to display the status of each path (for example, enabled or disabled). For example, to display details on a disk01, enter:

```
vxdisk list disk01
```

The Volume Manager returns this display:

```
Device    c2t0d0s2
devicetag c2t0d0
type      sliced
hostid    aparajita
disk      name=disk01 id=861086917.1052.aparajita
group     name=rootdg id=861086912.1025.aparajita
flags     online ready autoconfig autoimport imported
pubpaths  block=/dev/vx/dmp/c2t0d0s4 char=/dev/vx/rdmp/c2t0d0s4
privpaths block=/dev/vx/dmp/c2t0d0s3 char=/dev/vx/rdmp/c2t0d0s3
version   2.1
iosize    min=512 (bytes) max=2048 (blocks)
public    slice=4 offset=0 len=1043840
private   slice=3 offset=1 len=1119
update    time=861801175 seqno=0.48
headers   0 248
configs   count=1 len=795
logs      count=1 len=120
Defined regions
config    priv 000017-000247[000231]:copy=01 offset=000000 enabled
config    priv 000249-000812[000564]:copy=01 offset=000231 enabled
log       priv 000813-000932[000120]:copy=01 offset=000000 enabled
```

```
Multipathing information:
numpaths:       2
c2t0d0s2        active
c1t0d0s2        failed
```

The display shows two paths to a physical device represented by the metadevice `c2t0d0s2`. The path `c2t0d0s2` is active and the other path `c1t0d0s2` is in a failed state.

VxVM Performance Monitoring

Introduction

Logical volume management is a tool that can improve overall system performance. This chapter has performance management and configuration guidelines that can help you to benefit from the advantages provided by Volume Manager. This chapter provides information to establish performance priorities and describes ways to obtain and use appropriate data.

The following topics are covered in this chapter:

- Performance Guidelines
  - Data Assignment
  - Striping
  - Mirroring
  - Mirroring and Striping
  - Striping and Mirroring
  - Using RAID-5
- Performance Monitoring
  - Performance Priorities
  - Getting Performance Data
  - Using Performance Data
- Tuning the Volume Manager

- General Tuning Guidelines
- Tunables
- Tuning for Large Systems

Performance Guidelines

This section has information on Volume Manager features. Volume Manager provides flexibility in configuring storage to improve system performance. Two basic strategies can be used to optimize performance:

- assign data to physical drives to evenly balance the I/O load among the available disk drives
- identify the most frequently accessed data and increase access bandwidth to that data by using striping and mirroring

Volume Manager also provides data redundancy (through mirroring and RAID-5) that allows continuous access to data in the event of disk failure.

Data Assignment

When deciding where to locate file systems, a system administrator typically attempts to balance I/O load among available disk drives. The effectiveness of this approach can be limited by difficulty in anticipating future usage patterns, as well as an inability to split file systems across drives. For example, if a single file system receives most of the disk accesses, placing that file system on another drive moves the bottleneck to another drive.

Since Volume Manager can split volumes across multiple drives, a finer level of granularity in data placement can be achieved. After measuring actual access patterns, the system administrator can adjust file system placement decisions. Volumes can be reconfigured online after performance patterns have been established or have changed, without adversely impacting volume availability.

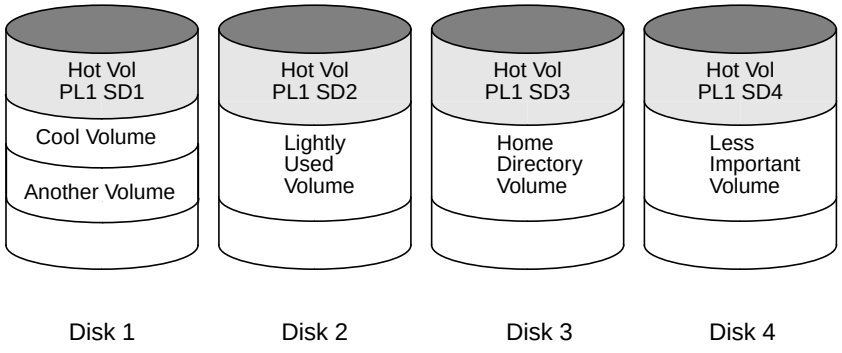
Striping

Striping is a way of “slicing” data and storing it across multiple devices in to improve access performance. Striping provides increased access bandwidth for a plex. Striped plexes improve access performance for both read and write operations.

If the most heavily-accessed volumes (containing file systems or databases) can be identified, then performance benefits can be realized. By striping this “high traffic” data across portions of multiple disks, you can increase access bandwidth to this data.

Figure 10 is an example of a single volume (`Hot Vol`) that has been identified as being a data access bottleneck. This volume is striped across four disks, leaving the remainder of those four disks free for use by less-heavily used volumes.

Figure 10 Use of Striping for Optimal Data Access



Mirroring

Mirroring is a technique for storing multiple copies of data on a system. When properly applied, mirroring can be used to provide continuous data availability by protecting against data loss due to physical media failure. The use of mirroring improves the chance of data recovery in the event of a system crash or disk failure.

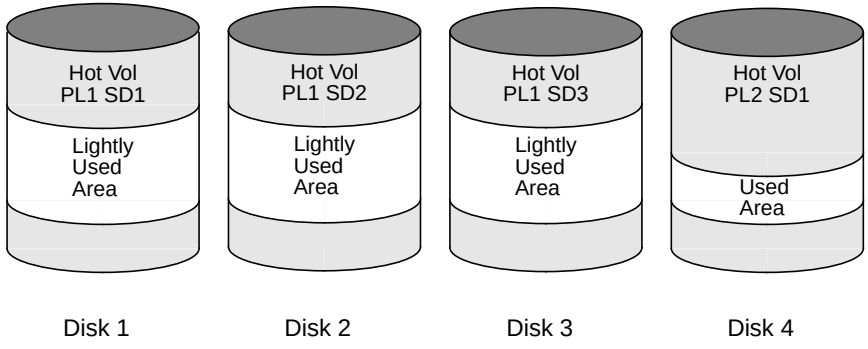
In some cases, mirroring can also be used to improve system performance. Mirroring heavily-accessed data not only protects the data from loss due to disk failure, but can also improve I/O performance. Unlike striping however, performance gained through the use of mirroring depends on the read/write ratio of the disk accesses. If the system workload is primarily write-intensive (for example, greater than 30 percent writes), then mirroring can result in somewhat reduced performance.

To provide optimal performance for different types of mirrored volumes, Volume Manager supports these read policies:

- The *round-robin* read policy (`round`), where read requests to the volume are satisfied in a round-robin manner from all plexes in the volume.
- The *preferred-plex* read policy (`prefer`), where read requests are satisfied from one specific plex (presumably the plex with the highest performance), unless that plex has failed, in which case another plex is accessed.
- The default read policy (`select`), which selects the appropriate read policy for the configuration. For example, selecting preferred-plex when there is only one striped plex associated with the volume and round-robin in most other cases.

In the configuration example shown in Figure 11, the read policy of the volume labeled `Hot Vol` should be set to `prefer` for the striped plex labeled `PL1`. In this way, reads going to `PL1` distribute the load across a number of otherwise lightly-used disks, as opposed to a single disk.

Figure 11 Use of Mirroring and Striping for Improved Performance



To improve performance for read-intensive workloads, up to 32 plexes can be attached to the same volume. However, this scenario results in a decrease of effective disk space utilization. Performance can also be improved by striping across half of the available disks to form one plex and across the other half to

form another plex. When feasible, this is usually the best way to configure the Volume Manager on a set of disks for best performance with reasonable reliability.

Mirroring and Striping

When used together, mirroring and striping provide the advantages of both spreading data across multiple disks and providing redundancy of data.

Mirroring and striping can be used together to achieve a significant improvement in performance when there are multiple I/O streams. Striping can improve serial access when I/O exactly fits across all stripe units in one stripe. Better throughput is achieved because parallel I/O streams can operate concurrently on separate devices.

Since mirroring is most often used to protect against loss of data due to disk failures, it may sometimes be necessary to use mirroring for write-intensive workloads. In these instances, mirroring can be combined with striping to deliver both high availability and performance.

Striping and Mirroring

When used together, striping and mirroring provide the advantages of both spreading data across multiple disks and providing redundancy of data.

Striping and mirroring can be used together to achieve a significant improvement in performance when there are multiple I/O streams. Striping can improve serial access when I/O exactly fits across all stripe units in one stripe. Better throughput is achieved because parallel I/O streams can operate concurrently on separate devices.

Since mirroring is most often used to protect against loss of data due to disk failures, it may sometimes be necessary to use mirroring for write-intensive workloads. In these instances, mirroring can be combined with striping to deliver both high availability and performance. See “Layered Volumes” in the *VERITAS Volume Manager Getting Started Guide*.

Using RAID-5

RAID-5 offers many of the advantages of using mirroring and striping together, but RAID-5 requires less disk space. RAID-5 read performance is similar to that of striping and RAID-5 parity offers redundancy similar to mirroring. Disadvantages of RAID-5 include relatively slow writes.

Note: RAID-5 is not generally seen as a performance improvement mechanism except in cases of high read-to-write ratios shown in the access patterns of the application.

Performance Monitoring

There are two sets of priorities for a system administrator. One set is *physical*, concerned with the hardware. The other set is *logical*, concerned with managing the software and its operations.

Performance Priorities

The physical performance characteristics address the balance of the I/O on each drive and the concentration of the I/O within a drive to minimize seek time. Based on monitored results, you can move subdisk locations to balance the disks.

The logical priorities involve software operations and how they are managed. Based on monitoring, certain volumes can be mirrored or striped to improve their performance. Overall throughput can be sacrificed to improve the performance of critical volumes. Only the system administrator can decide what is important on a system and what tradeoffs to make.

Best performance can generally be achieved by striping and mirroring all volumes across a reasonable number of disks and mirroring between controllers when possible. This tends to even out the load between all disks. However, this usually makes the Volume Manager more difficult to administer. If you have a large number of disks (hundreds or thousands), you can place disks in groups of 10 (using disk groups), where each group is used to stripe and mirror s set of volumes. This still provides good performance and eases the task of administration.

Getting Performance Data

Volume Manager provides two types of performance information: I/O statistics and I/O traces. Each type can help in performance monitoring. I/O statistics are retrieved using the `vxstat` utility, and I/O tracing can be retrieved using the `vxtrace` utility. A brief discussion of each of these utilities is included in this chapter.

Obtaining I/O Statistics (`vxstat`)

The `vxstat` utility accesses activity information on volumes, plexes, subdisks, and disks under Volume Manager control. `vxstat` reports statistics that reflect the activity levels of Volume Manager objects since boot time. Statistics for a specific Volume Manager object, or all objects, can be displayed at one time. A disk group can also be specified, in which case statistics for objects in that disk group only are displayed. If no disk group is specified, `rootdg` is assumed.

The amount of information displayed depends on what options are specified to `vxstat`. For detailed information on available options, refer to the `vxstat`(1M) manual page.

Volume Manager records the following I/O statistics:

- a count of operations
- the number of blocks transferred (one operation can involve more than one block)
- the average operation time (which reflects the total time through the Volume Manager interface and is not suitable for comparison against other statistics programs)

Volume Manager records the preceding I/O statistics for logical I/Os. The statistics include reads, writes, atomic copies, verified reads, verified writes, plex reads, and plex writes for each volume. As a result, one write to a two-plex volume results in at least five operations: one for each plex, one for each subdisk, and one for the volume. Also, one read that spans two subdisks shows at least four reads—one read for each subdisk, one for the plex, and one for the volume.

Volume Manager also maintains other statistical data. For each plex, read failures and write failures are maintained. For volumes, corrected read failures and write failures accompany the read failures and write failures.

`vxstat` can also reset the statistics information to zero. Use the command `vxstat -r` to clear all statistics. This can be done for all objects or for only those objects that are specified. Resetting just prior to an operation makes it possible to measure the impact of that particular operation.

Here is an example of `vxstat` output:

```
                    OPERATIONS          BLOCKS         AVG TIME(ms)
TYP  NAME         READ    WRITE      READ     WRITE     READ      WRITE
vol  blop           0        0         0         0      0.0        0.0
vol  foobarvol      0        0         0         0      0.0        0.0
vol  rootvol    73017   181735    718528   1114227     26.8       27.9
vol  swapvol    13197    20252    105569    162009     25.8      397.0
vol  testvol        0        0         0         0      0.0        0.0
```

Additional volume statistics are available for RAID-5 configurations. See the `vxstat`(1M) manual page for more information.

Tracing I/O (`vxtrace`)

The `vxtrace` command traces operations on volumes. `vxtrace` either prints kernel I/O errors or I/O trace records to the standard output or writes the records to a file in binary format. Tracing can be applied to specific kernel I/O object types or to specified objects or devices. For additional information, refer to the `vxtrace`(1M) manual page.

Using Performance Data

Once performance data has been gathered, it can be used to determine an optimum system configuration for efficient use of system resources. The following sections provide an overview of how this data can be used.

Using I/O Statistics

Examination of the I/O statistics can suggest reconfiguration. There are two primary statistics: volume I/O activity and disk I/O activity.

Before obtaining statistics, clear (reset) all existing statistics. Use the command `vxstat -r` to clear all statistics. Clearing statistics eliminates any differences between volumes or disks due to volumes being created, and also removes statistics from booting (which are not normally of interest).

After clearing the statistics, allow the system to run during typical system activity. To measure the effect of a particular application or workload, the system should be run on that particular application or workload. When monitoring a system that is used for multiple purposes, try not to exercise any one application more than it would be exercised normally. When monitoring a time-sharing system with many users, try to let statistics accumulate during normal use for several hours during the day.

To display volume statistics, use the command `vxstat` with no arguments. Here is a typical display:

```
                    OPERATIONS          BLOCKS          AVG TIME(ms)
TYP  NAME       READ   WRITE       READ     WRITE      READ    WRITE
vol  archive     865     807       5722      3809      32.5     24.0
vol  home       2980    5287       6504     10550      37.7    221.1
vol  local     49477   49230     507892    204975      28.5     33.5
vol  rootvol  102906  342664    1085520   1962946      28.1     25.6
vol  src       79174   23603     425472    139302      22.4     30.9
vol  swapvol   22751   32364     182001    258905      25.3    323.2
```

This output helps to identify volumes with an unusually large number of operations or excessive read or write times.

To display disk statistics, use the command `vxstat -d`. Here is a typical display of disk statistics:

```
                    OPERATIONS          BLOCKS          AVG TIME(ms)
TYP  NAME       READ    WRITE       READ    WRITE      READ    WRITE
dm   disk01    40473   174045     455898   951379      29.5     35.4
dm   disk02    32668    16873     470337   351351      35.2    102.9
dm   disk03    55249    60043     780779   731979      35.3     61.2
dm   disk04    11909    13745     114508   128605      25.0    30.7
```

You may want to move volumes from one disk to another. To move the volume `archive` onto another disk, first identify which disk(s) it is on using this command:

```
vxprint -tvh archive
```

Here is an example display:

```
V  NAME         USETYPE     KSTATE   STATE     LENGTH READPOL   PREFPLEX
PL NAME         VOLUME      KSTATE   STATE     LENGTH LAYOUT    NCOL/WDTH  MODE
SD NAME         PLEX        PLOFFS   DISKOFFS  LENGTH [COL/]OFF FLAGS
v  archive      fsgen       ENABLED  ACTIVE    204800 SELECT    -
pl archive-01   archive     ENABLED  ACTIVE    204800 CONCAT    -          RW
sd disk03-03    archive-01  0        409600    204800 0         c1t2d0s0
```

Note: Your system may use a *device name* that differs from the examples. See Chapter 1, "Understanding Volume Manager," in the *VERITAS Volume Manager Getting Started Guide*, for more information on device names.

The associated subdisks list indicates that the `archive` volume is on disk `disk03`. To move the volume off `disk03`, use the command:

```
vxassist move archive !disk03 dest_disk
```

where *dest_disk* is the disk to which you want to move the volume. It is not necessary to specify a *dest_disk*. If you do not, the volume is moved to an available disk with enough space to contain the volume.

For example, use this command to move the volume from `disk03` to `disk04`:

```
vxassist move archive !disk03 disk04
```

This command indicates that the volume is to be reorganized so that no part remains on `disk03`.

Note: The graphical user interface provides an easy way to move pieces of volumes between disks and may be preferable to using the command-line.

If there are two busy volumes (other than the root volume), move them so that each is on a different disk.

If there is one volume that is particularly busy (especially if it has unusually large average read or write times), stripe the volume (or split the volume into multiple pieces, with each piece on a different disk). If done online, converting a volume to use striping requires sufficient free space to store an extra copy of the volume. If sufficient free space is not available, a backup copy can be made instead. To convert to striping, create a striped plex of the volume and then remove the old plex. For example, to stripe the volume `archive` across disks `disk02`, `disk03`, and `disk04`, use this command:

```
vxassist mirror archive layout=stripe disk02 disk03 disk04

vxplex -o rm dis archive-01
```

After reorganizing any particularly busy volumes, check the disk statistics. If some volumes have been reorganized, clear statistics first and then accumulate statistics for a reasonable period of time.

If some disks appear to be excessively busy (or have particularly long read or write times), you may want to reconfigure some volumes. If there are two relatively busy volumes on a disk, move them closer together to reduce seek times on the disk. If there are too many relatively busy volumes on one disk, move them to a disk that is less busy.

Use I/O tracing (or subdisk statistics) to determine whether volumes have excessive activity in particular regions of the volume. If the active regions can be identified, split the subdisks in the volume and move those regions to a less busy disk.

CAUTION! Striping a volume, or splitting a volume across multiple disks, increases the chance that a disk failure results in failure of that volume. For example, if five volumes are striped across the same five disks, then failure of any one of the five disks requires that all five volumes be restored from a backup. If each volume were on a separate disk, only one volume would need to be restored. Use mirroring or RAID-5 to reduce the chance that a single disk failure results in failure of a large number of volumes.

Note that file systems and databases typically shift their use of allocated space over time, so this position-specific information on a volume is often not useful. For databases, it may be possible to identify the space used by a particularly busy index or table. If these can be identified, they are reasonable candidates for moving to non-busy disks.

Examining the ratio of reads and writes helps to identify volumes that can be mirrored to improve their performance. If the read-to-write ratio is high, mirroring could increase performance as well as reliability. The ratio of reads to writes where mirroring can improve performance depends greatly on the disks, the disk controller, whether multiple controllers can be used, and the speed of the system bus. If a particularly busy volume has a high ratio of reads to writes, it is likely that mirroring can significantly improve performance of that volume.

Using I/O Tracing

I/O statistics provide the data for basic performance analysis; I/O traces serve for more detailed analysis. With an I/O trace, focus is narrowed to obtain an event trace for a specific workload. This helps to explicitly identify the location and size of a hot spot, as well as which application is causing it.

Using data from I/O traces, real work loads on disks can be simulated and the results traced. By using these statistics, the system administrator can anticipate system limitations and plan for additional resources.

Tuning the Volume Manager

This section describes the mechanisms for controlling the resources used by the Volume Manager. Adjustments may be required for some of the tunable values to obtain best performance (depending on the type of system resources available).

General Tuning Guidelines

The Volume Manager is tuned for most configurations ranging from small systems to larger servers. In cases where tuning can be used to increase performance on larger systems at the expense of a valuable resource (such as memory), the Volume Manager is generally tuned to run on the smallest supported configuration. These tuning changes should be performed with care as they may adversely affect overall system performance or may even leave the Volume Manager unusable.

Various mechanisms exist for tuning the Volume Manager. On some systems, several parameters can be tuned using the global tunable file `/etc/system`. Other values can only be tuned using the command line interface to the Volume Manager.

Tunables

On some systems, the `idtune` command should be used to modify tunables. Refer to the `idtune`(1M) manual page for details.

On other systems, the tunables can be modified by adding lines to the `/etc/system` file and by then rebooting the system. Changed tunables are then in effect.

For example, to change the default value of a tunable called `vol_tuneme` to a value of 5000, the following line should be inserted into the appropriate section of the `/etc/system` file:

```
set vxio:vol_tuneme=5000
```

In many cases, the tunables are contained in the `volinfo` structure, as described in the `vxio`(7) manual page.

The sections that follow describe specific tunables.

vol_maxvol

This value controls the maximum number of volumes that can be created on the system. This value can be set to between 1 and the maximum number of minor numbers representable in the system.

The default value for this tunable is half the value of the maximum minor number value on the system.

vol_subdisk_num

This tunable is used to control the maximum number of subdisks that can be attached to a single plex. There is no theoretical limit to this number, but for practical purposes it has been limited to a default value of 4096. This default can be changed if required.

vol_maxioctl

This value controls the maximum size of data that can be passed into the Volume Manager via an `ioctl` call. Increasing this limit will allow larger operations to be performed. Decreasing the limit is not generally recommended since some utilities depend upon performing operations of a certain size and may fail unexpectedly if they issue oversized `ioctl` requests.

The default value for this tunable is 32768 bytes (32K).

vol_maxspecialio

This tunable controls the maximum size of an I/O that can be issued by an `ioctl` call. The `ioctl` request itself may be small, but may have requested a large I/O to be performed. This tunable limits the size of these I/Os. If necessary, a request that exceeds this value may be failed, or the I/O may be broken up and performed synchronously.

The default value for this tunable is 512 sectors (256K).

vol_maxio

This value controls the maximum size of logical I/O operations that can be performed without breaking up the request. Physical I/O requests larger than this value will be broken up and performed synchronously. Physical I/Os are broken up based on the capabilities of the disk device and are unaffected by changes to this maximum logical request limit.

The default value for this tunable is 512 sectors (256K).

To avoid undesirable I/O breakup, the vol_maxio parameter should be increased. To increase the value of vol_maxio, add an entry to /etc/system and reboot for the change to take effect. For example, the following line sets the maximum I/O size to 16M.

```
set vxio:vol_maxio=32768
```

This parameter is in sectors and is stored as a 16-bit number, so it cannot be set to a value larger than 65535. The value of vol_maxio determines the largest amount of memory that an I/O request can lock, so it should not be set to more than approximately 20 percent of memory.

vol_maxkiocount

This tunable controls the maximum number of I/Os that can be performed by the Volume Manager in parallel. Additional I/Os that attempt to use a volume device will be queued until the current activity count drops below this value.

The default value for this tunable is 2048.

Since most process threads can only issue a single I/O at a time, reaching the limit of active I/Os in the kernel would require 2K I/O requests being performed in parallel. Raising this limit seems unlikely to provide much benefit except on the largest of systems.

vol_default_iodelay

This value is the count in clock ticks that utilities will pause for between issuing I/Os if the utilities have been directed to throttle down the speed of their issuing I/Os, but have not been given a specific delay time. Utilities performing such operations as resynchronizing mirrors or rebuilding RAID-5 columns will use this value.

The default for this value is 50 ticks.

Increasing this value will result in slower recovery operations and consequently lower system impact while recoveries are being performed.

voldrl_min_regionsz

With Dirty Region Logging, the Volume Manager logically divides a volume into a set of consecutive regions. The `voldrl_min_regionsz` tunable specifies the minimum number of sectors for a DRL volume region.

The Volume Manager kernel currently sets the default value for this tunable to 1024 sectors.

Larger region sizes will tend to cause the cache hit-ratio for regions to improve. This will improve the write performance, but it will also prolong the recovery time.

voldrl_max_drtregs

This tunable specifies the maximum number of dirty regions that can exist on the system at any time. This is a global value applied to the entire system, regardless of how many active volumes the system has.

The default value for this tunable is 2048.

The tunable `voldrl_max_dtregs` can be used to regulate the worse-case recovery time for the system following a failure. A larger value may result in improved system performance at the expense of recovery time.

vol_maxparallelio

This tunable controls the number of I/O operations that the `vxconfigd`(1M) daemon is permitted to request from the kernel in a single `VOL_VOLDIO_READ` per `VOL_VOLDIO_WRITE` `ioctl` call.

The default value for this tunable is 256, and it is unlikely that it is desirable to change this value.

vol_mvr_maxround

This value controls the granularity of the round-robin policy for reading from mirrors. A read will be serviced by the same mirror as the last read if its offset is within the number of sectors described by this tunable of the last read.

The default for this value is 512 sectors (256K).

Increasing this value will cause less switches to alternate mirrors for reading. This is desirable if the I/O being performed is largely sequential with a few small seeks between I/Os. Large numbers of randomly distributed volume reads are generally best served by reading from alternate mirrors.

voliot_iobuf_limit

This value sets a limit to the size of memory that can be used for storing tracing buffers in the kernel. Tracing buffers are used by the Volume Manager kernel to store the tracing event records. As trace buffers are requested to be stored in the kernel, the memory for them is drawn from this pool.

Increasing this size can allow additional tracing to be performed at the expense of system memory usage. Setting this value to a size greater than can readily be accommodated on the system is inadvisable.

The default value for this tunable is 131072 bytes (128K).

voliot_iobuf_max

This value controls the maximum buffer size that can be used for a single trace buffer. Requests of a buffer larger than this size will be silently truncated to this size. A request for a maximal buffer size from the tracing interface will result (subject to limits of usage) in a buffer of this size.

The default size for this buffer is 65536 bytes (64K).

Increasing this buffer can provide for larger traces to be taken without loss for very heavily used volumes. Care should be taken not to increase this value above the value for the `voliot_iobuf_limit` tunable value.

voliot_iobuf_default

This value is the default size for the creation of a tracing buffer in the absence of any other specification of desired kernel buffer size as part of the trace `ioctl`.

The default size of this tunable is 8192 bytes (8K).

If trace data is often being lost due to this buffer size being too small, then this value can be tuned to a more generous amount.

voliot_errbuf_default

This tunable contains the default size of the buffer maintained for error tracing events. This buffer is allocated at driver load time and is not adjustable for size while the Volume Manager is running.

The default size for this buffer is 16384 bytes (16K).

Increasing this buffer can provide storage for more error events at the expense of system memory. Decreasing the size of the buffer could lead to a situation where an error cannot be detected via the tracing device. Applications that depend on error tracing to perform some responsive action are dependent on this buffer.

voliot_max_open

This value controls the maximum number of tracing channels that can be open simultaneously. Tracing channels are clone entry points into the tracing device driver. Each running `vxtrace` command on the system will consume a single trace channel.

The default number of channels is 32. The allocation of each channel takes up approximately 20 bytes even when not in use.

vol_checkpt_default

This tunable controls the interval at which utilities performing recoveries or resynchronization operations will load the current offset into the kernel such that a system failure will not require a full recovery, but can continue from the last reached checkpoint.

The default value of the checkpoint is 20480 sectors (10M).

Increasing this size reduces the overhead of checkpointing on recovery operations at the expense of additional recovery following a system failure during a recovery.

volraid_rsrtransmax

This RAID-5 tunable controls the maximum number of transient reconstruct operations that can be performed in parallel. A transient reconstruct operation is one which occurs on a non-degraded RAID-5 volume and was thus not predicted. By limiting the number of these operations that can occur simultaneously, the possibility of flooding the system with many reconstruct operations at the same time is removed, reducing the risk of causing memory starvation conditions.

The default number of these transient reconstructs that can be performed in parallel is 1.

Increasing this size may improve the initial performance on the system when a failure first occurs and before a detach of a failing object is performed, but can lead to possible memory starvation conditions.

voliomem_kvmap_size

This tunable defines the size of a kernel virtual memory region used for mapping I/O memory. This must be at least as large as (and should be larger than) `voliomem_max_memory`. This kernel virtual memory is allocated contiguously from kernelmap.

The default size for this tunable is 5M.

voliomem_base_memory

This tunable is the amount of memory allocated when the driver is loaded. It is intended to be large enough to support any possible single Volume Manager I/O, but no larger. More memory will be allocated as needed, but additional memory allocations are not guaranteed. If `voliomem_base_memory` is not large enough for an I/O, and no additional memory can be obtained, then the Volume Manager will hang.

The default size for this tunable is 512K.

voliomem_max_memory

This tunable is the maximum amount of memory that will be allocated by the Volume Manager for I/Os. This limit can be no larger than `voliomem_kvmap_size`. Smaller values lower the impact of the Volume Manager on other users of the system (for example, a small value ensures that more memory is available for file caching). Larger values improve Volume Manager throughput, particularly for RAID-5.

The default size for this tunable is 4M.

voliomem_chunk_size

System memory is allocated to and released from the Volume Manager using this granularity. A larger granularity reduces memory allocation overhead (somewhat) by allowing Volume Manager to keep hold of a larger amount of memory.

The default size for this tunable is 64K.

Tuning for Large Systems

On smaller systems (less than a hundred drives), tuning should be unnecessary and the Volume Manager should be capable of adopting reasonable defaults for all configuration parameters. On larger systems, however, there may be configurations that require additional control over the tuning of these parameters, both for capacity and performance reasons.

Generally, there are only a few significant decisions to be made when setting up the Volume Manager on a large system. One is to decide on the size of the disk groups and the number of configuration copies to maintain for each disk group. Another is to choose the size of the private region for all the disks in a disk group.

Larger disk groups have the advantage of providing a larger free-space pool for the `vxassist`(1M) command to select from, and also allow for the creation of larger arrays. Smaller disk groups do not, however, require as large a configuration database and so can exist with smaller private regions. Very large disk groups can eventually exhaust the private region size in the disk group with the result that no more configuration objects can be added to that disk group. At that point, the configuration either has to be split into multiple disk groups, or the private regions have to be enlarged. This involves re-initializing each disk in the disk group (and can involve reconfiguring everything and restoring from backup).

A general recommendation for users of disk array subsystems is to create a single disk group for each array so the disk group can be physically moved as a unit between systems.

The Number of Configuration Copies for a Disk Group

Selection of the number of configuration copies for a disk group is based on the trade-off between redundancy and performance. As a general rule, the fewer configuration copies that exist in a disk group, the quicker the group can be initially accessed, the faster the initial start of `vxconfigd`(1M) can proceed, and the quicker transactions can be performed on the disk group.

CAUTION! The risk of lower redundancy of the database copies is the loss of the configuration database. Loss of the database results in the loss of all objects in the database and all data contained in the disk group.

The default policy for configuration copies in the disk group is to allocate a configuration copy for each controller identified in the disk group, or for each target containing multiple addressable disks on the same target. This is sufficient from the redundancy perspective, but can lead to large numbers of configuration copies under some circumstances.

If this is the case, it is recommended to limit the number of configuration copies to a minimum of 4. The location of the copies is selected as before, according to maximal controller or target spread.

The mechanism for setting the number of copies for a disk group is to use the `vxdg init` command for a new group setup (see the `vxdg`(1M) manual page for details). Also, you can change copies of an existing group by using the `vxedit set` command (see the `vxedit`(1M) manual page for details). For example, to set a disk group called `foodg` to contain 5 copies, use this command:

```
vxedit set nconfig=5 foodg
```

Volume Manager Cluster Functionality

Introduction

This chapter discusses the cluster functionality provided with the VERITAS® Volume Manager (VxVM®). The Volume Manager includes an *optional* cluster feature that enables VxVM to be used in a cluster environment. The cluster functionality in the Volume Manager is a separately licensable feature.

The following topics are covered in this chapter:

- Cluster Functionality Overview
- Cluster-related Volume Manager Utilities and Daemons
- Error Messages
- Cluster Terminology

Cluster Functionality Overview

The cluster functionality in the Volume Manager allows multiple hosts to simultaneously access and manage a given set of disks under Volume Manager control (*VM disks*). 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. The Volume Manager cluster feature presents the same logical view of the disk configurations (including changes) on all nodes.

Note: With cluster support enabled, the Volume Manager supports up to four nodes per cluster.

The sections that follow provide more information on the cluster functionality provided by the Volume Manager.

Shared Volume Manager Objects

When the cluster feature is enabled, Volume Manager objects are capable of being shared by all of the nodes in a given cluster.

The Volume Manager cluster feature 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 may 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 may join the cluster.

In a Volume Manager cluster, most disk groups are shared. However, the root disk group (`rootdg`) is always a private disk group.

The VM 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 VM disk simultaneously. Likewise, all volumes in a shared disk group can be shared by all nodes in a cluster. A VM disk or volume is considered *shared* when it is accessed by more than one node at the same time.

Notes:

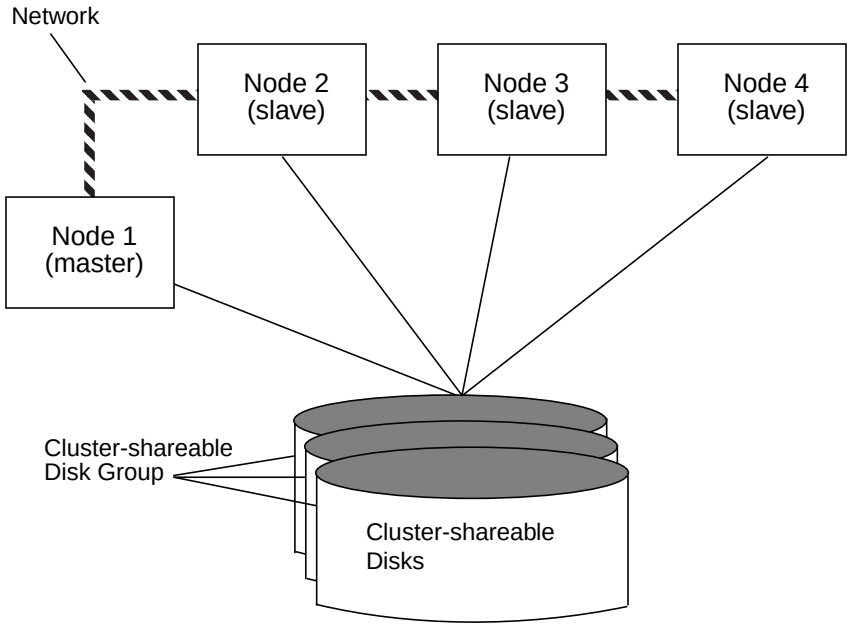
- The new features introduced in Volume Manager 3.0 are available in private disk groups, but are not yet supported for shared disk groups.
- Only raw device access is performed via the Volume Manager cluster feature. Shared volumes with file systems are not supported.
- The Volume Manager cluster feature does not currently support RAID-5 volumes in cluster-shareable disk groups. RAID-5 volumes can, however, be used in private disk groups attached to specific nodes of a cluster.

How Cluster Volume Management Works

The Volume Manager cluster feature works together with an externally-provided *cluster manager*, which is a daemon that informs VxVM of changes in cluster membership. Each node starts up independently and has its own copies of the operating system, VxVM with cluster support, 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 12 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, the Volume Manager cluster feature 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. VxVM 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 12 Example of a 4-Node Cluster



The system administrator designates a disk group as cluster-shareable using the `vxdg` utility (see the section entitled "`vxdg`" 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.

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 VM disks simultaneously.

Note: VxVM 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).

Configuration & 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 VxVM 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 VxVM 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 VxVM with cluster support, the cluster manager, and a distributed lock manager) on the node.

Once it is complete, applications may be started. The system administrator invokes the cluster manager startup procedure on each node to be joined to the cluster.

For VxVM in a cluster environment, 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, VxVM software chooses the master. Once the join for a given node is complete, that node has access to the shared disks.

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. Each node's cluster manager 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 VxVM 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 may 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 may 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 may 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 the section entitled "`vxclust`."

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, etc.). 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 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 it. 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 got. 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.

Node Shutdown

The system administrator can shut down the cluster on a given node by invoking the cluster manager's shutdown procedure on that node. This terminates cluster components after cluster applications have been stopped. VxVM 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.

The Volume Manager cluster feature maintains global state information for each volume. This enables VxVM 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, VxVM determines which volumes may have writes that have not completed and the master resynchronizes those volumes. If Dirty Region Logging 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 could 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 may take a long time if, for instance, a long-running transaction is active.

When VxVM's 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, VxVM makes them unavailable to applications. Since all nodes know that these volumes are closed on the leaving node, no resynchronizations are performed.

- If any volume in a shared disk group is open, the VxVM 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 shutdown has succeeded, the node has left the cluster. It is not possible to access the shared volumes until it joins the cluster again.

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

Node Abort

If a node does not leave cleanly, this 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 VxVM abort function. This function makes an attempt to halt all access to shared volumes at once, though the operation does wait 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.

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 also 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.

Disks in VxVM Clusters

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 can leave mirrored volumes 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 VxVM, 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.

Dirty Region Logging and Cluster Environments

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 describes how DRL behaves in a cluster environment. Refer to the other Volume Manager documentation for more information on DRL.

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* are used to store the dirty region log of a volume that has DRL enabled. A volume with DRL has at least one log subdisk, which 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.

In a cluster environment, the Volume Manager's implementation of DRL differs slightly from the normal implementation. The following sections outline some of the differences and discuss some aspects of the cluster environment implementation.

Log Format and Size

As with VxVM, the clustered dirty region log exists on a log subdisk in a mirrored volume.

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

The clustered dirty region log size is typically larger than a VxVM 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 three 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.

When nodes are added to an existing cluster, the existing DRL logs need to be detached and removed (using `vxplex -o rm dis`) and then recreated (using `vxassist addlog`). This increases the log sizes so that they can accommodate maps for the additional nodes.

Compatibility

Except for the addition of a cluster-specific magic number, DRL headers in a cluster environment are the same as their non-clustered counterparts.

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

If a shared disk group is imported by a VxVM system without cluster support, VxVM will consider the logs of the shared volumes to be invalid and will conduct a full volume recovery. After this recovery completes, the Volume Manager will use the cluster feature's Dirty Region Logging.

The Volume Manager cluster feature is capable of performing a DRL recovery on a non-shared VxVM volume. However, if a VxVM volume is moved to a VxVM system with cluster support and imported as shared, the dirty region log will probably be too small to accommodate all the nodes in the cluster. The cluster feature 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 the cluster feature will handle with a full volume recovery. In both cases, the system administrator is responsible for allocating a new log of sufficient size.

How DRL Works in a Cluster Environment

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 (i.e., 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 VxVM 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 VxVM 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.

Cluster-related Volume Manager Utilities and Daemons

The following utilities and/or daemons have been created or modified for use with the Volume Manager in a cluster environment:

- `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.

vxclust

Note: vxclust is not a portable utility. Although vxclust performs the same or similar operations in relation to the Volume Manager, 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 VxVM. Whenever there is a cluster reconfiguration, vxclust is called by the cluster manager. vxclust notifies the VxVM 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 the Volume Manager cluster feature should do next. After informing VxVM 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 decide whether to restart the reconfiguration or give up, depending on the circumstances. If the cause of the reconfiguration is a local, uncorrectable error, vxclust gives up. 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 may 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.

vxconfigd

The Volume Manager configuration daemon, vxconfigd, maintains configurations of VxVM 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):

- the cluster ID
- the node IDs
- the master node ID
- the node's role
- the network address of the vxconfigd on each node

On the master node, vxconfigd sets up the shared configuration (i.e., 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. See the section entitled "Volume Reconfiguration" for information on vxconfigd's role in volume reconfiguration.

vxconfigd Recovery

The Volume Manager vxconfigd daemon may 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 kernel's view 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 may 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's 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 VxVM, 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 would cause the loss of cluster information; if this option is used under these circumstances, vxconfigd will not start.

---

vxdg

The vxdg utility manages Volume Manager disk groups. vxdg can be used to specify that a disk group is cluster-shareable. 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, a shared disk group can be created 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, the disk groups can be imported 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 may 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 via 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, the system administrator must be careful to use the correct designation when adding a disk to a disk group. If the 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 the 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 only be used if the system administrator is fully aware of the possible consequences.

When a cluster is restarted, VxVM may 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, the system administrator can reimport the disk group 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 VxVM 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), the system administrator 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.

vxdisk

The `vxdisk` utility manages Volume Manager disks. `vxdisk` can be used 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 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`.

vxrecover

The `vxrecover` utility recovers plexes and volumes after disk replacement.

When a node leaves the cluster, it may 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 may be some degradation in system performance.

vxdctl

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

```
vxdctl -c mode
```

Depending on the circumstances, this displays output 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.

vxstat

vxstat returns statistics for specified objects. In a cluster 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 allows 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 might 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.

Error Messages

This section presents error messages that may occur with the Volume Manager in a cluster environment. Each message is accompanied by an explanation and a suggested user action.

Note: Some of these messages may appear on the console; others are returned by `vxclust`.

Message: `error in cluster processing`

```
error in cluster processing
```

▼ Clarification

This may 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 may also be caused by an unexpected sequence of commands from `vxclust`.

▼ Action

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

Message: `cannot find disk on slave node`

```
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.

Message: disk in use by another cluster

```
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.

Message: vxclust not there

```
vxclust not there
```

▼ Clarification

An error during an attempt to join the cluster caused `vxclust` to fail. This may 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 may clarify the problem.

Message: unable to add portal for cluster

```
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 may 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.

Message: vol recovery in progress

```
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).

Message: cannot assign minor

```
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.

Message: master sent no data

```
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.

Message: join in progress

```
join in progress
```

▼ Clarification

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

▼ Action

Retry later.

Message: join not allowed now

```
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.

Message: Disk reserved by other host

```
Disk reserved by other host
```

▼ Clarification

An attempt to online a disk 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 online it when the node joins the cluster.

Message: group exists

```
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.

Message: Plex detached from volume

```
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 may appear during a plex detach operation on a slave.

▼ Action

These messages provide information and require no user action.

Message: read error on Plex of shared volume; Plex detached from volume

```
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 may appear during a plex detach operation on the master.

▼ Action

These messages provide information and require no user action.

Message: return from cluster_establish is Configuration daemon error 242

```
return from cluster_establish is Configuration daemon error 242
```

▼ Clarification

This error may occur when a node fails to join the cluster or when a cluster join takes a long time. If the join fails, the node should retry the join automatically.

▼ Action

No action is necessary if the join is slow or a retry eventually succeeds.

Message: missing vxconfigd

```
node #: missing vxconfigd
```

▼ Clarification

The `vxconfigd` daemon is not running.

▼ Action

Restart the `vxconfigd` daemon.

Message: vxconfigd not ready

```
node #: vxconfigd is not communicating properly
```

▼ Clarification

The `vxconfigd` daemon is not responding properly.

▼ Action

Stop and restart the `vxconfigd` daemon.

Cluster Terminology

The following is a list of cluster-related terms and definitions:

clean node shutdown
The ability of a node to leave the cluster gracefully when all access to shared volumes has ceased.

cluster
A set of hosts that share a set of disks.

cluster manager
An externally-provided daemon that runs on each node in a cluster. The cluster managers on each node communicate with each other and inform VxVM of changes in cluster membership.

cluster-shareable disk group
A disk group in which the disks are shared by multiple hosts (also referred to as a *shared disk group*).

distributed lock manager
A lock manager that runs on different systems and ensures consistent access to distributed resources.

initiating node
The node on which the system administrator is running a utility that requests a change to Volume Manager objects. This node initiates a volume reconfiguration.

master node
A node that is designated by the software as the “master” node. Any node is capable of being the master node. The master node coordinates certain Volume Manager operations.

mastering node
The node to which a disk is attached. This is also known as a *disk owner.*

node
One of the hosts in a cluster.

node abort
A situation where a node leaves a cluster (on an emergency basis) without attempting to stop ongoing operations.

node join
The process through which a node joins a cluster and gains access to shared disks.

private disk group
A disk group in which the disks are accessed by only one specific host.

slave node
A node that is not designated as a master node.

shared disk group
A disk group in which the disks are shared by multiple hosts (also referred to as a *cluster-shareable disk group*).

shared volume
A volume that belongs to a shared disk group and is open on more than one node at the same time.

shared VM disk
A VM disk that belongs to a shared disk group.

Volume Manager Error Messages

Introduction

This appendix provides information on error messages associated with the Volume Manager configuration daemon (`vxconfigd`), the kernel, and other utilities. It covers most informational, failure, and error messages displayed (on the console) by `vxconfigd` and the kernel driver. These include some errors that are infrequently encountered and difficult to troubleshoot.

Note: Some error messages described here may not apply to your system.

Clarifications are included to elaborate on the situation or problem that generated a particular message. Wherever possible, a recovery procedure (Action) is provided to help you to locate and correct the problem.

The following sections are included in this appendix:

- Logging Error Messages
- Volume Manager Configuration Daemon Error Messages
  - `vxconfigd` Usage Messages
  - `vxconfigd` Error Messages
  - `vxconfigd` Fatal Error Messages
  - `vxconfigd` Notice Messages
  - `vxconfigd` Warning Messages
- Kernel Error Messages

  - Kernel Notice Messages
- Kernel Warning Messages
- Kernel Panic Messages

Logging Error Messages

The Volume Manager provides the option of logging console output to a file. This logging is useful in that any messages output just before a system crash will be available in the log file (presuming that the crash does not result in file system corruption). `vxconfigd` controls whether such logging is turned on or off. If enabled, the default log file is `/var/vxvm/vxconfigd.log`.

`vxconfigd` also supports the use of `syslog()` to log all of its regular console messages. When this is enabled, all console output is directed through the `syslog()` interface.

`syslog()` and log file logging can be used together to provide reliable logging (into a private log file), along with distributed logging through `syslogd`.

Both `syslog()` and log file logging are disabled by default.

To enable logging of console output to a file, you can either invoke `vxconfigd` as follows or edit Volume Manager startup scripts (described later):

```
vxconfigd -x log
```

To enable `syslog()` logging of console output, you can either invoke `vxconfigd` as follows or edit Volume Manager startup scripts (described later):

```
vxconfigd -x syslog
```

To enable log file and/or `syslog()` logging, you can also edit the following portion of the `/etc/init.d/vxvm-sysboot` startup script:

```
# comment-out or uncomment any of the following lines to enable or
# disable the corresponding feature in vxconfigd.

#opts="$opts -x syslog"                # use syslog for console
                                        messages
#opts="$opts -x log"                   # messages to
                                        /var/vxvm/vxconfigd.log
```

```
#opts="$opts -x logfile=/foo/bar"       # specify an alternate log
                                          file
#opts="$opts -x timestamp"             # timestamp console messages

# to turn on debugging console output, uncomment the following line.
# The debug level can be set higher for more output.  The highest
# debug level is 9.

#debug=1                               # enable debugging console output
```

Uncomment the line(s) corresponding to the feature(s) that you want enabled at startup. For example, to set up `vxconfigd` to automatically use file logging, uncomment the `opts="$opts -x log"` string.

For more information on logging options available through `vxconfigd`, refer to the `vxconfigd`(1M) manual page.

Volume Manager Configuration Daemon Error Messages

The Volume Manager is fault-tolerant and resolves most problems without system administrator intervention. If the Volume Manager configuration daemon (`vxconfigd`) recognizes what actions are necessary, it will queue up the transactions that are required. Volume Manager provides atomic changes of system configurations; either a transaction completes fully or the system appears as though the transaction was never attempted. When `vxconfigd` is unable to recognize and fix system problems, the system administrator needs to handle the task of problem solving.

The following sections cover the error messages associated with the Volume Manager configuration daemon.

`vxconfigd` Usage Messages

The following are usage messages associated with `vxconfigd`.

Message: `Usage: vxconfigd - long`

```
Usage: vxconfigd [-dkf] [-r reset] [-m mode] [-x level]
Recognized options:
```

```
-d          set initial mode to disabled for transactions
-k          kill the existing configuration daemon process
-f          operate in foreground; default is to operate in background
-r reset    reset kernel state; requires 'reset' option argument
-m mode     set vold's operating mode
            modes: disable, enable, bootload, bootstart
-x debug    set debugging level to <debug>, 0 turns off debugging
-R file     set filename for client request rendezvous
-D file     set filename for client diag request rendezvous
```

▼ Clarification

This is the full usage message for `vxconfigd`, which results from entering the command `vxconfigd help`.

Message: `Usage: vxconfigd - short`

```
Usage: vxconfigd [-dkf] [-r reset] [-m mode] [-x level]
For detailed help use: vxconfigd help
```

▼ Clarification

This is the standard `vxconfigd` usage error message. Appearance of this message implies that some option was supplied incorrectly.

▼ Action

If you need help in using `vxconfigd`, try using the command `vxconfigd help`.

For more detailed information, see the `vxconfigd`(1M) manual page.

Message: `-r must be followed by 'reset'`

```
-r must be followed by 'reset'
```

▼ Clarification

This is a usage error. The `-r` option requires an option argument consisting of the string `reset`.

▼ Action

Either don't use the `-r` option, or supply the `reset` option argument.

Message: -x argument: invalid debug string

-x argument: invalid debug string

▼ Clarification

An unrecognized string was specified as an argument to the -x option.

▼ Action

See vxconfigd(1M) for a list of valid arguments to -x.

Message: -x devprefix=device_prefix: prefix too long

-x devprefix=*device_prefix*: prefix too long

▼ Clarification

The -x devprefix=*device_prefix* option was used to define a prefix path for the /dev/dsk and /dev/rdsk directories, and that prefix was too long.

▼ Action

Use a shorter prefix.

vxconfigd Error Messages

The following are general error messages associated with vxconfigd.

Message: signal_name [core dumped]

vxvm:vxconfigd: ERROR: *signal_name* [- core dumped]

▼ Clarification

The vxconfigd daemon encountered an unexpected signal while starting up. The specific signal is indicated by *signal_name*. If the signal caused the vxconfigd process to dump core, then that will be indicated. This could be caused by a bug in vxconfigd, particularly if *signal_name* is "Segmentation fault." Alternately, this could have been caused by a user sending vxconfigd a signal with the kill utility.

▼ Action

Contact Customer Support.

Message: Unrecognized operating mode

vxvm:vxconfigd: ERROR: *mode_name*: Unrecognized operating mode

▼ Clarification

An invalid string was specified as an argument to the `-m` option. Valid strings are: `enable`, `disable`, and `boot`.

▼ Action

Supply a correct option argument.

Message: vxconfigd cannot boot-start RAID-5 volumes

vxvm:vxconfigd: ERROR: *volume_name*: vxconfigd cannot boot-start RAID-5 volumes

▼ Clarification

A volume that `vxconfigd` should start immediately upon booting the system (i.e., the volume for the `/usr` file system) has a RAID-5 layout. The `/usr` file system should never be defined on a RAID-5 volume.

▼ Action

It is likely that the only recovery for this is to boot the Volume Manager from a network-mounted root file system (or from a CD-ROM), and reconfigure the `/usr` file system to be defined on a regular non-RAID-5 volume.

Message: Cannot get all disk groups from the kernel

vxvm:vxconfigd: ERROR: Cannot get all disk groups from the kernel: *reason*

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `Cannot get all disks from the kernel`

`vxvm:vxconfigd: ERROR: Cannot get all disks from the kernel:` *reason*

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `Cannot get kernel transaction state`

`vxvm:vxconfigd: ERROR: Cannot get kernel transaction state:` *reason*

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `Cannot get private storage from kernel`

`vxvm:vxconfigd: ERROR: Cannot get private storage from kernel:` *reason*

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: Cannot get private storage size from kernel

vxvm:vxconfigd: ERROR: Cannot get private storage size from kernel: *reason*

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: Cannot get record from the kernel

vxvm:vxconfigd: ERROR: Cannot get record *record_name* from the kernel: *reason*

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: Cannot kill existing daemon, pid=process-ID

vxvm:vxconfigd: ERROR: Cannot kill existing daemon, pid=*process-ID*

▼ Clarification

The `-k` (kill existing `vxconfigd` process) option was specified, but a running configuration daemon process could not be killed. A configuration daemon process, for purposes of this discussion, is any process that opens the `/dev/vx/config` device (only one process can open that device at a time). If there is a configuration daemon process already running, then the `-k` option causes a `SIGKILL` signal to be send to that process. If, within a certain period of time, there is still a running configuration daemon process, then the above error message will be displayed.

▼ Action

This error can result from a kernel error that has made the configuration daemon process unkillable, from some other kind of kernel error, or from some other user starting another configuration daemon process after the SIGKILL signal. This last condition can be tested for by running vxconfigd -k again. If the error message appears again, contact Customer Support.

Message: Cannot make directory

vxvm:vxconfigd: ERROR: Cannot make directory *directory_path*: *reason*

▼ Clarification

vxconfigd failed to create a directory that it expects to be able to create. Directories that vxconfigd might try to create are: /dev/vx/dsk, /dev/vx/rdsk, and /var/vxvm/tempdb. Also, for each disk group, /dev/vx/dsk/*diskgroup* and /dev/vx/rdsk/*diskgroup* directories are created. The system error related to the failure is given in *reason*. A system error of "No such file or directory" indicates that one of the prefix directories (for example, /var/vxvm) does not exist.

This type of error normally implies that the Volume Manager packages were installed incorrectly. Such an error can also occur if alternate file or directory locations are specified on the command line, using the -x option. The _VXVM_ROOT_DIR environment variable may also relocate to a directory that lacks a var/vxvm subdirectory.

▼ Action

Try to create the directory manually and then issue the command vxdctl enable. If the error is due to incorrect installation of the Volume Manager packages, try to add the Volume Manager packages again.

Message: Cannot open /etc/vfstab

vxvm:vxconfigd: ERROR: Cannot open /etc/vfstab: *reason*

▼ Clarification

`vxconfigd` could not open the `/etc/vfstab` file, for the reason given. The `/etc/vfstab` file is used to determine which volume (if any) to use for the `/usr` file system. If the `/etc/vfstab` file cannot be opened, `vxconfigd` prints the above error message and exits.

▼ Action

This error implies that your root file system is currently unusable. You may be able to repair your root file system by mounting the root file system after booting from a network or CD-ROM root file system. If the root file system is defined on a volume, then see the procedures defined for recovering from a failed root file system in the "Recovery" appendix.

Message: `Cannot recover operation in progress`

```
vxvm:vxconfigd: ERROR: Cannot recover operation in progress
Failed to get group group from the kernel: error
```

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `Cannot reset VxVM kernel`

```
vxvm:vxconfigd: ERROR: Cannot reset VxVM kernel: reason
```

▼ Clarification

The `-r reset` option was specified to `vxconfigd`, but the VxVM kernel drivers could not be reset. The most common reason for this is "`A virtual disk device is open`." That error implies that a VxVM tracing device or volume device is open.

▼ Action

If, for some reason, you really want to reset the kernel devices, you will need to track down and kill all processes that have a volume or VxVM tracing device open. Also, if any volumes are mounted as file systems, unmount those file systems.

An error reason other than "`A virtual disk device is open`" should not normally occur unless there is a bug in the operating system or in the Volume Manager.

Message: `Cannot start volume, no valid complete plexes`

`vxvm:vxconfigd: ERROR: Cannot start` *volume* `volume, no valid complete plexes`

▼ Clarification

This error indicates that the volume for either the root or `/usr` file system cannot be started because the volume contains no valid plexes. This can happen, for example, if disk failures have caused all plexes to be unusable. It can also happen as a result of Actions that caused all plexes to become unusable (for example, forcing the dissociation of subdisks or detaching, dissociation, or offlining of plexes).

▼ Action

It is possible that this error results from a drive that failed to spin up. If so, rebooting may fix the problem. If that does not fix the problem, then the only recourse is to restore the root or `/usr` file system or to reinstall the system. Restoring the root or `/usr` file system requires that you have a valid backup. See the "Recovery" appendix for information on how to fix problems with root or `/usr` file system volumes.

Message: `Cannot start volume, no valid plexes`

`vxvm:vxconfigd: ERROR: Cannot start` *volume* `volume, no valid plexes`

▼ Clarification

This error indicates that the volume for either the root or /usr file system cannot be started because the volume contains no valid plexes. This can happen, for example, if disk failures have caused all plexes to be unusable. It can also happen as a result of Actions that caused all plexes to become unusable (for example, forcing the dissociation of subdisks or detaching, dissociating, or offlining plexes).

▼ Action

It is possible that this error results from a drive that failed to spin up. If so, rebooting may fix the problem. If that does not fix the problem, then the only recourse is to restore the root or /usr file system or to reinstall the system. Restoring the root or /usr file system requires that you have a valid backup. See the "Recovery" appendix for information on how to fix problems with root or /usr file system volumes.

Message: Cannot start volume, volume state is invalid

vxvm:vxconfigd: ERROR: Cannot start *volume* volume, volume state is invalid

▼ Clarification

The volume for the root or /usr file system is in an unexpected state (not ACTIVE, CLEAN, SYNC or NEEDSYNC). This should not happen unless the system administrator circumvents the mechanisms used by the Volume Manager to create these volumes.

▼ Action

The only recourse is to bring up the Volume Manager on a CD-ROM or NFS-mounted root file system and to fix the state of the volume. See the "Recovery" appendix for further information.

Message: Cannot store private storage into the kernel

vxvm:vxconfigd: ERROR: Cannot store private storage into the kernel: *error*

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: Differing version of vxconfigd installed

```
vxvm:vxconfigd: ERROR: Differing version of vxconfigd installed
```

▼ Clarification

A `vxconfigd` daemon was started after the stopping of an earlier `vxconfigd` with a non-matching version number. This can happen, for example, if you upgrade from an earlier release of Volume Manager to VxVM 3.0 and run `vxconfigd` without a reboot.

▼ Action

To fix, reboot the system.

Message: Disk, group, device: not updated with new host ID

```
vxvm:vxconfigd: ERROR: Disk disk, group group, device device: not
updated with new host ID
Error: reason
```

▼ Clarification

This can result from using `vxdctl hostid` to change the Volume Manager host ID for the system. The error indicates that one of the disks in a disk group could not be updated with the new host ID. Most likely, this indicates that the given disk has become inaccessible or has failed in some other way.

▼ Action

Try running the following to determine whether the disk is still operational:

`vxdisk check` *device*

If the disk is no longer operational, `vxdisk` should print a message such as:

device: Error: Disk write failure

This will result in the disk being taken out of active use in its disk group, if it has not been taken out of use already. If the disk is still operational (which should not be the case), vxdisk will print:

device: Okay

If the disk is listed as Okay, try vxdctl hostid again. If it still results in an error, contact Customer Support.

Message: Disk group, Disk: Cannot auto-import group

vxvm:vxconfigd: ERROR: Disk group *group*, Disk *disk*: Cannot auto-import group: *reason*

▼ Clarification

On system startup, vxconfigd failed to import the disk group associated with the named disk. A message related to the specific failure is given in *reason*. Additional error messages may be displayed that give more information on the specific error. In particular, this is often followed by:

vxvm:vxconfigd: ERROR: Disk group group: Errors in some configuration copies: Disk *device*, copy *number*: Block *bno*: *error* ...

The most common reason for auto-import failures is excessive numbers of disk failures, making it impossible for the Volume Manager to find correct copies of the disk group configuration database and kernel update log. Disk groups usually have enough copies of this configuration information to make such import failures unlikely.

A more serious failure is indicated by error types of:

```
Format error in configuration copy
Invalid magic number
Invalid block number
Duplicate record in configuration
Configuration records are inconsistent
```

These errors indicate that all configuration copies have become corrupt (due to disk failures, writing on the disk by an application or the administrator, or bugs in the Volume Manager).

Some correctable errors may be indicated by other error messages that appear in conjunction with the auto-import failure message. Look up those other errors for more information on their cause.

Failure of an auto-import implies that the volumes in that disk group will not be available for use. If there are file systems on those volumes, then the system may yield further errors resulting from inability to access the volume when mounting the file system.

▼ Action

If the error is clearly caused by excessive disk failures, then you may have to recreate the disk group and restore contents of any volumes from a backup. There may be other error messages that appear which provide further information. See those other error messages for more information on how to proceed. If those errors do not make it clear how to proceed, contact Customer Support.

Message: `Disk group, Disk: Group name collides with record in rootdg`

`vxvm:vxconfigd: ERROR: Disk group` *group*`, Disk` *device*`: Group name collides with record in rootdg`

▼ Clarification

The name of a disk group that is being imported conflicts with the name of a record in the `rootdg` disk group. Volume Manager does not allow this kind of conflict because of the way the `/dev/vx/dsk` directory is organized: devices corresponding to records in the root disk group share this directory with subdirectories for each disk group.

▼ Action

Either remove or rename the conflicting record in the root disk group, or rename the disk group on import. See the `vxdg`(1M) manual page for information on how to use the `import` operation to rename a disk group.

Message: `Disk group, Disk: Skip disk group with duplicate name`

`vxvm:vxconfigd: ERROR: Disk group` *group*`, Disk` *device*`: Skip disk group with duplicate name`

▼ Clarification

Two disk groups with the same name are tagged for auto-importing by the same host. Disk groups are identified both by a simple name and by a long unique identifier (disk group ID) assigned when the disk group is created. Thus, this error indicates that two disks indicate the same disk group name but a different disk group ID.

The Volume Manager does not allow you to create a disk group or import a disk group from another machine, if that would cause a collision with a disk group that is already imported. Therefore, this error is unlikely to occur under normal use. However, this error can occur in the following two cases:

- A disk group cannot be auto-imported due to some temporary failure. If you create a new disk group with the same name as the failed disk group and reboot, then the new disk group will be imported first, and the auto-import of the older disk group will fail with `group with duplicate name` (more recently modified disk groups have precedence over older disk groups).
- A disk group is deported from one host using the `-h` option to cause the disk group to be auto-imported on reboot from another host. If the second host was already auto-importing a disk group with the same name, then reboot of that host will yield this error.

▼ Action

If you want to import both disk groups, then rename the second disk group on import. See the `vxdg`(1M) manual page for information on how to use the `import` operation to rename a disk group.

Message: Disk group: Cannot recover temp database

`vxvm:vxconfigd: ERROR: Disk group` *group*`: Cannot recover temp database:` *reason*

`Consider use of "vxconfigd -x cleartempdir" [see vxconfigd(1M)].`

▼ Clarification

This can happen if you kill and restart `vxconfigd` or you if you disable and enable it with `vxdctl disable` and `vxdctl enable`. This error indicates a failure related to reading the file `/var/vxvm/tempdb/`*groupname*. This is a temporary file used to store information that is used when recovering the state of an earlier `vxconfigd`. The file is recreated on a reboot, so this error should never survive a reboot.

▼ Action

If you can reboot, do so. If you do not want to reboot, then do the following:

1. Ensure that no `vxvol`, `vxplex`, or `vxsd` processes are running.

   Use `ps -e` to search for such processes, and use `kill` to kill any that you find. You may have to run `kill` twice to make these processes go away. Killing utilities in this way may make it difficult to make administrative changes to some volumes until the system is rebooted.

2. Run the command:

   ```
   vxconfigd -x cleartempdir 2> /dev/console
   ```

   This will recreate the temporary database files for all imported disk groups.

   The `vxvol`, `vxplex`, and `vxsd` commands make use of these `tempdb` files to communicate locking information. If the file is cleared, then locking information can be lost. Without this locking information, two utilities can end up making incompatible changes to the configuration of a volume.

Message: `Disk group: Disabled by errors`

`vxvm:vxconfigd: ERROR: Disk group` *group*`: Disabled by errors`

▼ Clarification

This message indicates that some error condition has made it impossible for Volume Manager to continue to manage changes to a disk group. The major reason for this is that too many disks have failed, making it impossible for `vxconfigd` to continue to update configuration copies. There should be a preceding error message that indicates the specific error that was encountered.

If the disk group that was disabled is the rootdg disk group, then the following additional error should be displayed:

```
vxvm:vxconfigd: ERROR: All transactions are disabled
```

This additional message indicates that vxconfigd has entered the disabled state, which makes it impossible to change the configuration of any disk group, not just rootdg.

▼ Action

If the underlying error resulted from a transient failure, such as a disk cabling error, then you may be able to repair the situation by rebooting. Otherwise, the disk group may have to be recreated and restored from a backup. Failure of the rootdg disk group may require reinstallation of the system if your system uses a root or /usr file system defined on a volume.

Message: Disk group: Errors in some configuration copies: Disk, copy

vxvm:vxconfigd: ERROR: Disk group *group*: Errors in some configuration copies: Disk *disk*, copy *number*: [Block number]: *reason* ...

▼ Clarification

During a failed disk group import, some of the configuration copies in the named disk group were found to have format or other types of errors which make those copies unusable. This message lists all configuration copies that have uncorrected errors, including any appropriate logical block number. If no other reasons are displayed, then this may be the cause of the disk group import failure.

▼ Action

If some of the copies failed due to transient errors (such as cable failures), then a reboot or reimport may succeed in importing the disk group. Otherwise, the disk group may have to be recreated from scratch.

Message: Disk group: Reimport of disk group failed

vxvm:vxconfigd: ERROR: Disk group *group*: Reimport of disk group failed: *reason*

▼ Clarification

After `vxconfigd` was stopped and restarted (or disabled and then enabled), the Volume Manager failed to recreate the import of the indicated disk group. The reason for failure is specified. Additional error messages may be displayed that give further information describing the problem.

▼ Action

A major cause for this kind of failure is disk failures that were not addressed before `vxconfigd` was stopped or disabled. If the problem is a transient disk failure, then rebooting may take care of the condition.

Message: `Disk group: update failed`

`vxvm:vxconfigd: ERROR: Disk group` *group*`: update failed:` *reason*

▼ Clarification

I/O failures have prevented `vxconfigd` from updating any active copies of the disk group configuration. This usually implies a large number of disk failures. This error will usually be followed by the error:

`vxvm:vxconfigd: ERROR: Disk group` *group*`: Disabled by errors`

▼ Action

If the underlying error resulted from a transient failure, such as a disk cabling error, then you may be able to repair the situation by rebooting. Otherwise, the disk group may have to be recreated and restored from a backup.

Message: `Failed to store commit status list into kernel`

`vxvm:vxconfigd: ERROR: Failed to store commit status list into kernel:` *reason*

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `GET_VOLINFO ioctl failed`

`vxvm:vxconfigd: ERROR: GET_VOLINFO ioctl failed:` *reason*

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `Get of current rootdg failed`

`vxvm:vxconfigd: ERROR: Get of current rootdg failed:` *reason*

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `Memory allocation failure`

`vxvm:vxconfigd: ERROR: Memory allocation failure`

▼ Clarification

This implies that there is insufficient memory to start up the Volume Manager and to get the volumes for the root and `/usr` file systems running.

▼ Action

This error should not normally occur, unless your system has very small amounts of memory. Adding just swap space will probably not help because this error is most likely to occur early in the boot sequence, before swap areas have been added.

Message: Mount point: volume not in rootdg disk group

```
vxvm:vxconfigd: ERROR: Mount point path: volume not in rootdg disk
group
```

▼ Clarification

The volume device listed in the /etc/vfstab file for the given mount-point directory (normally /usr) is listed as in a disk group other than rootdg. This error should not occur if the standard Volume Manager procedures are used for encapsulating the disk containing the /usr file system.

▼ Action

You will need to boot the Volume Manager from a network or CD-ROM mounted root file system. Then, start up the Volume Manager using fixmountroot on a valid mirror disk of the root file system. After starting Volume Manager, mount the root file system volume and edit the /etc/vfstab file. Change the file to use a direct partition for the file system. There should be a comment in the /etc/vfstab file that indicates which partition to use, for example:

```
#NOTE: volume usr (/usr) encapsulated partition
```

c0t3d0s5 (or c0b0t3d0s5 if your system uses a bus).

Message: No convergence between root disk group and disk list

```
vxvm:vxconfigd: ERROR: No convergence between root disk group and
disk list
Disks in one version of rootdg:
   device type=device_type info=devinfo ...
Disks in alternate version of rootdg:
   device type=device_type info=devinfo ...
```

▼ Clarification

This message can appear when vxconfigd is not running in autoconfigure mode (see the vxconfigd(1M) manual page) and when, after several retries, it can not resolve the set of disks belonging to the root disk group. The algorithm for non-autoconfigure disks is to scan disks listed in the /etc/vx/volboot file and then examine the disks to find a database copy

for the rootdg disk group. The database copy is then read to find the list of disk access records for disks contained in the group. These disks are then examined to ensure that they contain the same database copy. As such, this algorithm expects to gain convergence on the set of disks and the database copies contained on them. If a loop is entered and convergence cannot be reached, then this message will appear and the root disk group importation will fail.

▼ Action

Reorganizing the physical locations of the devices attached to the system may break the deadlock. Failing this, contact Customer Support.

Message: Open of directory failed

vxvm:vxconfigd: ERROR: Open of directory *directory* failed: *reason*

▼ Clarification

An open failed for the /dev/vx/dsk or /dev/vx/rdsk directory (or a subdirectory of either of those directories). The only likely cause of such a failure should be that the directory was removed by the administrator or by an errant program. For this case, the *reason* should be "No such file or directory." An alternate possible cause is an I/O failure.

▼ Action

If the error was "No such file or directory," then create the directory (using mkdir). Then run the command vxdctl enable.

If the error was an I/O error, then there may be other serious damage to the root file system. You may need to reformat your root disk and restore the root file system from backup. Contact your system vendor or consult your system documentation.

Message: Read of directory failed

vxvm:vxconfigd: ERROR: Read of directory *directory* failed: *reason*

▼ Clarification

There was a failure in reading the /dev/vx/dsk or /dev/vx/rdsk directory (or a subdirectory of either of those directories). The only likely cause of this error is an I/O failure on the root file system.

▼ Action

If the error was an I/O error, then there may be other serious damage to the root file system. You may need to reformat your root disk and restore the root file system from backup. Contact your system vendor or consult your system documentation.

Message: System boot disk does not have a valid root plex

```
vxvm:vxconfigd: ERROR: System boot disk does not have a valid root
plex
Please boot from one of the following disks:
Disk: diskname Device: device ...
```

▼ Clarification

The system is configured to use a volume for the root file system, but was not booted on a disk containing a valid mirror of the root volume. Disks containing valid root mirrors are listed as part of the error message. A disk is usable as a boot disk if there is a root mirror on that disk which is not stale or offline.

▼ Action

Try to boot from one of the disks named in the error message.

Under some operating systems, you may be able to boot using a device alias for one of the named disks. For example, use this command:

```
boot vx-diskname
```

Message: System startup failed

```
vxvm:vxconfigd: ERROR: System startup failed
```

▼ Clarification

Either the root or the `/usr` file system volume could not be started, rendering the system unusable. The error that resulted in this condition should appear prior to this error message.

▼ Action

Look up other error messages appearing on the console and take the actions suggested in the descriptions of those messages.

Message: `There is no volume configured for the root device`

```
vxvm:vxconfigd: ERROR: There is no volume configured for the root
device
```

▼ Clarification

The system is configured to boot from a root file system defined on a volume, but there is no root volume listed in the configuration of the `rootdg` disk group.

There are two possible causes of this error:

- Case 1: The `/etc/system` file was erroneously updated to indicate that the root device is `/pseudo/vxio@0:0`. This should happen only as a result of direct manipulation by the administrator.
- Case 2: The system somehow has a duplicate `rootdg` disk group, one of which contains a root file system volume and one of which does not, and `vxconfigd` somehow chose the wrong one. Since `vxconfigd` chooses the more recently accessed version of `rootdg`, this error can happen if the system clock was updated incorrectly at some point (causing the apparent access order of the two disk groups to be reversed). This can also happen if some disk group was deported and renamed to `rootdg` with locks given to this host.

▼ Action

In case 1, boot the system on a CD-ROM or networking-mounted root file system, directly mount the disk partition of the root file system, and remove the following lines from `/etc/system`:

rootdev:/pseudo/vxio@0:0
set vxio:vol_rootdev_is_volume=1

In case 2, either boot with all drives in the offending version of `rootdg` turned off, or import and rename [see `vxdg`(1M)] the offending `rootdg` disk group from another host. In the case of turning off drives, run the following command after booting:

```
vxdg flush rootdg
```

This will update time stamps on the imported version of `rootdg`, which should make the correct version appear to be the more recently accessed. If this does not correct the problem, then contact Customer Support.

Message: Unexpected configuration tid for group found in kernel

`vxvm:vxconfigd: ERROR: Unexpected configuration tid for group` *group* `found in kernel`

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: Unexpected error during volume reconfiguration

`vxvm:vxconfigd: ERROR: Unexpected error during` *volume* `volume reconfiguration:` *reason*

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: Unexpected error fetching disk for volume

```
vxvm:vxconfigd: ERROR: Unexpected error fetching disk for disk
volume: reason
```

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: Unexpected values stored in the kernel

```
vxvm:vxconfigd: ERROR: Unexpected values stored in the kernel
```

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: Version number of kernel does not match vxconfigd

```
vxvm:vxconfigd: ERROR: Version number of kernel does not match
vxconfigd
```

▼ Clarification

The release of `vxconfigd` does not match the release of the Volume Manager kernel drivers. This should happen only as a result of upgrading Volume Manager, and then running `vxconfigd` without a reboot.

▼ Action

Reboot the system. If that does not cure the problem, then add the VxVM packages again.

Message: Volume for mount point /usr not found in rootdg disk group

vxvm:vxconfigd: ERROR: Volume *volume* for mount point /usr not found in rootdg disk group

▼ Clarification

The system is configured to boot with /usr mounted on a volume, but the volume associated with /usr is not listed in the configuration of the rootdg disk group. There are a couple of possible causes of this error:

- The /etc/vfstab file was erroneously updated to indicate the device for the /usr file system is a volume, but the volume named is not in the rootdg disk group. This should happen only as a result of direct manipulation by the administrator.
- The system somehow has a duplicate rootdg disk group, one of which contains the /usr file system volume and one of which does not (or uses a different volume name), and vxconfigd somehow chose the wrong rootdg. Since vxconfigd chooses the more recently accessed version of rootdg, this error can happen if the system clock was updated incorrectly at some point (causing the apparent access order of the two disk groups to be reversed). This can also happen if some disk group was deported and renamed to rootdg with locks given to this host.

▼ Action

In case 1, boot the system on a CD-ROM or networking-mounted root file system. If the root file system is defined on a volume, then start and mount the root volume using the procedures defined in the “Recovery” appendix. If the root file system is not defined on a volume, then just mount the root file system directly. Edit the /etc/vfstab file to correct the entry for the /usr file system.

In case 2, either boot with all drives in the offending version of rootdg turned off, or import and rename [see vxdg(1M)] the offending rootdg disk group from another host. In the case of turning off drives, run the following command after booting:

```
vxdg flush rootdg
```

This will update time stamps on the imported version of `rootdg`, which should make the correct version appear to be the more recently accessed. If this does not correct the problem, then contact Customer Support.

Message: `cannot open /dev/vx/config`

`vxvm:vxconfigd: ERROR: cannot open /dev/vx/config:` *reason*

▼ Clarification

The `/dev/vx/config` device could not be opened. `vxconfigd` uses this device to communicate with the Volume Manager kernel drivers. The *reason* string indicates the reason for the open failure. The most likely reason is `Device is already open`. This reason indicates that some process (most likely `vxconfigd`) already has `/dev/vx/config` open. Other less likely reasons are "`No such file or directory`" or "`No such device or address.`" For either of these two reasons, the two likely causes are:

- The Volume Manager package installation did not complete correctly.
- The device node was removed by the administrator or by an errant shell script.

▼ Action

For the reason "`Device is already open`," if you really want to run `vxconfigd`, then stop or kill the old one. You can kill whatever process has `vxconfigd` open by running the command:

```
vxdctl -k stop
```

For other failure reasons, consider re-adding the base Volume Manager package. This will reconfigure the device node and re-install the Volume Manager kernel device drivers. See the *VERITAS Volume Manager Installation Guide* for information on how to add the package using `pkgadd`. If you cannot re-add the package, then contact Customer Support for more information.

Message: `enable failed`

`vxvm:vxconfigd: ERROR: enable failed:` *reason*

▼ Clarification

Regular startup of `vxconfigd` failed for the stated reason. This error can also result from the command `vxdctl enable`. This error may include the following additional text:

additional-reason`; aborting`
This message indicates that the failure was fatal and that `vxconfigd` is forced to exit. The most likely cause that results in an abort is inability to create IPC channels for communicating with other utilities.

additional-reason`; transactions are disabled`
This message indicates that `vxconfigd` is continuing to run, but no configuration updates are possible until the error condition is repaired.

Additionally, this may be followed with:

`vxvm:vxconfigd: ERROR: Disk group` *group*`: Errors in some configuration copies:`
`Disk` *device*`, copy` *number*`: Block` *bno*`:` *error* `...`

Reasons for failure vary considerably. Other error messages may be displayed that further indicate the underlying problem. If the `Errors in some configuration copies` error occurs, then that may indicate the problem.

▼ Action

Evaluate other error messages occurring with this one to determine the root cause of the problem. Make changes suggested by the other errors and then retry the command.

Message: `/dev/vx/info`

`vxvm:vxconfigd: ERROR: /dev/vx/info:` *reason*

▼ Clarification

The `/dev/vx/info` device could not be opened, or did not respond to a Volume Manager kernel request. This error most likely indicates one of the following:

- The Volume Manager package installation did not complete correctly.
- The device node was removed by the administrator or by an errant shell script.

▼ Action

Consider re-adding the base Volume Manager package. This will reconfigure the device node and re-install the Volume Manager kernel device drivers. See the *VERITAS Volume Manager Installation Guide* for information on how to add the package using `pkgadd`.

`vxconfigd` Fatal Error Messages

The following are fatal error messages associated with `vxconfigd`.

Message: `Disk group rootdg: Inconsistency -- Not loaded into kernel`

```
vxvm:vxconfigd: FATAL ERROR: Disk group rootdg: Inconsistency -- Not
loaded into kernel
```

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `Group: Cannot update kernel`

`vxvm:vxconfigd: FATAL ERROR: Group` *group*`: Cannot update kernel`

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `Interprocess communication failure`

`vxvm:vxconfigd: FATAL ERROR: Interprocess communication failure:` *reason*

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `Invalid status stored in kernel`

```
vxvm:vxconfigd: FATAL ERROR: Invalid status stored in kernel
```

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `Memory allocation failure during startup`

```
vxvm:vxconfigd: FATAL ERROR: Memory allocation failure during
startup
```

▼ Clarification

This implies that there is insufficient memory to start up the Volume Manager and to get the volumes for the root and `/usr` file systems running.

▼ Action

This error should not normally occur, unless your system has very small amounts of memory. Adding just swap space probably will not help, because this error is most likely to occur early in the boot sequence, before swap areas have been added.

Message: `Rootdg cannot be imported during boot`

```
vxvm:vxconfigd: FATAL ERROR: Rootdg cannot be imported during boot
```

▼ Clarification

This is an internal Volume Manager error. This error should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: Unexpected threads failure

`vxvm:vxconfigd: FATAL ERROR: Unexpected threads failure:` *reason*

▼ Clarification

This is an unexpected operating system error. This error should not occur unless there is a bug in the Volume Manager or in the operating system multi-threading libraries.

▼ Action

Contact Customer Support for more information.

`vxconfigd` Notice Messages

The following are notice messages associated with `vxconfigd`.

Message: `Detached disk`

`vxvm:vxconfigd: NOTICE: Detached disk` *disk*

▼ Clarification

The named disk appears to have become unusable and was detached from its disk group. Additional messages may appear to indicate other records detached as a result of the disk detach.

▼ Action

If hot-relocation is enabled, the Volume Manager objects affected by the disk failure may be taken care of automatically. Mail will be sent to `root` indicating what actions were taken by the Volume Manager and what further actions the administrator should take.

Message: Detached log for volume

vxvm:vxconfigd: NOTICE: Detached log for volume *volume*

▼ Clarification

The DRL or RAID-5 log for the named volume was detached as a result of a disk failure, or as a result of the administrator removing a disk with vxdg -k rmdisk. A failing disk is indicated by a Detached disk *disk* message.

▼ Action

If the log is mirrored, hot-relocation may automatically relocate the failed log. Remove the failing logs using either vxplex dis or vxsd dis. Then, use vxassist addlog [see the vxassist(1M) manual page] to add a new log to the volume.

Message: Detached plex in volume

vxvm:vxconfigd: NOTICE: Detached plex *plex* in volume *volume*

▼ Clarification

The specified plex was disabled as a result of a disk failure, or as a result of the administrator removing a disk with vxdg -k rmdisk. A failing disk is indicated by a Detached disk *disk* message.

▼ Action

If hot-relocation is enabled, the Volume Manager objects affected by the disk failure may be taken care of automatically. Mail will be sent to root indicating what actions were taken by the Volume Manager and what further actions the administrator should take.

Message: Detached subdisk in volume

vxvm:vxconfigd: NOTICE: Detached subdisk *subdisk* in volume *volume*

▼ Clarification

The specified subdisk was disabled as a result of a disk failure, or as a result of the administrator removing a disk with vxdg -k rmdisk. A failing disk is indicated by a Detached disk *disk* message.

▼ Action

If hot-relocation is enabled, the Volume Manager objects affected by the disk failure may be taken care of automatically. Mail will be sent to `root` indicating what actions were taken by the Volume Manager and what further actions the administrator should take.

Message: `Detached volume`

`vxvm:vxconfigd: NOTICE: Detached volume` *volume*

▼ Clarification

The specified volume was detached as a result of a disk failure, or as a result of the administrator removing a disk with `vxdg -k rmdisk.` A failing disk is indicated by a `Detached disk` *disk* message. Unless the disk error is transient and can be fixed with a reboot, the contents of the volume should be considered lost.

▼ Action

There is no action to be taken. Contact Customer Support for more information.

Message: `Offlining config copy on disk`

`vxvm:vxconfigd: NOTICE: Offlining config copy` *number* `on disk` *disk*`: Reason:` *reason*

▼ Clarification

An I/O error caused the indicated configuration copy to be disabled. This is a notice only, and does not normally imply serious problems, unless this is the last active configuration copy in the disk group.

▼ Action

You should consider replacing the indicated disk, since this error implies that the disk has deteriorated to the point where write errors cannot be repaired automatically. This can also result from transient errors, such as cabling problems or power problems. Check for a cabling problem.

Message: Volume entering degraded mode

vxvm:vxconfigd: NOTICE: Volume *volume* entering degraded mode

▼ Clarification

The detach of a subdisk in the named RAID-5 volume has caused that volume to enter “degraded” mode. While in degraded mode, performance of the RAID-5 volume will be substantially reduced. More importantly, failure of another subdisk may leave the RAID-5 volume unusable. Also, if the RAID-5 volume does not have an active log, then failure of the system may leave the volume unusable.

▼ Action

If hot-relocation is enabled, the Volume Manager objects affected by the disk failure may be taken care of automatically. Mail will be sent to root indicating what actions were taken by the Volume Manager and what further actions the administrator should take.

vxconfigd **Warning Messages**

The following are warning messages associated with vxconfigd.

Message: Bad request: client, portal [REQUEST|DIAG], size

vxvm:vxconfigd: WARNING: Bad request *number*: client *number*, portal [REQUEST|DIAG], size *number*

▼ Clarification

This is a diagnostic message that indicates an invalid request generated by a utility that has connected to vxconfigd. This message indicates a bug in that connected utility.

▼ Action

If you are actually developing a new utility, then this error indicates a bug in your code. Otherwise, this error indicates a bug in the Volume Manager. Contact Customer Support for more information.

Message: Cannot change disk group record in kernel

vxvm:vxconfigd: WARNING: Cannot change disk group record in kernel: *reason*

▼ Clarification

This is an internal Volume Manager problem. This warning should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: Cannot create device

vxvm:vxconfigd: WARNING: Cannot create device *device_path*: *reason*

▼ Clarification

vxconfigd cannot create a device node either under /dev/vx/dsk or under /dev/vx/rdsk. This should happen only if the root file system has run out of inodes.

▼ Action

Try removing some files from the root file system. Then, regenerate the device node with the command:

```
vxdctl enable
```

Message: Cannot exec /usr/bin/rm to remove directory

vxvm:vxconfigd: WARNING: Cannot exec /usr/bin/rm to remove *directory*: *reason*

▼ Clarification

The given directory could not be removed because the /usr/bin/rm utility could not be executed by vxconfigd. This is not a serious error. The only side effect of a directory not being removed is that the directory and its contents continue to use space in the root file system. However, this does

imply that the /usr file system is not mounted, or on some systems, that the rm utility is missing or is not in its usual location. This may be a serious problem for the general running of your system.

▼ Action

If the /usr file system is not mounted, you need to determine how to get it mounted. If the rm utility is missing or is not in the /usr/bin directory, you should restore it from somewhere.

Message: Cannot fork to remove directory

vxvm:vxconfigd: WARNING: Cannot fork to remove directory *directory*: *reason*

▼ Clarification

The given directory could not be removed because vxconfigd could not fork in order to run the rm utility. This is not a serious error. The only side effect of a directory not being removed is that the directory and its contents will continue to use space in the root file system. The most likely cause for this error is that your system does not have enough memory or paging space to allow vxconfigd to fork.

▼ Action

If your system is this low on memory or paging space, then your overall system performance is probably being substantially effected. Consider adding more memory or paging space.

Message: Cannot issue internal transaction

vxvm:vxconfigd: WARNING: Cannot issue internal transaction: *reason*

▼ Clarification

This problem usually occurs only if there is a Volume Manager bug. However, it may occur in cases where memory is low.

▼ Action

Contact Customer Support for more information.

Message: Cannot open log file

vxvm:vxconfigd: WARNING: Cannot open log file *log_filename*: *reason*

▼ Clarification

The vxconfigd console output log file could not be opened for the given reason. A log file is opened if -x log is specified, or if a log file is specified with -x logfile=*file*. The default log file is /var/vxvm/vxconfigd.log. The most likely cause for failure is "No such file or directory," which indicates that the directory containing the log file does not exist.

▼ Action

Create any needed directories, or use a different log file path name.

Message: Detaching plex from volume

vxvm:vxconfigd: WARNING: Detaching plex *plex* from volume *volume*

▼ Clarification

The given plex is being detached from the given volume as part of starting the volume. This error only happens for volumes that are started automatically by vxconfigd at system startup (i.e., for the root and /usr file system volumes). The plex is being detached as a result of an I/O failure, a disk failure during startup or prior to the last system shutdown or crash, or a disk removal prior to the last system shutdown or crash.

▼ Action

If you want to ensure that the root or /usr file system retains the same number of active mirrors, then remove the given plex and add a new mirror using the vxassist mirror operation. You might also consider replacing any bad disks before using vxassist mirror.

Message: Disk in group flagged as shared; Disk skipped

vxvm:vxconfigd: WARNING: Disk *disk* in group *group* flagged as shared; Disk skipped

▼ Clarification

The given disk is listed as shared, but the running version of Volume Manager does not support shared disk groups. This message can usually be ignored.

▼ Action

There is no action to take. If you want to use the disk on this system, then use `vxdiskadd` to add the disk for use by the local system. However, do not do that if the disk really is in a shared disk group that is in use by other systems that are sharing this disk.

Message: `Disk in group locked by host Disk skipped`

`vxvm:vxconfigd: WARNING: Disk` *disk* `in group` *group* `locked by host` *hostid* `Disk skipped`

▼ Clarification

The given disk is listed as locked by the host with the listed Volume Manager hostid (usually the same as the system hostname). This message can usually be ignored.

▼ Action

There is no action to take. If you want to use the disk on this system, then use `vxdiskadd` to add the disk for use by the local system. However, *do not* do that if the disk really is in a disk group that is in use by another system that is sharing this disk.

Message: `Disk in group: Disk device not found`

`vxvm:vxconfigd: WARNING: Disk` *disk* `in group` *group*`: Disk device not found`

▼ Clarification

No physical disk can be found that matches the named disk in the given disk group. This is equivalent to failure of that disk. Physical disks are located by matching disk IDs stored in the Volume Manager header on a disk and disk IDs stored in the disk group configuration. The configuration contains the official list of disk IDs for all disks in a disk group (the IDs are contained in disk media configuration records). The physical disks are then

scanned to match that list against the disk IDs stored in disk headers. This error message is displayed for any disk IDs in the configuration that are not located in the disk header of any physical disk.

This may result from a transient failure (such as a poorly-attached cable, or from a disk that failed to spin up fast enough). Alternately, this may happen as a result of a disk being physically removed from the system, or from a disk that has become unusable due to a head crash or electronics failure.

Any RAID-5 or DRL log plexes on this disk will be unusable; any RAID-5 subdisks or mirrored plexes containing subdisks on this disk will also be unusable. These disk failures (particularly multiple disk failures) may cause one or more volumes to become unusable.

▼ Action

If hot-relocation is enabled, the Volume Manager objects affected by the disk failure may be taken care of automatically. Mail will be sent to `root` indicating what actions were taken by the Volume Manager and what further actions the administrator should take.

Message: `Disk in kernel is not a recognized type`

`vxvm:vxconfigd: WARNING: Disk` *disk* `in kernel is not a recognized type`

▼ Clarification

This is an internal Volume Manager problem. This warning should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `Disk names group, but group ID differs`

`vxvm:vxconfigd: WARNING: Disk` *disk* `names group` *group*`, but group ID differs`

▼ Clarification

As part of a disk group import, a disk was discovered that had a mismatched disk group name and disk group ID. This disk will not have been imported. This can only happen if two disk groups of the same name

exist that have different disk group ID values. In that case, one group will be imported along with all its disks and the other group will not. This message will appear for disks in the un-selected group.

▼ Action

If it turns out that the disk should be imported into the group, then this will have to be done by adding the disk to the group at a later stage. It will not happen automatically as part of the import. All configuration information for the disk will also be lost.

Message: Disk group is disabled, disks not updated with new host ID

vxvm:vxconfigd: WARNING: Disk group *group* is disabled, disks not updated with new host ID

▼ Clarification

As a result of failures, the named disk group has become disabled. Earlier error messages should indicate the cause of this. This warning message indicates that disks in that disk group were not updated with a new Volume Manager host ID.

This warning message should result only from a `vxdctl hostid` operation.

▼ Action

Typically, unless a disk group was disabled due to transient errors, there is no way to repair a disabled disk group. The disk group may have to be reconstructed from scratch. If the disk group was disabled due to a transient error (such as a cabling problem), then a future reboot may not automatically import the named disk group, due to the change in Volume Manager host ID for the system. In that case, the disk group should be imported directly using `vxdg import` with the `-C` option.

Message: Disk group: Disk group log may be too small

vxvm:vxconfigd: WARNING: Disk group *group*: Disk group log may be too small

Log size should be at least *number* blocks

▼ Clarification

The log areas for the disk group have become too small for the size of configuration currently in the group. This should normally never happen without first displaying a message about the database area size. This message only occurs during disk group import; it can only occur if the disk was inaccessible while new database objects were added to the configuration, and the disk was then made accessible and the system restarted.

▼ Action

If this situation does occur, then the disks in the group will have to be explicitly reinitialized with larger log areas (which would require data to be restored from backup). See the `vxdisk`(1M) manual page. To reinitialize all of the disks, they must be detached from the group with which they are associated and then reinitialized and re-added. The disk group should then be deported and re-imported for the changes to the log areas for the group to take effect.

Message: `Disk group: Errors in some configuration copies: Disk, copy`

`vxvm:vxconfigd: WARNING: Disk group` *group*`: Errors in some configuration copies: Disk` *disk*`, copy` *number*`: [Block number]:` *reason* `...`

▼ Clarification

During a disk group import, some of the configuration copies in the named disk group were found to have format or other types of errors which make those copies unusable. This message lists all configuration copies that have uncorrected errors, including any appropriate logical block number.

▼ Action

There are usually enough configuration copies for any disk group to ensure that these errors do not become a serious problem. No action is usually necessary.

Message: `Error in volboot file`

`vxvm:vxconfigd: WARNING: Error in volboot file:` *reason* `Entry: disk` *device disk_type disk_info*

▼ Clarification

The `/etc/vx/volboot` file includes an invalid disk entry. This error should occur only if the file was edited directly (for example, using the `vi` editor).

▼ Action

This is just a warning message. The offending entry can be removed using the command:

`vxdctl rm disk` *device*

Message: `Failed to store commit status list into kernel`

`vxvm:vxconfigd: WARNING: Failed to store commit status list into kernel:` *reason*

▼ Clarification

This is an internal Volume Manager problem. This warning should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `Failed to update voldinfo area in kernel`

`vxvm:vxconfigd: WARNING: Failed to update voldinfo area in kernel:` *reason*

▼ Clarification

This is an internal Volume Manager problem. This warning should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: Field too long in volboot file

vxvm:vxconfigd: WARNING: Field too long in volboot file:
Entry: disk *device disk_type disk_info*

▼ Clarification

The /etc/vx/volboot file includes a disk entry with a field that is larger than the size the Volume Manager supports. This error should occur only if the file was edited directly (for example, using the vi editor).

▼ Action

This is just a warning message. The offending entry can be removed using the command:

```
vxdctl rm disk device
```

Message: Get of record from kernel failed

vxvm:vxconfigd: WARNING: Get of record *record_name* from kernel failed: *reason*

▼ Clarification

This is an internal Volume Manager problem. This warning should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: Group: Duplicate virtual device number(s)

vxvm:vxconfigd: WARNING: Group *group*: Duplicate virtual device number(s):
Volume *volume* remapped from *major,minor* to *major,minor* ...

▼ Clarification

The configuration of the named disk group includes conflicting device numbers. A disk group configuration lists the recommended device number to use for each volume in the disk group. If two volumes in two disk groups happen to list the same device number, then one of the volumes must use

an alternate device number. This is called device number remapping. Remapping is a temporary change to a volume. If the other disk group is deported and the system is rebooted, then the volume that was remapped may no longer be remapped. Also, volumes that are remapped once are not guaranteed to be remapped to the same device number in further reboots.

▼ Action

You should use the `vxdg reminor` operation to renumber all volumes in the offending disk group permanently. See the `vxdg`(1M) manual page for more information.

Message: `Internal transaction failed`

`vxvm:vxconfigd: WARNING: Internal transaction failed:` *reason*

▼ Clarification

This problem usually occurs only if there is a Volume Manager bug. However, it may occur in cases where memory is low.

▼ Action

Contact Customer Support for more information.

Message: `cannot remove group from kernel`

`vxvm:vxconfigd: WARNING: cannot remove group` *group* `from kernel:` *reason*

▼ Clarification

This is an internal Volume Manager problem. This warning should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `client not recognized by VXVM library`

`vxvm:vxconfigd: WARNING: client` *number* `not recognized by VXVM library`

▼ Clarification

This is an internal Volume Manager problem. This warning should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `client not recognized`

`vxvm:vxconfigd: WARNING: client` *number* `not recognized`

▼ Clarification

This is an internal Volume Manager problem. This warning should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `library and vxconfigd disagree on existence of client`

`vxvm:vxconfigd: WARNING: library and vxconfigd disagree on existence of client` *number*

▼ Clarification

This is an internal Volume Manager problem. This warning should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `library specified non-existent client`

`vxvm:vxconfigd: WARNING: library specified non-existent client` *number*

▼ Clarification

This is an internal Volume Manager problem. This warning should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `response to client failed`

`vxvm:vxconfigd: WARNING: response to client` *number* `failed:` *reason*

▼ Clarification

This is an internal Volume Manager problem. This warning should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `vold_turnclient failed`

`vxvm:vxconfigd: WARNING: vold_turnclient(`*number*`) failed` *reason*

▼ Clarification

This is an internal Volume Manager problem. This warning should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Note: The following error messages refer to systems that use the DMP feature.

Message: `Path Failure detected by vxdmp driver`

`vxvm:vxdmp:NOTICE: Path failure on <major>/<minor>`

▼ Clarification

Appears when a path under the control of the DMP driver fails. The device number of the failed path is part of the message.

▼ Action

None.

Message: `Load of sd driver failed`

`vxvm:vxdmp:NOTICE:` *Could not load sd driver*

▼ Clarification

Appears when a path under the control of the DMP driver fails. The device number of the failed path is part of the message.

▼ Action

None.

Message: `Install of sd driver failed`

`vxvm:vxdmp:NOTICE:` *Could not install sd driver*

▼ Clarification

During initialization, if the `vxdmp` driver tries to load the `sd` driver. If the attempt fails, this message appears.

▼ Action

None.

Message: `Can't lock sd driver`

`vxvm:vxdmp:NOTICE:` *could not lock sd driver*

▼ Clarification

The `sd` driver is locked during `vxdmp` driver initialization to avoid unloading of the driver. The message is printed when the `sd` driver cannot be locked.

▼ Action

None.

Message: Load of ssd driver failed

vxvm:vxdmp:NOTICE: *Could not load ssd driver*

▼ Clarification

Appears when a path under the control of the DMP driver fails. The device number of the failed path is part of the message.

▼ Action

None.

Message: Install of ssd driver failed

vxvm:vxdmp:NOTICE: *Could not install ssd driver*

▼ Clarification

During initialization, the vxdmp driver tries to load the ssd driver. If the attempt fails, this message appears.

▼ Action

None.

Message: Can't lock ssd driver

vxvm:vxdmp:NOTICE: *could not lock ssd driver*

▼ Clarification

The ssd driver is locked during vxdmp driver initialization to avoid unloading of the driver. The message is printed when the ssd driver cannot be locked.

▼ Action

None.

Message: Attempt to disable controller failed

vxvm:vxdmp:NOTICE:Attempt to disable controller *controller_name* failed. Rootdisk has just one enabled path.

▼ Clarification

There is only one remaining active path to the root disk. This cannot be disabled. The user is trying to disable a controller accessible through this path.

▼ Action

Do not try to disable this controller. This is not allowed.

Kernel Error Messages

The following sections cover the kernel level error messages.

Kernel Notice Messages

The following are notice messages associated with the kernel.

Message: Can't open disk in group

vxvm:vxio:NOTICE: Can't open disk *disk* in group *disk_group*. If it is removable media (like a floppy), it may not be mounted or ready. Otherwise, there may be problems with the drive. Kernel error code *number*

▼ Clarification

The named disk cannot be accessed in the named disk group.

▼ Action

Ensure that the disk exists, is powered on, and is visible to the system.

Message: Can't close disk in group

vxvm:vxio:NOTICE: Can't close disk *disk* in group *disk_group*. If it is removable media (like a floppy), it may have been removed. Otherwise, there may be problems with the drive. Kernel error code *public_region_error/private_region_error*

▼ Clarification

This is unlikely to happen; closes cannot fail.

▼ Action

None.

Message: Read error on object of mirror in volume corrected

vxvm:vxio:NOTICE: read error on object *subdisk* of mirror plex in volume *volume* (start *offset*, length *length*) corrected.

▼ Clarification

A read error occurred, which caused a read of an alternate mirror and a writeback to the failing region. This writeback was successful and the data was corrected on disk.

▼ Action

No action is required. The problem was corrected automatically. The administrator may, however, note the failure as a reference because if the same region fails again or frequently, the error could indicate a more insidious failure and the disk should be reformatted at the next reasonable opportunity.

Message: *String* on volume device in disk group

vxvm:vxio:NOTICE: *string* on volume device_*#* (*device_name*) in disk group *group_name*

▼ Clarification

An application requested message. The application running on top of the Volume Manager has requested the output of this message.

▼ Action

Refer to documentation for the appropriate application for more information.

The following messages are associated with the Event Notification Mechanism in DMP:

Message: Disabled path belonging to dmpnode

vxvm:vxdmp:NOTICE:disabled path *path device number* belonging to dmpnode *dmpnode device number*

▼ Clarification

The path with the device number indicated by the message, has been marked disabled in the DMP database. This path is controlled by the DMP node indicated by the specified device number. This could have happened due to a hardware failure.

▼ Action

Check the underlying hardware in case you want to recover the desired path.

Message: Enabled path belonging to dmpnode

vxvm:vxdmp:NOTICE:enabled path *path device number* belonging to dmpnode *dmpnode device number*

▼ Clarification

The path with the device number indicated by the message, has been marked enabled in the DMP database. This path is controlled by the DMP node indicated by the specified device number. This can happen if a previously disabled path has been repaired and the user has reconfigured the DMP database using the vxdctl(1M) command or, automatically.

▼ Action

None.

Message: Disabled dmpnode

vxvm:vxdmp:NOTICE:disabled dmpnode *dmpnode device number*

▼ Clarification

The dmpnode with the device number indicated in the message, has been marked disabled in the DMP database. It will no longer be accessible to further IOs. It happens when all paths controlled by a DMP node are in the disabled state, and therefore inaccessible.

▼ Action

Check hardware or enable the appropriate controllers in order to get at least one path under this dmpnode in the enabled state. This will enable the dmpnode specified.

Message: Enabled dmpnode

vxvm:vxdmp:NOTICE:enabled dmpnode *dmpnode device number*

▼ Clarification

The dmpnode with the device number indicated by the message has been marked enabled in the DMP database. It will now allow IOs. This happens when at least one path controlled by this dmpnode has been enabled.

▼ Action

None.

Message: Disabled controller connected to a disk array

vxvm:vxdmp:NOTICE:disabled controller *controller name* connected to disk array *disk array serial number*

▼ Clarification

All paths that go through the specified controller connected to the specified disk array, are in disabled state. This happens user chooses to disable a particular controller for maintenance tasks.

▼ Action

None.

Message: Enabled controller connected to a disk array

vxvm:vxdmp:NOTICE:enabled controller *controller name* connected to disk array *disk array serial number*

▼ Clarification

All paths that go through the specified controller connected to the specified disk array, have been put into enabled state. This happens when the user chooses to enable a particular controller.

▼ Action

None.

Message: `Removed disk array`

`vxvm:vxdmp:NOTICE:removed disk array` *disk array serial number*

▼ Clarification

The specified disk array has been disconnected from the host or some hardware failure has resulted in the disk array becoming inaccessible to the host.

▼ Action

None.

Message: `Added disk array`

`vxvm:vxdmp:NOTICE:added disk array` *disk array serial number*

▼ Clarification

The new disk array has been added to the host.

▼ Action

None.

Kernel Warning Messages

The following are warning messages associated with the kernel.

Message: `Received spurious close`

`vxvm:vxio:WARNING: Device` *major*`,` *minor*`: Received spurious close`

▼ Clarification

This message happens if a close was received for an object that was previously not opened. This will only happen if the operating system is not correctly tracking opens and closes.

▼ Action

No action is necessary; the system will continue.

Message: `Failed to log the detach of the DRL volume`

`vxvm:vxio:WARNING: Failed to log the detach of the DRL volume` *volume*

▼ Clarification

An attempt to write a kernel log entry indicating the loss of a DRL volume failed. The attempted write to the log failed either because the kernel log is full or because of a write error to the drive. The volume will become detached.

▼ Action

Messages about log failures are often fatal, unless the problem is transient. However, the kernel log is sufficiently redundant that such errors are unlikely to occur.

If the problem is not transient (i.e. the drive cannot be fixed and brought back online without data loss), the disk group must be recreated from scratch and all of its volumes must be restored from backup. Even if the problem is transient, the system must be rebooted after correcting the problem.

If error messages were seen from the disk driver, it is likely that the last copy of the log failed due to a disk error. The failed drive in the disk group should be replaced and the log will then be re-initialized on the new drive. The failed volume can then be forced into an active state and the data recovered. See Chapter 1, “Recovery” for further information.

Message: `DRL volume is detached`

`vxvm:vxio:WARNING: DRL volume` *volume* `is detached`

▼ Clarification

A Dirty Region Logging volume became detached because a DRL log entry could not be written. This might be because of a media failure, in which case other errors may have been logged to the console.

▼ Action

The volume containing the DRL log will continue. If the system fails before the DRL can be repaired, a full recovery of the volume's contents may be necessary and will be performed automatically when the system is restarted. To recover the DRL capability, a new DRL log should be added to the volume using the `vxassist addlog` command.

Message: `Read error on mirror of volume`

`vxvm:vxio:WARNING: read error on mirror` *plex* `of volume` *volume* `offset` *offset* `length` *length*

▼ Clarification

An error was detected while reading a mirror. This error may lead to further action shown by later error messages.

▼ Action

If the volume is mirrored, no action is necessary at this time since the alternate mirror's contents will be written to the failing mirror; this is often sufficient to correct media failures. If this error occurs often but never leads to a plex detach, there may be a marginal region on the disk at the position shown. It may eventually be necessary to remove data from this disk (see the vxevac(1M) manual page) and then to reformat the drive. In the unmirrored case, this message indicates that some data could not be read. The file system or other application reading the data may report an additional error, but in either event, data has been lost. The volume can be partially salvaged and moved to another location if desired.

Message: `Write error on mirror of volume offset length`

`vxvm:vxio:WARNING: write error on mirror` *plex* `of volume` *volume* `offset` *offset* `length` *length*

▼ Clarification

An error was detected while writing a mirror. This error will generally be followed by a detach message, unless the volume is un-mirrored.

▼ Action

The disk reporting the error is failing to correctly store written data. If the volume is not mirrored, consider removing the data and reformatting the disk. If the volume is mirrored, it will become detached and you should consider replacing or reformatting the disk.

If this error occurs often but never leads to a plex detach, there may be a marginal region on the disk at the position shown. It may eventually be necessary to remove data from this disk (see the vxevac(1M) manual page) and then to reformat the drive.

Message: Object detached from volume

vxvm:vxio:WARNING: object *plex* detached from volume *volume*

▼ Clarification

An uncorrectable error was detected by the mirroring code and a mirror copy was detached.

▼ Action

To restore redundancy, it may be necessary to add another mirror. The disk on which the failure occurred should be evacuated and reformatted, if possible. If the drive has failed completely, it may need to be replaced.

Message: Overlapping mirror detached from volume

vxvm:vxio:WARNING: Overlapping mirror *plex* detached from volume *volume*

▼ Clarification

An error has occurred on the last complete plex in the mirrored volume. Any sparse mirrors that map the failing region must be detached so that they cannot be accessed to satisfy that failed region inconsistently. This message indicates that such an overlapping mirror was found and is being detached.

▼ Action

No Action is directly necessary. The message indicates that the volume may have left some data inaccessible at the failing region and that it is no longer redundantly stored.

Message: `Kernel log full`

`vxvm:vxio:WARNING: Kernel log full:` *volume* `detached`

▼ Clarification

A plex detach failed because the kernel log was full. As a result, the mirrored volume will become detached.

▼ Action

It is unlikely that this condition could ever occur. The only corrective action for the detached volume is to reboot the system.

Message: `Kernel log update failed`

`vxvm:vxio:WARNING: Kernel log update failed:` *volume* `detached`

▼ Clarification

A plex detach failed because the kernel log could not be flushed to disk. As a result, the mirrored volume will become detached. This could be caused by all the disks containing a kernel log going bad.

▼ Action

Correct the failed disks so that kernel logging can once again function.

Message: `Detaching RAID-5 volume`

`vxvm:vxio:WARNING: detaching RAID-5` *raidvol*

▼ Clarification

Either a double-failure condition in the RAID-5 volume has been detected in the kernel or some other fatal error is preventing further use of the array.

▼ Action

If two or more drives were lost due to a controller or power failure, then once the disks can be re-attached to the system, they should be recovered using the `vxrecover` utility. Check the console for other errors that may provide additional information as to the nature of the failure.

Message: Object detached from RAID-5 volume

vxvm:vxio:WARNING: object *subdisk* detached from RAID-5 *raidvol* at column *column* offset *offset*

▼ Clarification

A subdisk was detached from a RAID-5 volume at the specified column number and offset. This is caused by the failure of a disk or an uncorrectable error occurring on that disk.

▼ Action

Check the console for other error messages indicating the cause of the failure. If the disk has failed, then it should be replaced as soon as possible.

Message: RAID-5 volume entering degraded mode operation

vxvm:vxio:WARNING: RAID-5 *raidvol* entering degraded mode operation

▼ Clarification

This message occurs when an uncorrectable error has forced the detach of a subdisk. At this point, not all data disks exist to provide the data upon request. Instead, parity regions are required to regenerate the data for each stripe in the array. Accesses will consequently take longer and will involve reading from all drives in the stripe.

▼ Action

Check the console for other error messages indicating the cause of the failure. If the disk has failed, it should be replaced as soon as possible.

Message: Double failure condition detected on RAID-5 volume

vxvm:vxio:WARNING: Double failure condition detected on RAID-5 *raidvol*

▼ Clarification

Double-failures occur if I/O errors are received at the same altitude in the array from more than one column of the array. This could be caused by a controller failure causing more than a single drive to become unavailable;

by the loss of a second drive after having run in a degraded state for significant periods of time; or by two separate disk drives failing simultaneously (which is unlikely to happen).

▼ Action

If the condition is correctable and the drives recoverable, the conditions should be corrected. The volume can then be recovered using the `vxrecover`(1M) command.

Message: `Failure in RAID-5 logging operation`

`vxvm:vxio:WARNING: Failure in RAID-5 logging operation`
`vxvm:vxio:WARNING: log object` *object_name* `detached from RAID-5` *volume*

▼ Clarification

These two messages will appear together when a RAID-5 log has failed and has been detached.

▼ Action

To restore RAID-5 logging to the RAID-5 volume, simply create a new log region and attach it to the volume.

Message: `Stranded ilock on object`

`vxvm:vxio:WARNING: check_ilocks: stranded ilock on` *object_name* `start` *offset* `len` *length*

▼ Clarification

This is an internal Volume Manager problem. This warning should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `Overlapping ilocks`

`vxvm:vxio:WARNING: check_ilocks: overlapping ilocks:` *offset* `for length,` *offset* `for` *length*

▼ Clarification

This is an internal Volume Manager problem. This warning should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Message: `Illegal vminor encountered`

```
vxvm:vxio:WARNING: Illegal vminor encountered
```

▼ Clarification

If a volume device other than the root volume device is opened before a configuration has been loaded, this message could result.

▼ Action

No action should be necessary; an attempt to access a volume device was made before the volume daemon (`vxconfigd`) loaded the volume configuration. Under normal startup conditions, this message should not occur. If the operation is necessary, start the Volume Manager and re-attempt the operation.

Message: `Uncorrectable read error`

`vxvm:vxio:WARNING:` *object_type object_name* `block` *offset*`: Uncorrectable read error`

▼ Clarification

A read or write operation from the specified object failed. An error will be returned to the application.

▼ Action

This error represents lost data. The data may need to be restored and failed media may need to be repaired. Depending on the type of object failing and on the type of recovery suggested for that type, an appropriate recovery operation may be necessary.

Message: Uncorrectable read/write error

vxvm:vxio:WARNING: *object_type object_name* block *offset*:
Uncorrectable read error on *object_type object_name* block *offset*
vxvm:vxio:WARNING: *object_type* object_name block *offset*:
Uncorrectable write error on *object_type object_name* block *offset*

▼ Clarification

A read or write operation from the specified object failed. An error will be returned to the application. Although similar to the previous message, this message is able to supply more specific information about the failing object.

▼ Action

This error represents lost data. The data may need to be restored and failed media may need to be repaired. Depending on the type of object failing and on the type of recovery suggested for that type, an appropriate recovery operation may be necessary.

Message: Root volumes are not supported on your PROM version

vxvm:vxio:WARNING: Root volumes are not supported on your PROM version.

▼ Clarification

The Volume Manager requires the ability to access the PROMs for your SPARC hardware. If the PROMs are not a recent OpenBoot PROM type, then root volumes will not be usable.

▼ Action

If you have set up a root volume, then undo the configuration (by running vxunroot or removing the rootdev line from /etc/system) as soon as possible and contact your hardware vendor for an upgrade to your PROM level.

Message: Cannot find device number

vxvm:vxio:WARNING: Cannot find device number for *boot_path*

▼ Clarification

The supplied boot path was retrieved from the PROMs for your system. It cannot be converted to a valid device number.

▼ Action

Check your PROM settings for the correct boot string.

Message: `mod_install returned` *errno*

`vxvm:vxio:WARNING: mod_install returned` *errno*

▼ Clarification

A call made to the operating system `mod_install()` function to load the `vxio` driver failed.

▼ Action

Check your console for additional messages that may explain why the load failed. Also check the console messages log file for any additional messages that were logged but not displayed on the console.

Message: `subdisk failed in plex in volume`

`vxvm:vxio:WARNING:` *subdisk* `subdisk failed in plex` *plex* `in volume` *volume*

▼ Clarification

The kernel has detected a subdisk failure, which may mean that the underlying disk is failing.

▼ Action

Check for obvious problems with the disk (such as a disconnected cable). If hot-relocation is enabled and the disk is failing, the subdisk failure may be taken care of automatically.

Kernel Panic Messages

The following are panic messages associated with the kernel.

Message: Object association depth overflow

```
vxvm:vxio:PANIC: Object association depth overflow
```

▼ Clarification

This is an internal Volume Manager problem. This warning should not occur unless there is a bug in the Volume Manager.

▼ Action

Contact Customer Support for more information.

Utility Error Messages

Following is the error message from the API while trying to change the state of a controller.

Message: Attempt to enable a controller that is not available

```
vxvm:vxdmpadm:ERROR:Attempt to enable a controller that is not
available
```

▼ Clarification

This message is returned by the `vxdmpadm` utility when an attempt to enable a controller that is not working or physically present is made. To enable a controller, it should be visible to the OS and I/Os should be possible through it.

▼ Action

▼ Check hardware and see if this controller is present and I/Os can be done through it

Disk Array Overview

Introduction

This chapter describes the traditional disk arrays and provides a general overview of available arrays.

Disk Array Overview

This section provides an overview of traditional disk arrays.

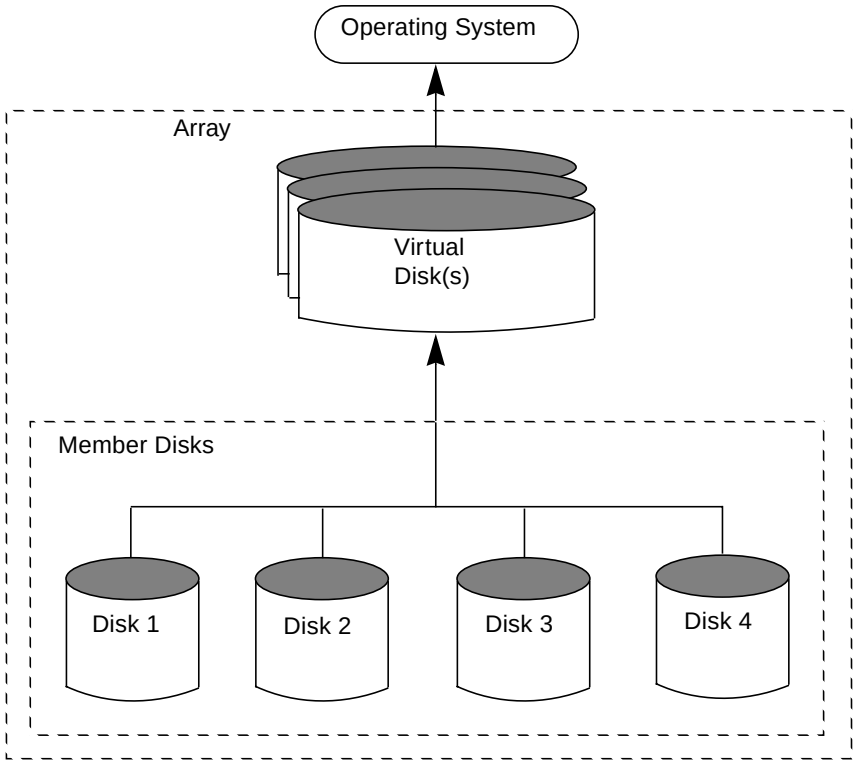
Performing I/O to disks is a slow process because disks are physical devices that require time to move the heads to the correct position on the disk before reading or writing. If all of the read or write operations are done to individual disks, one at a time, the read-write time can become unmanageable. Performing these operations on multiple disks can help to reduce this problem.

A *disk array* is a collection of disks that appears to the system as one or more virtual disks (also referred to as *volumes*). The virtual disks created by the software controlling the disk array look and act (to the system) like physical disks. Applications that interact with physical disks should work exactly the same with the virtual disks created by the array.

Data is spread across several disks within an array, which allows the disks to share I/O operations. The use of multiple disks for I/O improves I/O performance by increasing the data transfer speed and the overall throughput for the array.

Figure 1 shows a standard disk array.

Figure 1 Standard Disk Array



Redundant Arrays of Independent Disks (RAID)

A *Redundant Array of Independent Disks* (RAID) is a disk array set up so that part of the combined storage capacity is used for storing duplicate information about the data stored in the array. The duplicate information allows you to regenerate the data in case of a disk failure.

Several levels of RAID exist. These are introduced in the following sections.

Note: The Volume Manager supports RAID levels 0, 1, and 5 only.

For information on the Volume Manager's implementations of RAID, refer to "Volume Manager and RAID-5." in Chapter 1 of the *VERITAS Volume Manager Getting Started Guide.*

RAID-0

Although it does not provide redundancy, striping is often referred to as a form of RAID, known as RAID-0. The Volume Manager's implementation of striping is described in "Striping (RAID-0)." in Chapter 1 of the *VERITAS Volume Manager Getting Started Guide.* RAID-0 offers a high data transfer rate and high I/O throughput, but suffers lower reliability and availability than a single disk.

RAID-1

Mirroring is a form of RAID, which is known as RAID-1. The Volume Manager's implementation of mirroring is described in "Mirroring (RAID-1)." in Chapter 1of the *VERITAS Volume Manager Getting Started Guide.* Mirroring uses equal amounts of disk capacity to store the original plex and its mirror. Everything written to the original plex is also written to any mirrors. RAID-1 provides redundancy of data and offers protection against data loss in the event of physical disk failure.

RAID-2

RAID-2 uses bitwise striping across disks and uses additional disks to hold Hamming code check bits. RAID-2 is described in a University of California at Berkeley research paper entitled *A Case for Redundant Arrays of Inexpensive Disks (RAID)*, by David A. Patterson, Garth Gibson, and Randy H. Katz (1987).

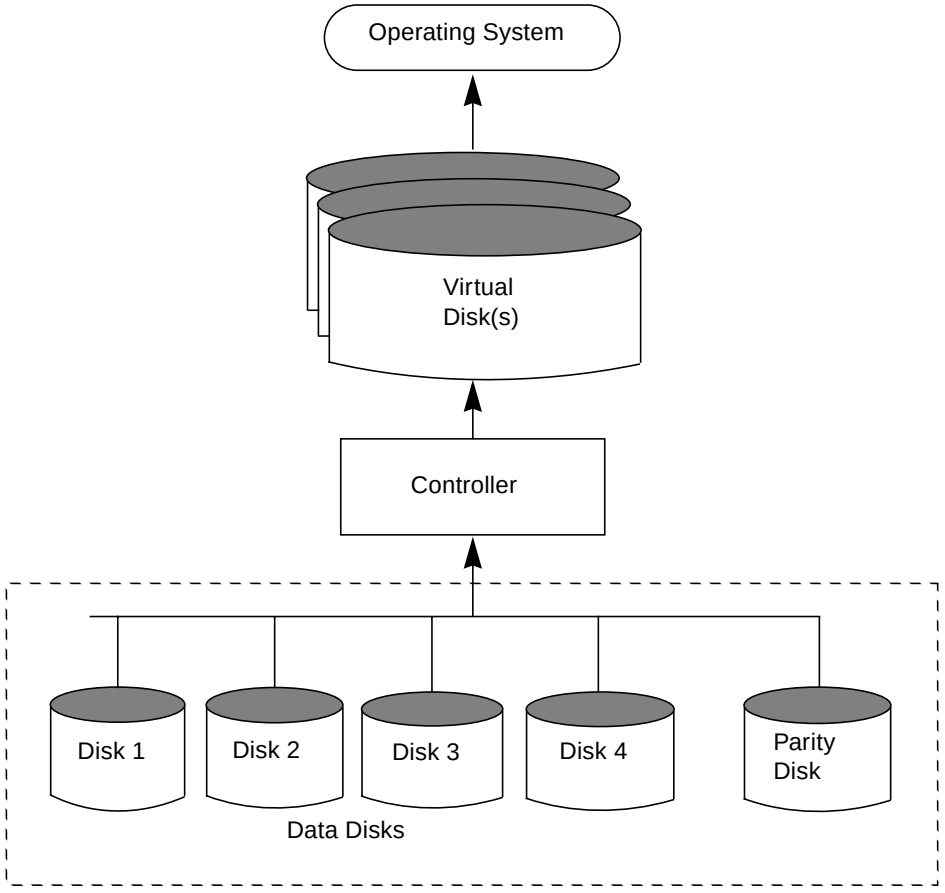
RAID-2 deals with error detection, but does not provide error correction. RAID-2 also requires large system block sizes, which limits its use.

RAID-3

RAID-3 uses a *parity disk* to provide redundancy. RAID-3 distributes the data in stripes across all but one of the disks in the array. It then writes the parity in the corresponding stripe on the remaining disk. This disk is the parity disk.

Figure 2 shows a RAID-3 disk array.

Figure 2 RAID-3 Disk Array

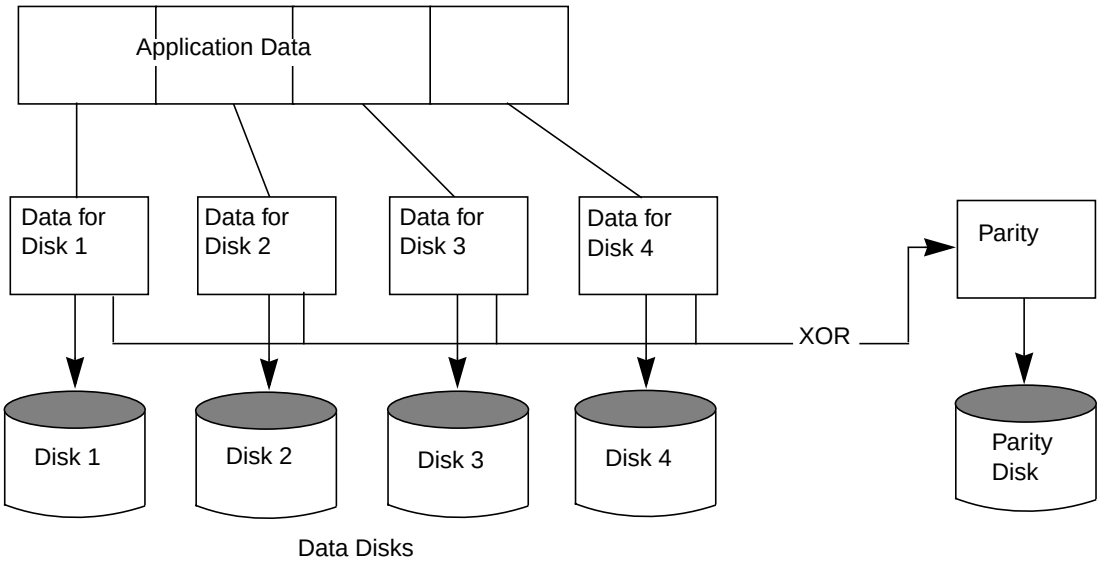


The user data is striped across the data disks. Each stripe on the parity disk contains the result of an exclusive OR (XOR) procedure done on the data in the data disks. If the data on one of the disks is inaccessible due to hardware or software failure, data can be restored by XORing the contents of the remaining data disks with the parity disk. The data on the failed disk can be rebuilt from the output of the XOR process.

RAID-3 typically uses a very small stripe unit size (also historically known as a *stripe width*), sometimes as small as one byte per disk (which requires special hardware) or one sector (block) per disk.

Figure 3 shows a data write to a RAID-3 array.

Figure 3 Data Writes to RAID-3



The parity disk model uses less disk space than mirroring, which uses equal amounts of storage capacity for the original data and the copy.

The RAID-3 model is often used with synchronized spindles in the disk devices. This synchronizes the disk rotation, providing constant rotational delay. This is useful in large parallel writes.

RAID-3 type performance can be emulated by configuring RAID-5 (described later) with very small stripe units.

RAID-4

RAID-4 introduces the use of independent-access arrays (also used by RAID-5). With this model, the system does not typically access all disks in the array when executing a single I/O procedure. This is achieved by ensuring that the stripe unit size is sufficiently large that the majority of I/Os to the array will only affect a single disk (for reads).

An array attempts to provide the highest rate of data transfer by spreading the I/O load as evenly as possible across all the disks in the array. In RAID-3, the I/O load is spread across the data disks, as shown in Figure 3, and each write is executed on all the disks in the array. The data in the data disk is XORed and the parity is written to the parity disk.

RAID-4 maps data and uses parity in the same manner as RAID-3, by striping the data across all the data disks and XORing the data for the information on the parity disk. The difference between RAID-3 and RAID-4 is that RAID-3 accesses all the disks at one time and RAID-4 accesses each disk independently. This allows the RAID-4 array to execute multiple I/O requests simultaneously (provided they map to different member disks), while RAID-3 can only execute one I/O request at a time.

RAID-4 read performance is much higher than its write performance. It performs well with applications requiring high read I/O rates. RAID-4 performance is not as high in small, write-intensive applications.

The parity disk can cause a bottleneck in the performance of RAID-4. This is because all the writes that are taking place simultaneously on the data disks must each wait its turn to write to the parity disk. The transfer rate of the entire RAID-4 array in a write-intensive application is limited to the transfer rate of the parity disk.

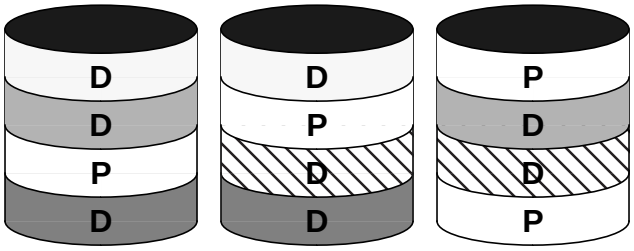
Since RAID-4 is limited to parity on one disk only, it is less useful than RAID-5.

RAID-5

RAID-5 is similar to RAID-4, using striping to spread the data over all the disks in the array and using independent access. However, RAID-5 differs from RAID-4 in that the parity is striped across all the disks in the array, rather than being concentrated on a single parity disk. This breaks the write bottleneck caused by the single parity disk write in the RAID-4 model.

Figure 4 shows parity locations in a RAID-5 array configuration. Every stripe has a column containing a parity stripe unit and columns containing data. The parity is spread over all of the disks in the array, reducing the write time for large independent writes because the writes do not have to wait until a single parity disk can accept the data.

Figure 4 Parity Locations in a RAID-5 Model



D = Data Stripe Unit
P = Parity Stripe Unit

For additional information on RAID-5 and how it is implemented by the Volume Manager, refer to “Volume Manager and RAID-5.” in Chapter 1 of the *VERITAS Volume Manager Getting Started Guide.*

Multipathed disk arrays

Some disk arrays provide multiple ports to access their disk devices. These ports, coupled with the HBA controller and any array-local data bus and I/O processor, make up multiple hardware paths to access the disk devices. Such disk arrays are called multipathed disk arrays. This type of disk array can be connected to host systems in many different configurations, for example, multiple ports connected to different controllers on a single host, chaining of the ports through a single controller on a host, or ports connected to different hosts simultaneously.

Multipathed disk arrays can be classified as either Active/Active type, or Active/Passive type.

Active/Passive type disk arrays

This type of disk array designates one of the multiple paths to a disk device as the primary path, and the others as secondary paths. Access to a disk is enabled through its primary path. If the primary path fails, one of the secondary paths is made the new primary path to the disk device, either automatically, or through administrator intervention. Access to the disk device through a secondary path may be disabled, or may degrade system performance severely.

In this type of disk array, a disk device is bound to one of the redundant hardware components local to the disk array, for example, the I/O bus, I/O controller, cache and access port. This disk is then said to be owned by the port, and the I/O path through that port is designated as the active path

Some disk arrays have a mode setting, called autotresspass mode, in which the disk arrays automatically mark a path as primary path when I/O is attempted through that path. This switch of primary path is an expensive operation, and degrades the performance of the disk array severely if different paths are used to access the disk alternately, by the same or different hosts.

Active/Active type disk arrays

This type of disk array allows access to the disk devices simultaneously (at any time) through all the paths that are available, without significant performance degradation. Thus, all the paths are active all the time, except for failed paths.

Glossary

Active/Active disk arrays
This type of multipathed disk array allows you to access a disk in the disk array through all the paths to the disk simultaneously, without any performance degradation.

Active/Passive disk arrays
This type of multipathed disk array allows one path to a disk to be designated as primary and used to access the disk at any time. Using a path other than the designated active path results in severe performance degradation in some disk arrays. See “path”, “primary path”, “secondary path”.

associate
The process of establishing a relationship between Volume Manager objects; for example, a subdisk that has been created and defined as having a starting point within a plex is referred to as being associated with that plex.

associated plex
A plex associated with a volume.

associated subdisk
A subdisk associated with a plex.

atomic operation
An operation that either succeeds completely or fails and leaves everything as it was before the operation was started. If the operation succeeds, all aspects of the operation take effect at once and the intermediate states of change are invisible. If any aspect of the operation fails, then the operation aborts without leaving partial changes.

attached
A state in which a VxVM object is both associated with another object and enabled for use.

block
The minimum unit of data transfer to a disk or array.

boot disk
A disk used for booting purposes. This disk may be under VxVM control.

clean node shutdown
The ability of a node to leave the cluster gracefully when all access to shared volumes has ceased.

cluster
A set of hosts that share a set of disks.

cluster manager
An externally-provided daemon that runs on each node in a cluster. The cluster managers on each node communicate with each other and inform VxVM of changes in cluster membership.

cluster-shareable disk group
A disk group in which the disks are shared by multiple hosts (also referred to as a *shared disk group*).

column
A set of one or more subdisks within a striped plex. Striping is achieved by allocating data alternately and evenly across the columns within a plex.

concatenation
A layout style characterized by subdisks that are arranged sequentially and contiguously.

configuration database
A set of records containing detailed information on existing Volume Manager objects (such as disk and volume attributes). A single copy of a configuration database is called a configuration copy.

data stripe
This represents the usable data portion of a stripe and is equal to the stripe minus the parity region.

detached
A state in which a VxVM object is associated with another object, but not enabled for use.

device name

The device name or address used to access a physical disk, such as `c0t0d0s2`. The `c#t#d#s#` syntax identifies the controller, target address, disk, and partition.

Dirty Region Logging

The procedure by which the Volume Manager monitors and logs modifications to a plex. A bitmap of changed regions is kept in an associated subdisk called a *log subdisk*.

disabled path

A path to a disk that is not available for I/O. A path can be *disabled* due to real hardware failures or if the user has used the `vxdmpadm disable` command on that controller.

disk

A collection of read/write data blocks that are indexed and can be accessed fairly quickly. Each disk has a universally unique identifier.

disk access name

The name used to access a physical disk, such as `c0t0d0s2`. The `c#t#d#s#` syntax identifies the controller, target address, disk, and partition. The term *device name* can also be used to refer to the disk access name.

disk access records

Configuration records used to specify the access path to particular disks. Each disk access record contains a name, a type, and possibly some type-specific information, which is used by the Volume Manager in deciding how to access and manipulate the disk that is defined by the disk access record.

disk array

A collection of disks logically arranged into an object. Arrays tend to provide benefits such as redundancy or improved performance.

disk array serial number

This is the serial number of the disk array. It is usually printed on the disk array cabinet or can be obtained by issuing a vendor specific SCSI command to the disks on the disk array. This number is used by the DMP subsystem to uniquely identify a disk array.

disk controller

The controller (HBA) connected to the host *or* the disk array that is represented as the parent node of the disk by the Operating System, is called the disk controller by the multipathing subsystem of Volume Manager.

For example, if a disk is represented by the device name:

```
/devices/sbus@1f,0/QLGC,isp@2,10000/sd@8,0:c
```

then the disk controller for the disk sd@8,0:c is:

```
QLGC,isp@2,10000
```

This controller (HBA) is connected to the host.

For an SSA disk connected to some machines, if a disk is represented by the device name:

```
/devices/iommu@0,10000000/sbus@0,10001000/SUNW,soc@2,0/SUNW,pln@b0
000000,78cd9f/ssd@5,0:c
```

then the disk controller for the disk `ssd@5,0:c` is:

```
SUNW,pln@b0000000,78cd9f
```

This controller is on the SSA disk array.

disk group

A collection of disks that share a common configuration. A disk group configuration is a set of records containing detailed information on existing Volume Manager objects (such as disk and volume attributes) and their relationships. Each disk group has an administrator-assigned name and an internally defined unique ID. The root disk group (`rootdg`) is a special private disk group that always exists.

disk group ID

A unique identifier used to identify a disk group.

disk ID

A universally unique identifier that is given to each disk and can be used to identify the disk, even if it is moved.

disk media name

A logical or administrative name chosen for the disk, such as `disk03`. The term *disk name* is also used to refer to the disk media name.

disk media record
A configuration record that identifies a particular disk, by disk ID, and gives that disk a logical (or administrative) name.

dissociate
The process by which any link that exists between two Volume Manager objects is removed. For example, dissociating a subdisk from a plex removes the subdisk from the plex and adds the subdisk to the free space pool.

dissociated plex
A plex dissociated from a volume.

dissociated subdisk
A subdisk dissociated from a plex.

distributed lock manager
A lock manager that runs on different systems and ensures consistent access to distributed resources.

enabled path
A path to a disk that is available for I/O.

encapsulation
A process that converts existing partitions on a specified disk to volumes. If any partitions contain file systems, `/etc/vfstab` entries are modified so that the file systems are mounted on volumes instead.

file system
A collection of files organized together into a structure. The UNIX file system is a hierarchical structure consisting of directories and files.

free space
An area of a disk under VxVM control that is not allocated to any subdisk or reserved for use by any other Volume Manager object.

free subdisk
A subdisk that is not associated with any plex and has an empty `putil[0]` field.

hostid
A string that identifies a host to the Volume Manager. The *hostid* for a host is stored in its `volboot` file, and is used in defining ownership of disks and disk groups.

hot-relocation
A technique of automatically restoring redundancy and access to mirrored and RAID-5 volumes when a disk fails. This is done by relocating the affected subdisks to disks designated as spares and/or free space in the same disk group.

initiating node
The node on which the system administrator is running a utility that requests a change to Volume Manager objects. This node initiates a volume reconfiguration.

log plex
A plex used to store a RAID-5 log. The term *log plex* may also be used to refer to a Dirty Region Logging plex.

log subdisk
A subdisk that is used to store a dirty region log. See *Dirty Region Logging*.

master node
A node that is designated by the software as the "master" node. Any node is capable of being the master node. The master node coordinates certain Volume Manager operations.

mastering node
The node to which a disk is attached. This is also known as a *disk owner*.

mirror
A duplicate copy of a volume and the data therein (in the form of an ordered collection of subdisks). Each mirror is one copy of the volume with which the mirror is associated. The terms *mirror* and *plex* can be used synonymously.

mirroring
A layout technique that mirrors the contents of a volume onto multiple plexes. Each plex duplicates the data stored on the volume, but the plexes themselves may have different layouts.

multipathing
Where there are multiple physical access paths to a disk connected to a system, the disk is called multipathed. Any software residing on the host, (e.g., the DMP driver) that hides this fact from the user is said to provide multipathing functionality.

node
One of the hosts in a cluster.

node abort
A situation where a node leaves a cluster (on an emergency basis) without attempting to stop ongoing operations.

node join
The process through which a node joins a cluster and gains access to shared disks.

object
An entity that is defined to and recognized internally by the Volume Manager. The VxVM objects are: volume, plex, subdisk, disk, and disk group. There are actually two types of disk objects—one for the physical aspect of the disk and the other for the logical aspect.

parity
A calculated value that can be used to reconstruct data after a failure. While data is being written to a RAID-5 volume, parity is also calculated by performing an *exclusive OR* (XOR) procedure on data. The resulting parity is then written to the volume. If a portion of a RAID-5 volume fails, the data that was on that portion of the failed volume can be recreated from the remaining data and the parity.

parity stripe unit
A RAID-5 volume storage region that contains parity information. The data contained in the parity stripe unit can be used to help reconstruct regions of a RAID-5 volume that are missing because of I/O or disk failures.

partition
The standard division of a physical disk device, as supported directly by the operating system and disk drives.

path
When a disk is connected to a host, the path to the disk consists of the HBA (Host Bus Adapter) on the host, the SCSI or fibre cable connector and the controller on the disk or disk array. These components constitute a path to a disk. A failure on any of these results in DMP trying to shift all I/Os for that disk onto the remaining(alternate) paths.

persistent state logging
A logging type that ensures that only active mirrors are used for recovery purposes and prevents failed mirrors from being selected for recovery. This is also known as *kernel logging*.

physical disk
The underlying storage device, which may or may not be under Volume Manager control.

plex
A duplicate copy of a volume and the data therein (in the form of an ordered collection of subdisks). Each plex is one copy of the volume with which the plex is associated. The terms *mirror* and *plex* can be used synonymously.

primary path
In Active/Passive type disk arrays, a disk can be bound to one particular controller on the disk array or owned by a controller. The disk can then be accessed using the path through this particular controller. See “path”, “secondary path”.

private disk group
A disk group in which the disks are accessed by only one specific host.

private region
A region of a physical disk used to store private, structured Volume Manager information. The *private region* contains a disk header, a table of contents, and a configuration database. The table of contents maps the contents of the disk. The disk header contains a disk ID. All data in the private region is duplicated for extra reliability.

public region
A region of a physical disk managed by the Volume Manager that contains available space and is used for allocating subdisks.

RAID
A Redundant Array of Independent Disks (RAID) is a disk array set up with part of the combined storage capacity used for storing duplicate information about the data stored in that array. This makes it possible to regenerate the data if a disk failure occurs.

read-writeback mode
A recovery mode in which each read operation recovers plex consistency for the region covered by the read. Plex consistency is recovered by reading data from blocks of one plex and writing the data to all other writable plexes.

root configuration
The configuration database for the root disk group. This is special in that it always contains records for other disk groups, which are used for backup purposes only. It also contains disk records that define all disk devices on the system.

root disk
The disk containing the root file system. This disk may be under VxVM control.

root disk group
A special private disk group that always exists on the system. The root disk group is named `rootdg`.

root file system
The initial file system mounted as part of the UNIX kernel startup sequence.

root partition
The disk region on which the root file system resides.

root volume
The VxVM volume that contains the root file system, if such a volume is designated by the system configuration.

rootability
The ability to place the `root` file system and the `swap` device under Volume Manager control. The resulting volumes can then be mirrored to provide redundancy and allow recovery in the event of disk failure.

secondary path
In Active/Passive type disk arrays, the paths to a disk other than the primary path are called secondary paths. A disk is supposed to be accessed only through the primary path until it fails, after which ownership of the disk is transferred to one of the secondary paths. See “path”, “primary path”.

sector
A unit of size, which can vary between systems. A sector is commonly 512 bytes.

shared disk group
A disk group in which the disks are shared by multiple hosts (also referred to as a *cluster-shareable disk group*).

shared volume
A volume that belongs to a shared disk group and is open on more than one node at the same time.

shared VM disk
A VM disk that belongs to a shared disk group.

slave node
A node that is not designated as a master node.

slice
The standard division of a logical disk device. The terms *partition* and *slice* are sometimes used synonymously.

spanning
A layout technique that permits a volume (and its file system or database) too large to fit on a single disk to span across multiple physical disks.

sparse plex
A plex that is not as long as the volume or that has holes (regions of the plex that don't have a backing subdisk).

stripe
A set of stripe units that occupy the same positions across a series of columns.

stripe size
The sum of the stripe unit sizes comprising a single stripe across all columns being striped.

stripe unit
Equally-sized areas that are allocated alternately on the subdisks (within columns) of each striped plex. In an array, this is a set of logically contiguous blocks that exist on each disk before allocations are made from the next disk in the array. A *stripe unit* may also be referred to as a *stripe element*.

stripe unit size
The size of each stripe unit. The default stripe unit size is 32 sectors (16K). A *stripe unit size* has also historically been referred to as a *stripe width*.

striping
A layout technique that spreads data across several physical disks using stripes. The data is allocated alternately to the stripes within the subdisks of each plex.

subdisk
A consecutive set of contiguous disk blocks that form a logical disk segment. Subdisks can be associated with plexes to form volumes.

swap area
A disk region used to hold copies of memory pages swapped out by the system pager process.

swap volume
A VxVM volume that is configured for use as a swap area.

transaction
A set of configuration changes that succeed or fail as a group, rather than individually. Transactions are used internally to maintain consistent configurations.

volboot file

A small file that is used to locate copies of the root configuration. The file may list disks that contain configuration copies in standard locations, and can also contain direct pointers to configuration copy locations. `volboot` is stored in a system-dependent location.

VM disk

A disk that is both under Volume Manager control and assigned to a disk group. VM disks are sometimes referred to as *Volume Manager disks* or simply *disks*. In the graphical user interface, VM disks are represented iconically as cylinders labeled `D`.

volume

A virtual disk, representing an addressable range of disk blocks used by applications such as file systems or databases. A volume is a collection of from one to 32 plexes.

volume configuration device

The volume configuration device (`/dev/vx/config`) is the interface through which all configuration changes to the volume device driver are performed.

volume device driver

The driver that forms the virtual disk drive between the application and the physical device driver level. The volume device driver is accessed through a virtual disk device node whose character device nodes appear in `/dev/vx/rdsk`, and whose block device nodes appear in `/dev/vx/dsk`.

volume event log

The volume event log device (`/dev/vx/event`) is the interface through which volume driver events are reported to the utilities.

vxconfigd

The Volume Manager configuration daemon, which is responsible for making changes to the VxVM configuration. This daemon must be running before VxVM operations can be performed.

Index

A

adding disks, 68
    format, 68
autoboot flag, 2

B

boot disk
    failure, 16
    failure and hot-relocation, 15
    re-adding, 16
    replacing, 16, 18
boot process, 2
booting
    after failure, 3

C

changing volume attributes, 56
checkpoint, 48
cluster
    disks, 124
    shared objects, 116
    terminology, 146
cluster environment, 115
cluster functionality, 115, 116
cluster reconfiguration, 120
cluster-shareable disk group, 116
command-line utilities, 67
configuration guidelines, 93
creating disk groups, 82
    vxdg, 82
creating RAID-5 volumes, 43
creating volumes, 42, 43

D

daemons
    configuration, 151
    vxrelocd, 75
data assignment, 94
degraded mode, 45
deporting
    disk groups, 83
device name, 64
dirty region logging
    in cluster environment, 125
disk arrays, 213
disk failures, 45
    and recovery, 1
disk group utilities, 67
disk groups, 66, 82
    creating, 82
    deporting, 83, 85
    importing, 83, 85
    moving, 83, 85
    moving between systems, 83
    removing, 83

renaming, 86
using, 83
disk media name, 64
disk names, 64, 66
disk utilities, 67
disks
adding, 68
boot disk, 15, 16
detached, 78
encapsulation, 4, 88
failure, 45, 73
and hot-relocation, 73
and recovery, 1
hot-relocation spares, 76
in a cluster, 124
initialization, 68
moving, 72
re-adding, 16
reattaching, 19
removing, 71
replacing, 16, 80, 81
root disk, 3, 4, 15
volatile, 90
DMP display, 91
DMP error messages, 195

E

encapsulation, 4, 88

F

failed disks, 73
detecting, 77
failures, 8, 49
and recovery procedures, 8
disk, 45
system, 44
forcibly starting volumes, 54
format utility, 68

G

getting performance data, 99
graphical user interface, 67

H

hosts
multiple, 116
hot-relocation, 73
boot disk, 15
modifying vxrelocd, 75

I

I/O
statistics, 100
obtaining, 99
tracing, 100, 104
importing
disk groups, 83
initializing disks, 68

L

log plexes, 39
log subdisks, 125
logs, 40, 50

M

manipulating subdisks, 50, 51
master node, 117
minor numbers
reserving, 87
mirroring, 95, 97, 215
mirrors
recover, 79
moving disk groups
vxdg, 84
vxrecover, 84
moving disks, 72

N

nodes, 116
nopriv, 89
    devices, 90

P

parity, 48
parity recovery, 47, 48
performance, 98
    guidelines, 94
    management, 93
    monitoring, 98
    optimizing, 94
    priorities, 98
performance data, 99
    getting, 99
    using, 100
plex kernel states, 35
    DETACHED, 35
    DISABLED, 35
    ENABLED, 35
plex states, 31, 32
    ACTIVE, 32
    CLEAN, 32
    EMPTY, 32
    IOFAIL, 34
    OFFLINE, 33
    STALE, 33
    TEMP, 33
    TEMPRM, 34
plex states cycle, 34
plexes, 39
private disk group, 116
private region, 65
public region, 65

R

RAID, 214
RAID-0, 215
RAID-1, 215
RAID-2, 215
RAID-3, 215
RAID-4, 218
RAID-5, 39, 40, 42, 43, 44, 45, 47, 48, 50, 51, 52, 53, 54, 56, 98, 218
    recovery, 46, 54
    subdisk moves, 51
RAID-5 plexes, 39
RAID-5 volumes, 54
read
    policies, 96
re-adding disks, 16
reattaching disks, 19
reconfiguration procedures, 21
recontructing-read, 45
recovery, 1, 54
    logs, 49
    procedures, 8
    RAID-5 volumes, 46, 54
reinstallation, 21, 22
removing disk groups, 83
    vxdg, 83
removing disks, 71
    vxdg, 71
renaming disk groups, 86
replacing disks, 16, 80
    vxdiskadm, 80
root disk, 3, 4, 15
root file system, 4
rootability
    cleanup, 25
rootdg, 66
    renaming, 86

S

shared objects, 116
slave node, 117
special devices

using, 88
special encapsulations
vxdisk, 88
spindles, synchronized, 217
standard disk devices, 64
starting volumes, 52
states
plex, 31
volume, 35
striping, 94, 97, 215
subdisk moves
RAID-5, 51
subdisks
log, 125
system failures, 44

T

tracing I/O, 100
vxtrace, 100
tunables, 104
tuning
Volume Manager, 104

U

unusable volumes, 52, 53
using disk groups
vxassist, 83
using disks, 68
using I/O statistics, 100
using performance data, 100
using special devices, 88

V

volume kernel states, 37
DETACHED, 37
DISABLED, 37
ENABLED, 37
Volume Manager graphical user interface, 67
Volume Manager Support Operations, 65
volume reconfiguration, 121
volume states, 35
ACTIVE, 35, 36
CLEAN, 35, 36
EMPTY, 35, 36
SYNC, 35, 36
volumes
cleanup, 25
layout, 39
vxassist, 82, 83
vxclust, 129
vxconfigd, 130, 151
vxdctl, 137
vxdg, 67, 71, 82, 84, 85, 132
moving disk groups, 84
removing disk groups, 83
vxdisk, 67, 71, 89, 135
rm, 71
special encapsulations, 88
vxdiskadd, 65, 67, 68, 82
vxdiskadm, 65, 67, 68, 71, 72, 80, 85
replacing disks, 80
vxedit, 25
vxinfo, 80
vxmake, 42, 43
vxmirror, 4
vxprint, 77
vxreattach, 19
vxrecover, 79, 136
moving disk groups, 84
vxrelocd, 75
modifying, 75
vxstat, 78, 99, 101, 138
vxtrace, 99, 100
vxvol, 81

W

writes
full-stripe, 57
parallel, 217
read-modify, 57
reconstruct, 60