

VERITAS Volume Manager 3.1 Administrator's Guide

for HP-UX 11i and HP-UX 11i Version 1.5

June, 2001



**Manufacturing Part Number: B7961-90018
E0601**

United States

© Copyright 1983-2000 Hewlett-Packard Company. All rights reserved.

Legal Notices

The information in this document is subject to change without notice.

Hewlett-Packard makes no warranty of any kind with regard to this manual, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Warranty

A copy of the specific warranty terms applicable to your Hewlett-Packard product and replacement parts can be obtained from your local Sales and Service Office.

Restricted Rights Legend

Use, duplication or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 for DOD agencies, and subparagraphs (c) (1) and (c) (2) of the Commercial Computer Software Restricted Rights clause at FAR 52.227-19 for other agencies.

HEWLETT-PACKARD COMPANY
3000 Hanover Street
Palo Alto, California 94304 U.S.A.

Use of this document and any supporting software media (CD-ROMs, flexible disk, and tape cartridges) supplied for this pack is restricted to this product only. Additional copies of the programs may be made for security and back-up purposes only. Resale of the programs, in their present form or with alterations, is expressly prohibited.

Copyright Notices

Copyright © 1983-2000 Hewlett-Packard Company. All rights reserved. Reproduction, adaptation, or translation of this document without prior written permission is prohibited, except as allowed under the copyright laws.

Copyright © 1979, 1980, 1983, 1985-93 Regents of the University of California. This software is based in part on the Fourth Berkeley Software Distribution under license from the Regents of the University of California.

Copyright © 2000 VERITAS Software Corporation
Copyright © 1988 Carnegie Mellon University
Copyright © 1990-1995 Cornell University
Copyright © 1986 Digital Equipment Corporation.
Copyright © 1997 Isogon Corporation
Copyright © 1985, 1986, 1988 Massachusetts Institute of Technology.
Copyright © 1991-1997 Mentat, Inc.
Copyright © 1996 Morning Star Technologies, Inc.
Copyright © 1990 Motorola, Inc.
Copyright © 1980, 1984, 1986 Novell, Inc.
Copyright © 1989-1993 The Open Software Foundation, Inc.
Copyright © 1996 Progressive Systems, Inc.
Copyright © 1989-1991 The University of Maryland
Copyright © 1986-1992 Sun Microsystems, Inc.

Trademark Notices

UNIX® is a registered trademark in the United States and other countries, licensed exclusively through The Open Group.

VERITAS® is a registered trademark of VERITAS Software Corporation in the US and other countries.

VERITAS File System™ is a trademark of VERITAS Software Corporation.

Copyright © 2000 VERITAS Software Corporation. All rights reserved.

VERITAS Volume Manager™ is a trademark of VERITAS Software Corporation.

VERITAS Volume Manager Storage Administrator™ is a trademark of VERITAS Software Corporation.

All other trademarks or registered trademarks are the property of their respective owners.

Publication History

The manual publication date and part number indicate its current edition. The publication date will change when a new edition is released. The manual part number will change when extensive changes are made.

To ensure that you receive the new editions, you should subscribe to the appropriate product support service. See your HP sales representative for details.

1. Introduction to Volume Manager

Introduction	26
How Data is Stored	28
Volume Manager Overview	29
Physical Objects	30
Physical Disks and Disk Naming	30
Partitions	30
Volumes and Virtual Objects	31
Volume Manager Disks	31
Disk Groups	32
Subdisks	32
Plexes	33
Volumes	34
Connection Between Volume Manager Virtual Objects	36
Virtual Object Data Organization (Volume Layouts)	38
Related Graphical User Interface (GUI) Terms	38
Concatenation	38
Striping (RAID-0)	40
RAID-5	44
Mirroring (RAID-1)	45
Mirroring Plus Striping (RAID-1 + RAID-0)	46
Striping Plus Mirroring (RAID-0 + RAID-1)	46
Volume Manager and RAID-5	48
Traditional RAID-5 Arrays	48
Volume Manager RAID-5 Arrays	49
Logging	52
Layered Volumes	54
Volume Manager User Interfaces	56
User Interface Overview	56
Volume Manager Conceptual Overview	58
Why You Should Use Volume Manager	58
Volume Manager Objects	58
Volume Manager and the Operating System	60

Volume Manager Layouts.....	62
Volume Administration Tasks.....	64
2. Initialization and Setup	
Introduction	68
Volume Manager Initialization	69
Volume Manager Daemons	71
Configuration Daemon vxconfigd.....	71
Volume I/O Daemon vxiod	72
System Setup	73
Example System Setup Sequence	73
System Setup Guidelines.....	74
Hot-Relocation Guidelines	74
Striping Guidelines.....	75
Mirroring Guidelines	77
Dirty Region Logging (DRL) Guidelines	78
Mirroring and Striping Guidelines	79
Striping and Mirroring Guidelines	79
RAID-5 Guidelines	80
Protecting Your System	81
3. Volume Manager Operations	
Introduction	84
Displaying Disk Configuration Information.....	86
Displaying Disk Listings	86
Displaying Volume Manager Object Listings	86
Displaying Free Space in a Disk Group.....	87
Displaying Subdisk Information.....	89
Disk group: rootdg	89
Disk group: rootdg	89
Creating Volumes.....	91
Volume Manager Task Monitor.....	93

Task Monitor Options	93
Performing Online Backup	94
Exiting the Volume Manager Support Tasks	96
Online Relayout	97
Storage Layout	97
How Online Relayout Works	98
Types of Transformation	98
Transformation Characteristics	101
Transformations and Volume Length	101
Transformations Not Supported	102
Hot-Relocation	103
How Hot-Relocation Works	103
How Space is Chosen for Relocation	104
Modifying the vxrelocd Command	105
vxunreloc Unrelocate Utility	105
Volume Resynchronization	110
Dirty Region Logging	112
FastResync (Fast Mirror Resynchronization)	114
FMR Components	114
FMR Enhancements to VxVM Snapshot Functionality	115
FMR Enhancements to VxVM Snapshot Functionality	119
Volume Manager Rootability	120
Booting With Root Volumes	120
Boot Time Volume Restrictions	121
Dynamic Multipathing (DMP)	122
Path Failover Mechanism	122
Load Balancing	123
Disabling and Enabling DMP	123
Input/Output (I/O) Controllers	124
Displaying DMP Database Information	125
VxSmartSync Recovery Accelerator	126
Data Volume Configuration	126
Redo Log Volume Configuration	127

Common Volume Manager Commands	128
vxedit Command	128
vxtask Utility	128
vxassist Command	131
vxdctl Daemon	135
vxmake Command	136
vxplex Command	137
vxsd Command	137
vxmend Command	138
vxprint Command	138
vxrelocd Command	138
vxstat Command	140
vxtrace Command	140
vxunrelocate Command	140
vxvol Command	143

4. Disk Tasks

Introduction	146
Disk Devices	148
Disk Utilities	150
vxdiskadm Main Menu	152
vxdiskadm Menu Description	153
Initializing Disks	157
Formatting the Disk Media	157
Volume Manager Disk Installation	157
Adding a Disk to Volume Manager	160
Placing Disks Under Volume Manager Control	163
Placing a Formatted Disk Under VM Control	164
Placing Multiple Disks Under Volume Manager Control	167
Moving Disks	172
Enabling a Physical Disk	173
Detecting Failed Disks	175

Partial Disk Failure	175
Complete Disk Failure	176
Disabling a Disk	178
Replacing a Disk	179
Replacing a Failed or Removed Disk	180
Removing Disks	182
Removing a Disk with No Subdisks	183
Removing a Disk with Subdisks	184
Removing a Disk as a Hot-Relocation Spare	185
Taking a Disk Offline	186
Adding a Disk to a Disk Group	187
Adding a VM Disk to the Hot-Relocation Pool	190
Designating a Disk from the Command Line	190
Designating a Disk with vxdiskadm	190
Removing a VM Disk From the Hot-Relocation Pool	192
Removing a Disk with vxdiskadm	192
Excluding a Disk from Hot-Relocation Use	194
Including a Disk for Hot-Relocation Use	195
Reinitializing a Disk	197
Renaming a Disk	200
Reserving Disks	201
Displaying Disk Information	202
Displaying Multipaths Under a VM Disk	202
Displaying Multipathing Information	203
Displaying Disk Information with the vxdiskadm Program	205

5. Disk Group Tasks

Introduction	208
Disk Groups	209
Disk Group Utilities	211

Creating a Disk Group	212
Renaming a Disk Group	216
Importing a Disk Group	218
Deporting a Disk Group	220
Upgrading a Disk Group	222
Moving a Disk Group	223
Moving Disk Groups Between Systems	224
Using Disk Groups	227
Removing a Disk Group	228
Destroying a Disk Group	229
Reserving Minor Numbers for Disk Groups	230
Displaying Disk Group Information	231

6. Volume Tasks

Introduction	234
Volume, Plex, and Subdisk Tasks	234
Creating a Volume	237
Creating a Concatenated Volume	238
Creating a Striped Volume	239
Creating a RAID-5 Volume	240
Creating a Mirrored Volume	240
Starting a Volume	241
Listing Unstartable Volumes	241
Stopping a Volume	242
Resizing a Volume	243
Resizing Volumes With the vxassist Command	243
Resizing Volumes with the vxvol Command	244
Resizing Volumes with the vxresize Command	246
Removing a Volume	247
Mirroring a Volume	248

Creating a Volume with Dirty Region Logging Enabled	248
Mirroring an Existing Volume	248
Mirroring All Volumes	249
Mirroring Volumes on a VM Disk	249
Backing Up Volumes Using Mirroring	251
Removing a Mirror	252
Displaying Volume Configuration Information	254
Preparing a Volume to Restore From Backup	255
Recovering a Volume	257
Moving Volumes from a VM Disk	258
Adding a RAID-5 Log	260
Removing a RAID-5 Log	261
Adding a DRL Log	262
Removing a DRL Log	263
Creating Plexes	264
Creating a Striped Plex	264
Associating Plexes	265
Dissociating and Removing Plexes	266
Displaying Plex Information	267
Changing Plex Attributes	268
Changing Plex Status: Detaching and Attaching Plexes	270
Detaching Plexes	270
Attaching Plexes	271
Moving Plexes	273
Copying Plexes	274
Creating Subdisks	275
Removing Subdisks	276
Moving Subdisks	277
Moving Relocated Subdisks	277

Moving Hot-Relocate Subdisks Back to a Disk	278
Splitting Subdisks	280
Joining Subdisks	281
Associating Subdisks	282
Associating Log Subdisks.....	283
Dissociating Subdisks	285
Changing Subdisk Attributes	286
Performing Online Backup	287
FastResync (Fast Mirror Resynchronization)	289
Mirroring Volumes on a VM Disk	291
Moving Volumes from a VM Disk	292

7. Cluster Functionality

Introduction	296
Cluster Functionality Overview	297
Shared Volume Manager Objects.....	297
How Cluster Volume Management Works	298
Disks in VxVM Clusters	306
Disk Detach Policies	306
Disk Group Activation	306
Dirty Region Logging and Cluster Environments	308
Log Format and Size.....	308
Compatibility	309
How DRL Works in a Cluster Environment	310
Dynamic Multipathing (DMP).....	311
Disabling and Enabling Controllers	311
Assigning a User Friendly Name.....	312
Display DMP Nodes	312
Display Subpaths	312
List Controllers.....	313
List Enclosure.....	313
DMP Restore Daemon	313

FastResync (Fast Mirror Resynchronization)	314
Upgrading Volume Manager Cluster Functionality	316
Cluster-related Volume Manager Utilities and Daemons	318
vxclustd Daemon	318
vxconfigd Daemon	319
vxdctl Utility	322
vxdg Utility	323
vxdisk Utility	327
vxdmpadm Utility	328
vxrecover Utility	329
vxstat Utility	329
Cluster Terminology	331

8. Recovery

Introduction	334
Reattaching Disks	335
VxVM Boot Disk Recovery	336
Reinstallation Recovery	337
General Reinstallation Information	337
Reinstallation and Reconfiguration Procedures	338
Detecting and Replacing Failed Disks	347
Hot-Relocation	347
Moving Hot-Relocated Subdisks	351
Detecting Failed Disks	354
Replacing Disks	356
Plex and Volume States	358
Plex States	358
The Plex State Cycle	361
Plex Kernel State	361
Volume States	362
Volume Kernel State	363
RAID-5 Volume Layout	364
RAID-5 Plexes	364

RAID-5 Logs	364
Creating RAID-5 Volumes.....	366
vxassist Command and RAID-5 Volumes	366
vxmake Command and RAID-5 Volumes.....	366
Initializing RAID-5 Volumes.....	368
Failures and RAID-5 Volumes.....	369
System Failures	369
Disk Failures.....	370
RAID-5 Recovery	372
Parity Recovery.....	373
Subdisk Recovery	374
Recovering Logs After Failures	374
Miscellaneous RAID-5 Operations	375
Manipulating RAID-5 Logs	375
Manipulating RAID-5 Subdisks.....	376
Starting RAID-5 Volumes.....	377
Unstartable RAID-5 Volumes	378
Forcibly Starting RAID-5 Volumes	379
Recovery When Starting RAID-5 Volumes	380
Changing RAID-5 Volume Attributes	382
Writing to RAID-5 Arrays	382

9. Performance Monitoring

Introduction	390
Performance Guidelines:	391
Data Assignment	391
Striping	391
Mirroring.....	392
Mirroring and Striping.....	394
Striping and Mirroring.....	394
Using RAID-5	395
Performance Monitoring	396
Performance Priorities	396

Getting Performance Data	396
Using Performance Data	398
Tuning the Volume Manager.....	402
General Tuning Guidelines	402
Tunables	402
Tuning for Large Systems	408

Glossary

Preface

Introduction

The *VERITAS Volume Manager 3.1 Administrator's Guide* provides information on how to use Volume Manager.

Audience and Scope

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

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 with the following chapters:

- Chapter 1, “Introduction to Volume Manager”
- Chapter 2, “Initialization and Setup”
- Chapter 3, “Volume Manager Operations”
- Chapter 4, “Disk Tasks”
- Chapter 5, “Disk Group Tasks”
- Chapter 6, “Volume Tasks”
- Chapter 7, “Cluster Functionality”
- Chapter 8, “Recovery”
- Chapter 9, “Performance Monitoring”
- “Glossary”

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:

- *VERITAS Volume Manager 3.1 Migration Guide*
- *VERITAS Volume Manager 3.1 Reference Guide*
- *VERITAS Volume Manager 3.1 Storage Administrator Administrator's Guide*
- *VERITAS Volume Manager 3.1 for HP-UX Release Notes*
- VERITAS Volume Manager manual pages

Conventions

We use the following typographical conventions.

<i>audit</i> (5)	An HP-UX manpage. <i>audit</i> is the name and 5 is the section in the <i>HP-UX Reference</i> . On the web and on the Instant Information CD, it may be a hot link to the manpage itself. From the HP-UX command line, you can enter “man audit” or “man 5 audit” to view the manpage. See <i>man</i> (1).
<i>Book Title</i>	The title of a book. On the web and on the Instant Information CD, it may be a hot link to the book itself.
KeyCap	The name of a keyboard key. Note that Return and Enter both refer to the same key.
<i>Emphasis</i>	Text that is emphasized.
Emphasis	Text that is strongly emphasized.
Term	The defined use of an important word or phrase.
ComputerOut	Text displayed by the computer.
UserInput	Commands and other text that you type.
Command	A command name or qualified command phrase.
<i>Variable</i>	The name of a variable that you may replace in a command or function or information in a display that represents several possible values.
[]	The contents are optional in formats and command descriptions. If the contents are a list separated by , choose one of the items.
{ }	The contents are required in formats and command descriptions. If the contents are a list separated by , you must choose one of the items.
...	The preceding element may be repeated an arbitrary number of times.
	Separates litems in a list of choices.

1 Introduction to Volume Manager

Introduction

This chapter describes what VERITAS Volume Manager is, how it works, how you can communicate with it through the user interfaces, and Volume Manager concepts. Related documents that provide specific information about Volume Manager are listed in the Preface.

The following topics are covered in this chapter:

- “How Data is Stored”
- “Volume Manager Overview”
- “Physical Objects”
- “Volumes and Virtual Objects”
- “Virtual Object Data Organization (Volume Layouts)”
- “Volume Manager and RAID-5”
- “Layered Volumes”
- “Volume Manager User Interfaces”
- “Volume Manager Conceptual Overview”
- “Volume Administration Tasks”

Volume Manager provides easy-to-use online disk storage management for computing environments. Traditional disk storage management often requires that machines be taken off-line at a major inconvenience to users. In the distributed client/server environment, databases and other resources must maintain high availability, be easy to access, and be Volume Manager provides the tools to improve performance and ensure data availability and integrity. Volume Manager also *dynamically configures disk storage while the system is active*.

This chapter introduces the VERITAS Volume Manager concepts and describes the tools that Volume Manager uses to perform storage management.

NOTE

Rootability or bringing the root disk under VxVM control is not supported on HP-UX 11i systems, but it is supported on HP-UX 11i

Version 1.5 systems.

How Data is Stored

There are several methods used to store data on physical disks. These methods organize data on the disk so the data can be stored and retrieved efficiently. The basic method of disk organization is called **formatting**. Formatting prepares the hard disk so that files can be written to and retrieved from the disk by using a prearranged storage pattern.

Hard disks are formatted, and information stored, in two ways: physical-storage layout and logical-storage layout. Volume Manager uses the **logical-storage layout** method. The types of storage layout supported by Volume Manager are introduced in this chapter and described in detail in “Storage Layout”.

Volume Manager Overview

The Volume Manager uses **objects** to do storage management. The two types of objects used by Volume Manager are **physical objects** and **virtual objects**.

- physical objects

Volume Manager uses two *physical objects*: physical disks and partitions. Partitions are created on the physical disks (on systems that use partitions).

- virtual objects

Volume Manager creates **virtual objects**, called **volumes**. Each volume records and retrieves data from one or more physical disks. Volumes are accessed by a file system, a database, or other applications in the same way that physical disks are accessed. Volumes are also composed of other virtual objects that are used to change the volume configuration. Volumes and their virtual components are called **virtual objects**.

Physical Objects

This section describes the physical objects (physical disks and partitions) used by Volume Manager.

Physical Disks and Disk Naming

A **physical disk** is the basic storage device (media) where the data is ultimately stored. You can access the data on a physical disk by using a **device name (devname)** to locate the disk. The physical disk device name varies with the computer system you use. Not all parameters are used on all systems. Typical device names can include: `c#t#d#`, where:

- `c#` is the controller
- `t#` is the target ID
- `d#` is the disk number

On an HP-UX 11i Version 1.5 system, the HP-UX partition number on the boot disk drive is: `c#t#d#s2`, where:

- `s2` is the slice number

Partitions

On some computer systems, a physical disk can be divided into one or more **partitions**. The **partition number**, or `s#`, is added at the end of the device name. Note that a partition can be an entire physical disk.

For HP-UX 11i, all the disks (except the root disk) are treated and accessed by the Volume Manager as entire physical disks using a device name such as `c#t#d#`. On HP-UX 11i Version 1.5, the root disk is on partition 2, `c#t#d#s2`.

Volumes and Virtual Objects

The connection between physical objects and Volume Manager objects is made when you place a physical disk under Volume Manager control.

Volume Manager creates **virtual objects** and makes logical connections between the objects. The virtual objects are then used by Volume Manager to do storage management tasks.

NOTE

The `vxprint` command displays detailed information on existing Volume Manager objects. For additional information on the `vxprint` command, see “Displaying Volume Manager Object Listings”.

A **volume** is a virtual disk device that appears to applications, databases, and file systems as a physical disk. However, a volume does not have the limitations of a physical disk. When you use Volume Manager, applications access volumes created on Volume Manager disks (VM Disks) rather than physical disks.

Volumes contain other virtual objects that you can use to manipulate data within a volume. The virtual objects contained in volumes are: VM disks, disk groups, subdisks, and plexes. Details of the virtual objects are described in the following sections. Virtual objects and volume manipulation of them are described in “Volumes”.

Volume Manager Disks

When you place a physical disk under Volume Manager control, a Volume Manager disk (or VM Disk) is assigned to the physical disk. A VM Disk is under Volume Manager control and is usually in a disk group. Each VM disk corresponds to at least one physical disk. Volume Manager allocates storage from a contiguous area of Volume Manager disk space.

A VM disk typically includes a **public region** (allocated storage) and a **private region** where Volume Manager internal configuration information is stored.

Each VM Disk has a unique **disk media name** (a virtual disk name). You can supply the disk name or allow Volume Manager to assign a

default name that typically takes the form `disk##`. See Figure 1-1, Example of a VM Disk, which shows a VM disk with a media name of `disk01` that is assigned to the disk **devname**.

Figure 1-1

Example of a VM Disk



Disk Groups

A **disk group** is a collection of VM disks that share a common configuration. A disk group configuration is a set of records with detailed information about related Volume Manager objects, their attributes, and their connections. The default disk group is `rootdg` (the root disk group).

You can create additional disk groups as necessary. Disk groups allow the administrator to group disks into logical collections. A disk group and its components can be moved as a unit from one host machine to another.

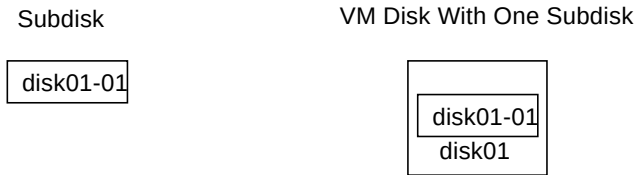
Volumes are created within a disk group. A given volume must be configured from disks in the same disk group.

Subdisks

A **subdisk** is a set of contiguous disk blocks. A block is a unit of space on the disk. Volume Manager allocates disk space using subdisks. A VM disk can be divided into one or more subdisks. Each subdisk represents a specific portion of a VM disk, which is mapped to a specific region of a physical disk.

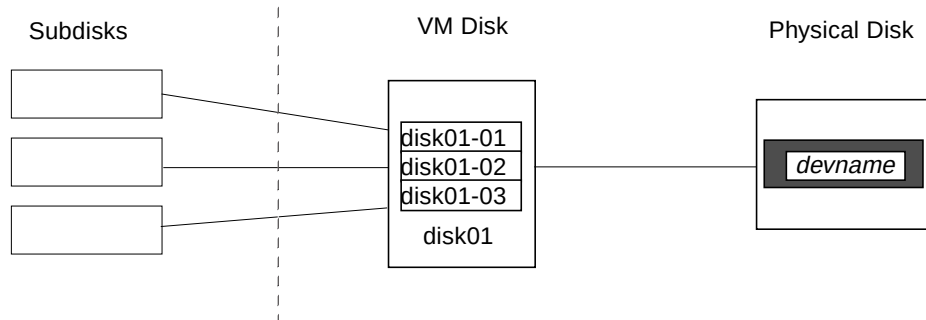
The default name for a VM disk is `disk##` (such as `disk01`) and the default name for a subdisk is `disk##-##`. As shown in Figure 1-2, Example of a Subdisk, `disk01-01` is the name of the first subdisk on the VM disk named `disk01`.

Figure 1-2 Example of a Subdisk



A VM disk can contain multiple subdisks, but subdisks cannot overlap or share the same portions of a VM disk. See Figure 1-3, Example of Three Subdisks Assigned to One VM Disk,

Figure 1-3 Example of Three Subdisks Assigned to One VM Disk



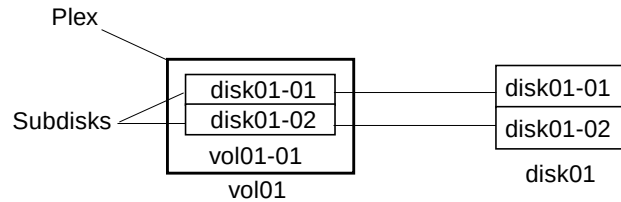
Any VM disk space that is not part of a subdisk is free space. You can use free space to create new subdisks.

Volume Manager release 3.0 or higher supports the concept of layered volumes in which subdisk objects can contain volumes. For more information, see "Layered Volumes".

Plexes

The Volume Manager uses subdisks to build virtual objects called **plexes**. A plex consists of one or more subdisks located on one or more physical disks. See Figure 1-4, Example Plex With Two Subdisks,.

Figure 1-4 **Example Plex With Two Subdisks**



You can organize data on the subdisks to form a plex by using these methods:

- concatenation
- striping (RAID-0)
- striping with parity (RAID-5)
- mirroring (RAID-1)

Concatenation, striping (RAID-0), RAID-5, and mirroring (RAID-1) are described in “Virtual Object Data Organization (Volume Layouts)”.

NOTE

You may need an additional license to use this feature.

Volumes

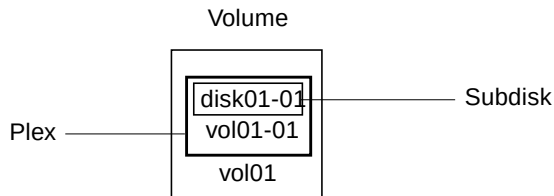
A **volume** is a virtual disk device that appears to applications, databases, and file systems like a physical disk device, but does not have the physical limitations of a physical disk device. A volume consists of one or more plexes, each holding a copy of the selected data in the volume. Due to its virtual nature, a volume is not restricted to a particular disk or a specific area of a disk. The configuration of a volume can be changed by using the Volume Manager user interfaces. Configuration changes can be done without causing disruption to applications or file systems that are using the volume. For example, a volume can be mirrored on separate disks or moved to use different disk storage.

The Volume Manager uses the default naming conventions of `vol###` for volumes and `vol###-##` for plexes in a volume. Administrators must

select meaningful names for their volumes.

A volume can consist of up to 32 plexes, each of which contains one or more subdisks. A volume must have at least one associated plex that has a complete set of the data in the volume with at least one associated subdisk. Note that all subdisks within a volume must belong to the same disk group. See Figure 1-5, Example of a Volume with One Plex,.

Figure 1-5 **Example of a Volume with One Plex**

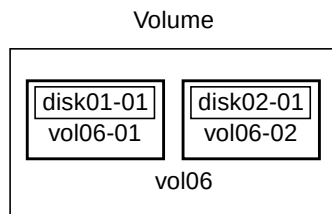


Volume vol01 in Figure 1-5, Example of a Volume with One Plex, has the following characteristics:

- it contains one plex named vol01-01
- the plex contains one subdisk named disk01-01
- the subdisk disk01-01 is allocated from VM disk disk01

A volume with two or more data plexes is “mirrored” and contains mirror images of the data. See Figure 1-6, Example of a Volume with Two Plexes,

Figure 1-6 **Example of a Volume with Two Plexes**



Each plex contains an identical copy of the volume data. For more information, see “Mirroring (RAID-1)”.

Volume vol06 in Figure 1-6, Example of a Volume with Two Plexes, has the following characteristics:

- it contains two plexes named vol06-01 and vol06-02
- each plex contains one subdisk
- each subdisk is allocated from a different VM disk (disk01 and disk02)

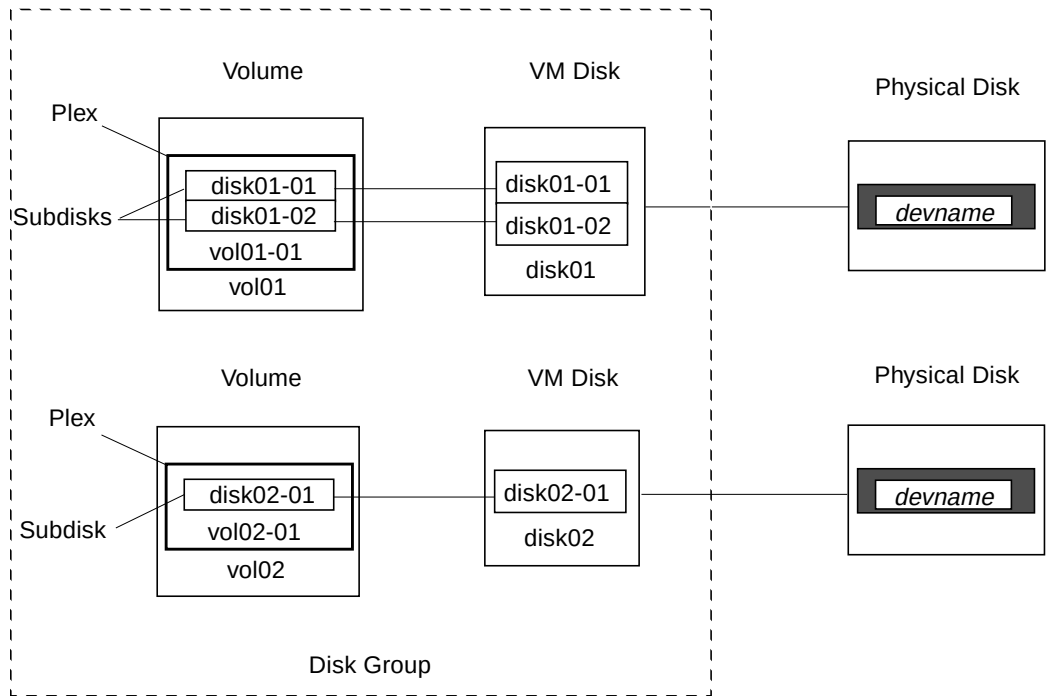
Connection Between Volume Manager Virtual Objects

Volume Manager virtual objects are combined to build volumes. The virtual objects contained in volumes are: VM disks, disk groups, subdisks, and plexes. Volume Manager objects have the following connections:

- Volume Manager disks are grouped into disk groups
- one or more subdisks (each representing a specific region of a disk) are combined to form plexes
- a volume is composed of one or more plexes

The example in Figure 1-7, Connection Between Volume Manager Objects, shows the connections between Volume Manager virtual objects and how they relate to physical disks. Figure 1-7, Connection Between Volume Manager Objects, shows a disk group with two VM disks (disk01 and disk02). In disk01, there is a volume with one plex and two subdisks. In disk02, there is a volume with one plex and a single subdisk.

Figure 1-7 **Connection Between Volume Manager Objects**



Virtual Object Data Organization (Volume Layouts)

Data in virtual objects is organized to create volumes by using the following layout methods:

- concatenation
- striping (RAID-0)
- RAID-5 (striping with parity)
- mirroring (RAID-1)
- mirroring plus striping
- striping plus mirroring

The following sections describe each layout method.

Related Graphical User Interface (GUI) Terms

The following graphical user interface (GUI) terms refer to types of layered volumes created on the Storage Administrator:

- *Concatenated Pro*—a layered concatenated volume that is mirrored
- *Striped Pro*—a layered striped volume that is mirrored

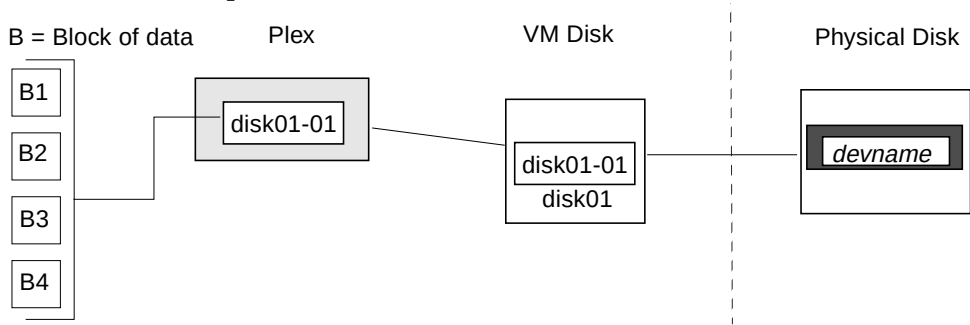
Concatenation

Concatenation maps data in a linear manner onto one or more subdisks in a plex. To access all the data in a concatenated plex sequentially, data is first accessed in the first subdisk from beginning to end. Data is then accessed in the remaining subdisks sequentially from beginning to end until the end of the last subdisk.

The subdisks in a concatenated plex do not have to be physically contiguous and can belong to more than one VM disk. Concatenation using subdisks that reside on more than one VM disk is called **spanning**.

Figure 1-8, Example of Concatenation, shows concatenation with one subdisk.

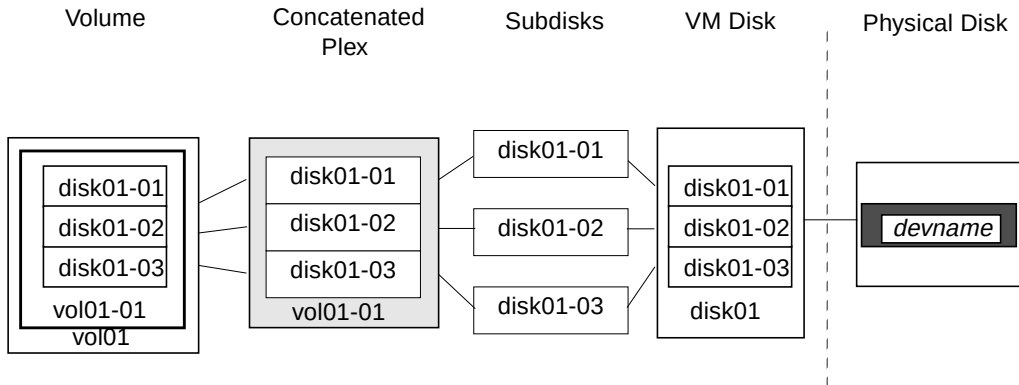
Figure 1-8 Example of Concatenation



You can use concatenation with multiple subdisks when there is insufficient contiguous space for the plex on any one disk. This form of concatenation can be used for load balancing between disks, and for head movement optimization on a particular disk.

For an example of a volume in a concatenated configuration, see Figure 1-9, Example of a Volume in a Concatenated Configuration,

Figure 1-9 Example of a Volume in a Concatenated Configuration



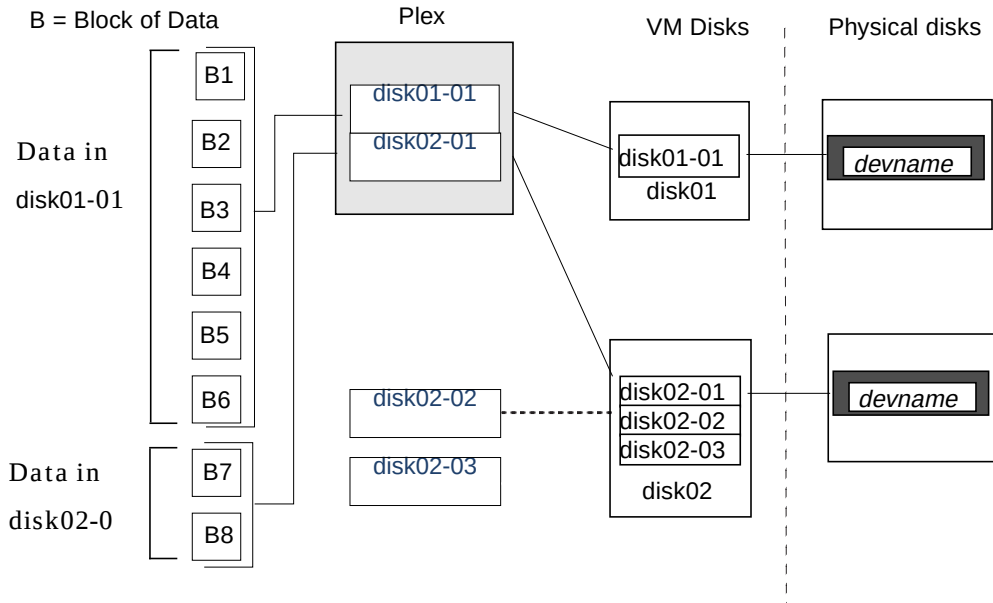
In the example shown in Figure 1-10, Example of Spanning,, the first six blocks of data (B1 through B6) use most of the space on the disk that VM disk disk01 is assigned to. This requires space only on subdisk disk01-01 on VM disk disk01. However, the last two blocks of data, B7 and B8, use only a portion of the space on the disk that VM disk disk02 is assigned to.

The remaining free space on VM disk disk02 can be put to other uses. In

this example, subdisks disk02-02 and disk02-03 are available for other disk management tasks.

Figure 1-10, Example of Spanning, shows data spread over two subdisks in a spanned plex.

Figure 1-10 Example of Spanning



CAUTION

Spanning a plex across multiple disks increases the chance that a disk failure results in failure of the assigned volume. Use mirroring or RAID-5 (both described later) to reduce the risk that a single disk failure results in a volume failure.

Striping (RAID-0)

Striping (RAID-0) maps data so that the data is interleaved among two or more physical disks. A striped plex contains two or more subdisks, spread out over two or more physical disks. Data is allocated alternately and evenly to the subdisks of a striped plex.

The subdisks are grouped into “columns,” with each physical disk limited

to one column. Each column contains one or more subdisks and can be derived from one or more physical disks. The number and sizes of subdisks per column can vary. Additional subdisks can be added to columns, as necessary.

CAUTION

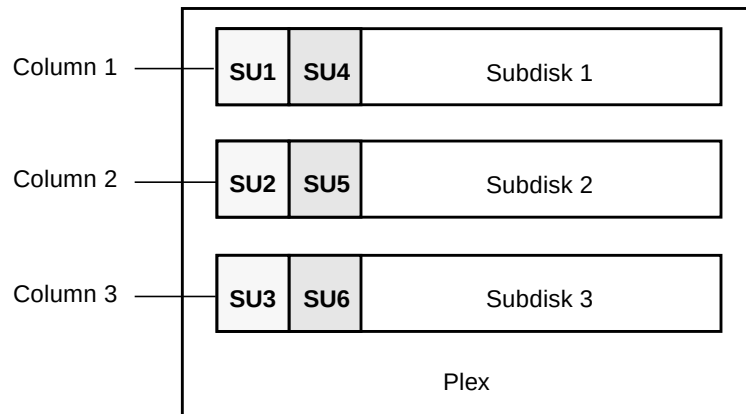
Striping a volume, or splitting a volume across multiple disks, increases the chance that a disk failure will result in failure of that volume. For example, if five volumes are striped across the same five disks, then failure of any one of the five disks will require that all five volumes be restored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

Data is allocated in equal-sized units (**stripe units of size** called **stripe unit size**) that are interleaved between the columns. Each stripe unit is a set of contiguous blocks on a disk. The default stripe unit size is 64 kilobytes.

For example, if there are three columns in a striped plex and six stripe units, data is striped over three physical disks, as shown in Figure 1-11, *Striping Across Three Disks (Columns)*. In Figure 1-11, *Striping Across Three Disks (Columns)*,:

- the first and fourth stripe units are allocated in column 1
- the second and fifth stripe units are allocated in column 2
- the third and sixth stripe units are allocated in column 3

Figure 1-11 Striping Across Three Disks (Columns)



SU = Stripe Unit

A **stripe** consists of the set of stripe units at the same positions across all columns. In Figure 1-11, Striping Across Three Disks (Columns),, stripe units 1, 2, and 3 constitute a single stripe.

Viewed in sequence, the first stripe consists of:

- stripe unit 1 in column 1
- stripe unit 2 in column 2
- stripe unit 3 in column 3

The second stripe consists of:

- stripe unit 4 in column 1
- stripe unit 5 in column 2
- stripe unit 6 in column 3

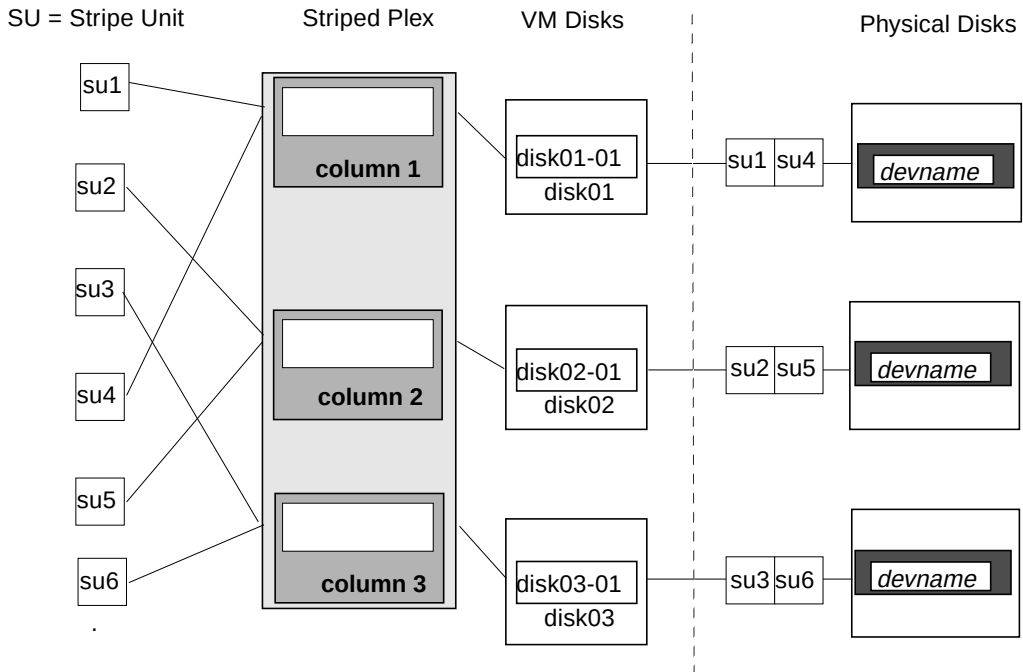
Striping continues for the length of the columns (if all columns are the same length) or until the end of the shortest column is reached. Any space remaining at the end of subdisks in longer columns becomes unused space.

Striping is useful if you need large amounts of data written to or read from the physical disks quickly by using parallel data transfer to multiple disks. Striping is also helpful in balancing the I/O load from

multi-user applications across multiple disks.

Figure 1-12, Example of a Striped Plex with One Subdisk per Column, shows a striped plex with three equal-sized, single-subdisk columns. There is one column per physical disk.

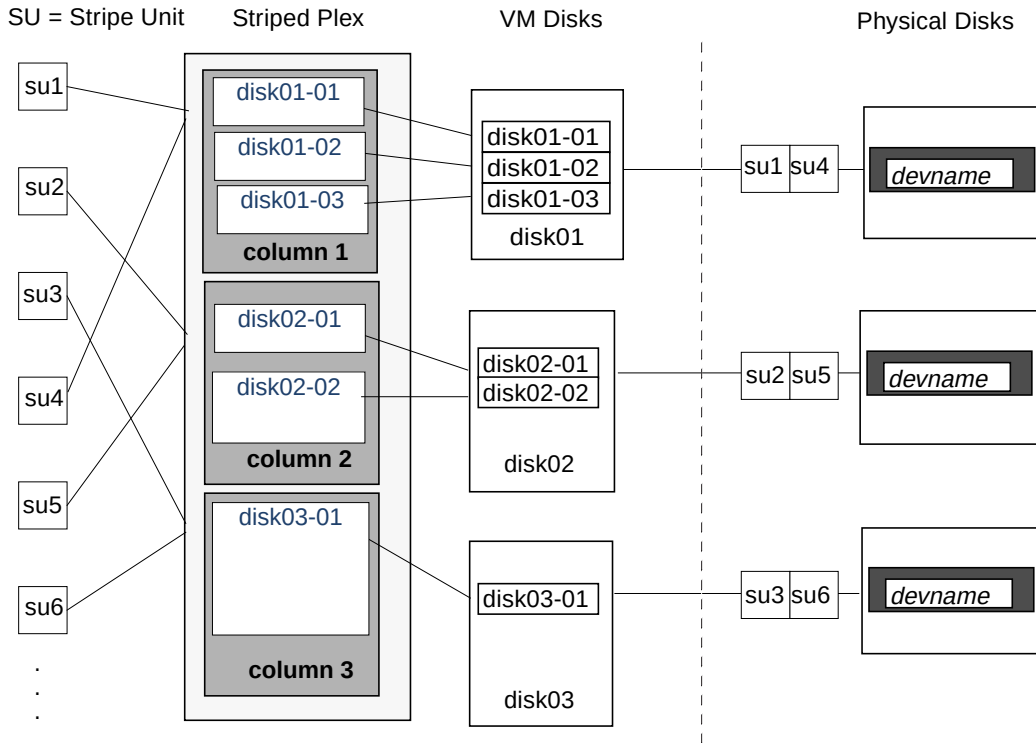
Figure 1-12 Example of a Striped Plex with One Subdisk per Column



The example in Figure 1-12, Example of a Striped Plex with One Subdisk per Column, shows three subdisks that occupy all of the space on the VM disks. It is also possible for each subdisk in a striped plex to occupy only a portion of the VM disk, which leaves free space for other disk management tasks.

Figure 1-13, Example of a Striped Plex with Concatenated Subdisks per Column, shows a striped plex with three columns containing subdisks of different sizes. Each column contains a different number of subdisks. There is one column per physical disk. Striped plexes can be created by using a single subdisk from each of the VM disks being striped across. It is also possible to allocate space from different regions of the same disk or from another disk (for example, if the plex is grown). Columns can contain subdisks from different VM disks if necessary.

Figure 1-13 Example of a Striped Plex with Concatenated Subdisks per Column



RAID-5

NOTE

You may need an additional license to use this feature.

RAID-5 provides data redundancy by using **parity**. Parity is a calculated value used to reconstruct data after a failure. While data is being written to a RAID-5 volume, parity is calculated by doing an **exclusive OR** (XOR) procedure on the 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 parity information.

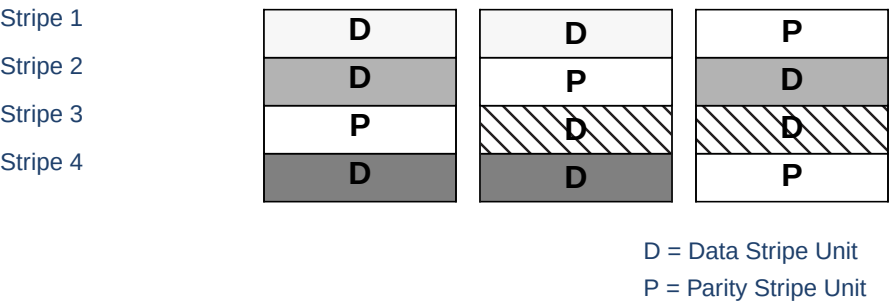
RAID-5 volumes maintain redundancy of the data within a volume.

RAID-5 volumes keep a copy of the data and calculated parity in a plex that is “striped” across multiple disks. In the event of a disk failure, a RAID-5 volume uses parity to reconstruct the data. It is possible to mix concatenation and striping in the layout.

RAID-5 volumes can do logging to minimize recovery time. RAID-5 volumes use RAID-5 logs to keep a copy of the data and parity currently being written. RAID-5 logging is optional and can be created along with RAID-5 volumes or added later.

Figure 1-14, Parity Locations in a RAID-5 Model, 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 1-14 Parity Locations in a RAID-5 Model



For more information, see “Volume Manager and RAID-5”.

Mirroring (RAID-1)

NOTE

You may need an additional license to use this feature.

Mirroring uses multiple mirrors (plexes) to duplicate the information contained in a volume. In the event of a physical disk failure, the plex on the failed disk becomes unavailable, but the system continues to operate using the unaffected mirrors. Although a volume can have a single plex, at least two plexes are required to provide redundancy of data. Each of these plexes must contain disk space from different disks to achieve

redundancy.

When striping or spanning across a large number of disks, failure of any one of those disks can make the entire plex unusable. The chance of one out of several disks failing is sufficient to make it worthwhile to consider mirroring in order to improve the reliability (and availability) of a striped or spanned volume.

Mirroring Plus Striping (RAID-1 + RAID-0)

NOTE

You may need an additional license to use this feature.

The Volume Manager supports the combinations of mirroring plus striping. When used together on the same volume, mirroring plus striping offers the benefits of spreading data across multiple disks (striping) while providing redundancy (mirror) of data.

For mirroring plus striping to be effective when used together, the mirror and its striped plex must be allocated from separate disks. The layout type of the mirror can be concatenated or striped.

Striping Plus Mirroring (RAID-0 + RAID-1)

NOTE

You may need an additional license to use this feature.

The Volume Manager supports the combination of striping with mirroring. In previous releases, whenever mirroring was used, the mirroring had to happen above striping. Now there can be mirroring both above and below striping.

By putting mirroring below striping, each column of the stripe is mirrored. If the stripe is large enough to have multiple subdisks per column, each subdisk can be individually mirrored. This layout enhances redundancy and reduces recovery time in case of an error.

In a mirror-stripe layout, if a disk fails, the entire plex is detached, thereby losing redundancy on the entire volume. When the disk is replaced, the entire plex must be brought up to date. Recovering the entire plex can take a substantial amount of time. If a disk fails in a

stripe-mirror layout, only the failing subdisk must be detached, and only that portion of the volume loses redundancy. When the disk is replaced, only a portion of the volume needs to be recovered.

Compared to mirroring plus striping, striping plus mirroring offers a volume more tolerant to disk failure. If a disk failure occurs, the recovery time is shorter for striping plus mirroring. See “Layered Volumes” for more information.

Volume Manager and RAID-5

NOTE

You may need an additional license to use this feature.

This section describes how Volume Manager implements RAID-5. For general information on RAID-5, see “RAID-5”.

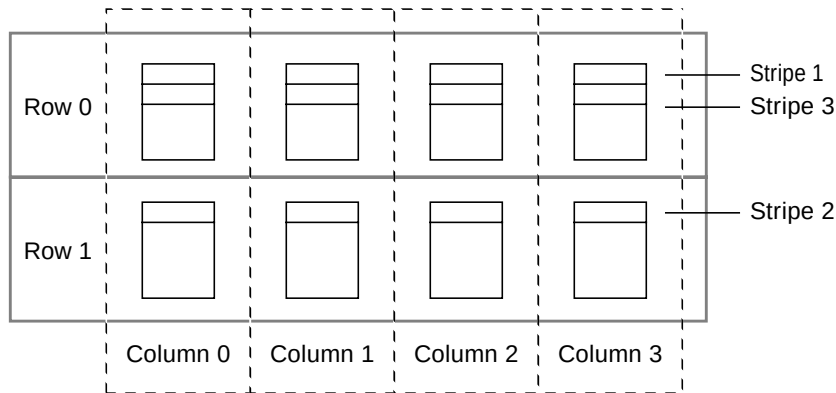
Although both mirroring (RAID-1) and RAID-5 provide redundancy of data, they use different methods. Mirroring provides data redundancy by maintaining multiple complete copies of the data in a volume. Data being written to a mirrored volume is reflected in all copies. If a portion of a mirrored volume fails, the system continues to use the other copies of the data.

RAID-5 provides data redundancy by using parity. Parity is a calculated value used to reconstruct data after a failure. While data is being written to a RAID-5 volume, parity is calculated by doing an XOR procedure on the 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 parity information.

Traditional RAID-5 Arrays

A **traditional** RAID-5 array is several disks organized in rows and columns. A **column** is a number of disks located in the same ordinal position in the array. A **row** is the minimal number of disks necessary to support the full width of a parity stripe. Figure 1-15, Traditional RAID-5 Array, shows the row and column arrangement of a traditional RAID-5 array.

Figure 1-15 **Traditional RAID-5 Array**

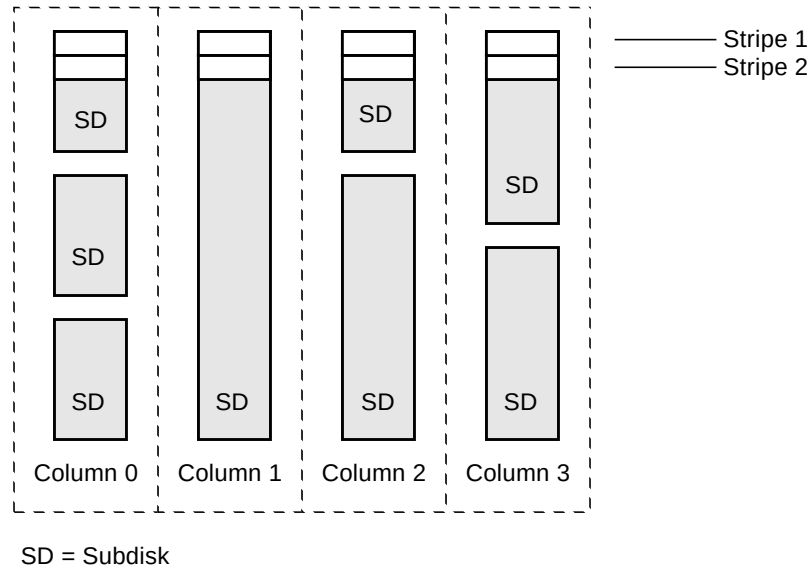


This traditional array structure supports growth by adding more rows per column. Striping is accomplished by applying the first stripe across the disks in Row 0, then the second stripe across the disks in Row 1, then the third stripe across the Row 0 disks, and so on. This type of array requires all disks (partitions), columns, and rows to be of equal size.

Volume Manager RAID-5 Arrays

The Volume Manager RAID-5 array structure differs from the traditional structure. Due to the virtual nature of its disks and other objects, the Volume Manager does not use rows. Instead, the Volume Manager uses columns consisting of variable length subdisks, as shown in Figure 1-16, Volume Manager RAID-5 Array,. Each subdisk represents a specific area of a disk.

Figure 1-16 **Volume Manager RAID-5 Array**



With the Volume Manager RAID-5 array structure, each column can consist of a different number of subdisks. The subdisks in a given column can be derived from different physical disks. Additional subdisks can be added to the columns as necessary. Striping (see “Striping (RAID-0)”) is done by applying the first stripe across each subdisk at the top of each column, then another stripe below that, and so on for the length of the columns. For each stripe, an equal-sized stripe unit is placed in each column. With RAID-5, the default stripe unit size is 16 kilobytes.

NOTE

Mirroring of RAID-5 volumes is not currently supported.

Left-Symmetric Layout

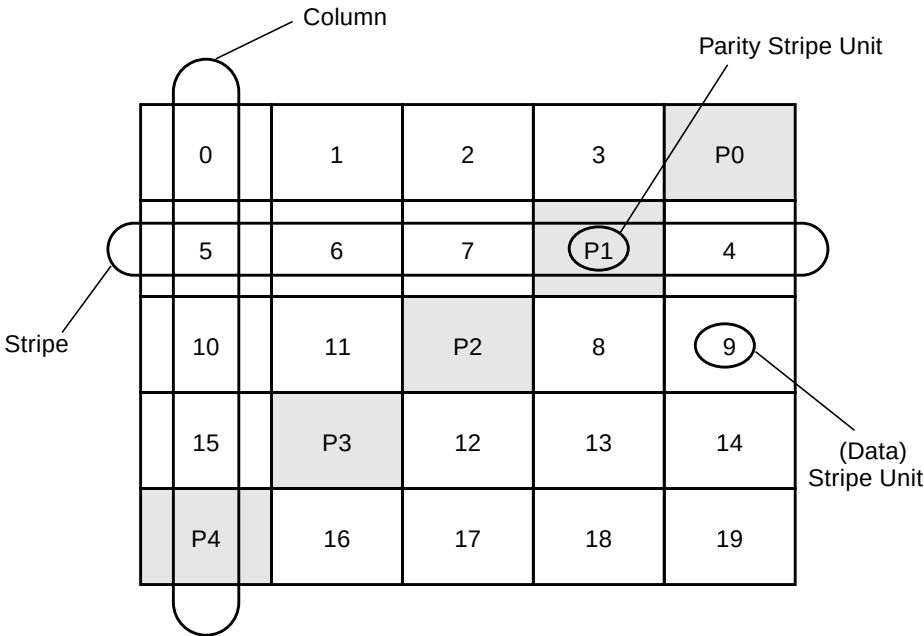
There are several layouts for data and parity that can be used in the setup of a RAID-5 array. The Volume Manager implementation of RAID-5 is the left-symmetric layout. The left-symmetric parity layout provides optimal performance for both random I/O operations and large sequential I/O operations. In terms of performance, the layout selection

is not as critical as the number of columns and the stripe unit size selection.

The left-symmetric layout stripes both data and parity across columns, placing the parity in a different column for every stripe of data. The first parity stripe unit is located in the rightmost column of the first stripe. Each successive parity stripe unit is located in the next stripe, shifted left one column from the previous parity stripe unit location. If there are more stripes than columns, the parity stripe unit placement begins in the rightmost column again.

Figure 1-17, Left-Symmetric Layout, shows a left-symmetric parity layout with five disks (one per column).

Figure 1-17 Left-Symmetric Layout



For each stripe, data is organized starting to the right of the parity stripe unit. In Figure 1-17, Left-Symmetric Layout,, data organization for the first stripe begins at P0 and continues to stripe units 0-3. Data organization for the second stripe begins at P1, then continues to stripe unit 4, and on to stripe units 5-7. Data organization proceeds in this manner for the remaining stripes.

Each parity stripe unit contains the result of an exclusive OR (XOR) procedure done on the data in the data stripe units within the same stripe. If data on a disk corresponding to one column is inaccessible due to hardware or software failure, data can be restored. Data is restored by XORing the contents of the remaining columns data stripe units against their respective parity stripe units (for each stripe).

For example, if the disk corresponding to the far left column in Figure 1-17, Left-Symmetric Layout, fails, the volume is placed in a degraded mode. While in degraded mode, the data from the failed column can be recreated by XORing stripe units 1-3 against parity stripe unit P0 to recreate stripe unit 0, then XORing stripe units 4, 6, and 7 against parity stripe unit P1 to recreate stripe unit 5, and so on.

NOTE

Failure of multiple columns in a plex with a RAID-5 layout detaches the volume. The volume is no longer allowed to satisfy read or write requests. Once the failed columns have been recovered, it may be necessary to recover the user data from backups.

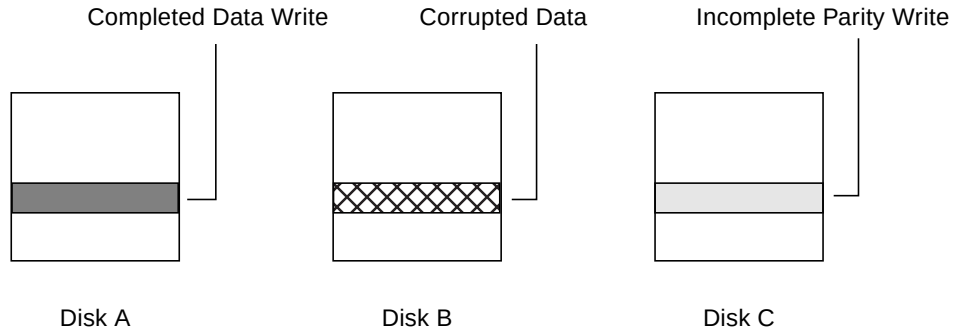
Logging

Logging (recording) is used to prevent corruption of recovery data. A log of the new data and parity is made on a persistent device (such as a disk-resident volume or non-volatile RAM). The new data and parity are then written to the disks.

Without logging, it is possible for data not involved in any active writes to be lost or silently corrupted if a disk fails and the system also fails. If this double-failure occurs, there is no way of knowing if the data being written to the data portions of the disks or the parity being written to the parity portions have actually been written. Therefore, the recovery of the corrupted disk may be corrupted itself.

In Figure 1-18, Incomplete Write,, the recovery of Disk B is dependent on the data on Disk A and the parity on Disk C having both been completed. The diagram shows a completed data write and an incomplete parity write causing an incorrect data reconstruction for the data on Disk B.

Figure 1-18 **Incomplete Write**



This failure case can be avoided by logging all data writes before committing them to the array. In this way, the log can be replayed, causing the data and parity updates to be completed before the reconstruction of the failed drive takes place.

Logs are associated with a RAID-5 volume by being attached as additional, non-RAID-5 layout plexes. More than one log plex can exist for each RAID-5 volume, in which case the log areas are mirrored.

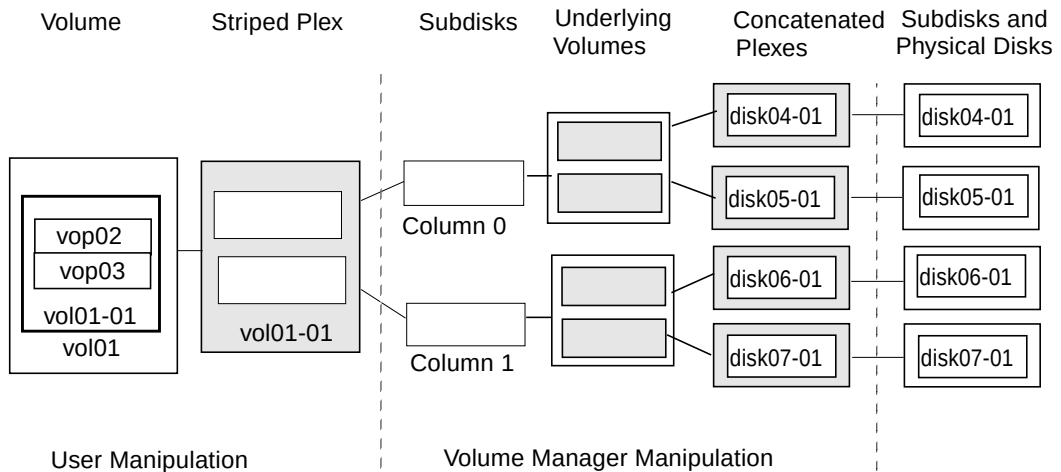
Layered Volumes

Another Volume Manager virtual object is the **layered volume**. A layered volume is built on top of volume(s). The layered volume structure tolerates failure better and has greater redundancy than the standard volume structure. For example, in a striped and mirrored layered volume, each mirror (plex) covers a smaller area of storage space, so recovery is quicker than with a standard mirror volume. Figure 1-19, Example of a Striped-Mirrored Layered Volume, shows an example layered volume design.

In Figure 1-19, Example of a Striped-Mirrored Layered Volume,, illustrates how the volume and striped plex in the “User Manipulation” area allow you to perform normal Volume Manager tasks. User tasks can be performed only on the top-level volume of a layered volume. You cannot detach a layered volume or perform any other operation on the underlying volumes by manipulating the internal structure. You can perform all necessary operations from the user manipulation area that includes the volume and striped plex (for example, to change the column width, or to add a column).

The “Volume Manager Manipulation” area shows subdisks with two columns, built on underlying volumes with each volume internally mirrored. Layered volumes are an infrastructure within Volume Manager and they allow the addition of certain features to be added to Volume Manager. Underlying volumes are used exclusively by the Volume Manager and are not designed for user manipulation. The underlying volume structure is described here to help you understand how layered volumes work and why they are used by Volume Manager.

Figure 1-19 Example of a Striped-Mirrored Layered Volume



System administrators may need to manipulate the layered volume structure for troubleshooting or other operations (for example, to place data on specific disks). Layered volumes are used by Volume Manager to perform these tasks and operations:

- striped-mirrors (see the vxassist(1M) manual page)
- concatenated mirrors (see the vxassist(1M) manual page)
- Online Relayout (see the vxrelayout(1M) and vxassist(1M) manual pages)
- RAID-5 subdisk moves (see the vxsd(1M) manual page)
- RAID-5 snapshot (see the vxassist(1M) manual page)

Volume Manager User Interfaces

This section briefly describes the VERITAS Volume Manager user interfaces.

User Interface Overview

The Volume Manager supports the following user interfaces:

- Volume Manager Storage Administrator

The Storage Administrator is a graphical user interface to the Volume Manager. The Storage Administrator provides visual elements such as icons, menus, and dialog boxes to manipulate Volume Manager objects. The Storage Administrator also acts as an interface to some common file system operations. The Storage Administrator is described in the *Storage Administrator Administrator's Guide*.

- Command Line Interface

The Volume Manager command set (for example, the `vxassist` command) ranges from commands requiring minimal user input to commands requiring detailed user input. For information on the Command Line Interface, see Chapter 3, Volume Manager Operations,.

- Volume Manager commands range from simple commands to complex commands requiring detailed user input. Many Volume Manager commands require you to have an understanding of Volume Manager concepts. Volume Manager concepts are described in this chapter. Most Volume Manager commands require superuser or other appropriate privileges. For a listing and explanation of Volume Manager commands, see the “Volume Manager Commands” in the *Volume Manager Reference Guide*.

- Volume Manager Support Operations

Volume Manager Support Operations interface (`vxdiskadm`) is a menu-driven interface for disk and volume administration functions. The `vxdiskadm` interface uses a command line main menu where you can select storage management tasks to be performed. For more information on the `vxdiskadm` interface, see “`vxdiskadm` Main Menu” and “`vxdiskadm` Menu Description”.

Volume Manager objects created by one interface are compatible with those created by the other interfaces.

Volume Manager Conceptual Overview

This section describes key terms and relationships between Volume Manager concepts. Figure 1-20, Volume Manager System Concepts,, illustrates the terms and concepts discussed in this section.

Why You Should Use Volume Manager

Volume Manager provides enhanced data storage service by separating the physical and logical aspects of data management. Volume Manager enhances data storage by controlling these aspects of storage:

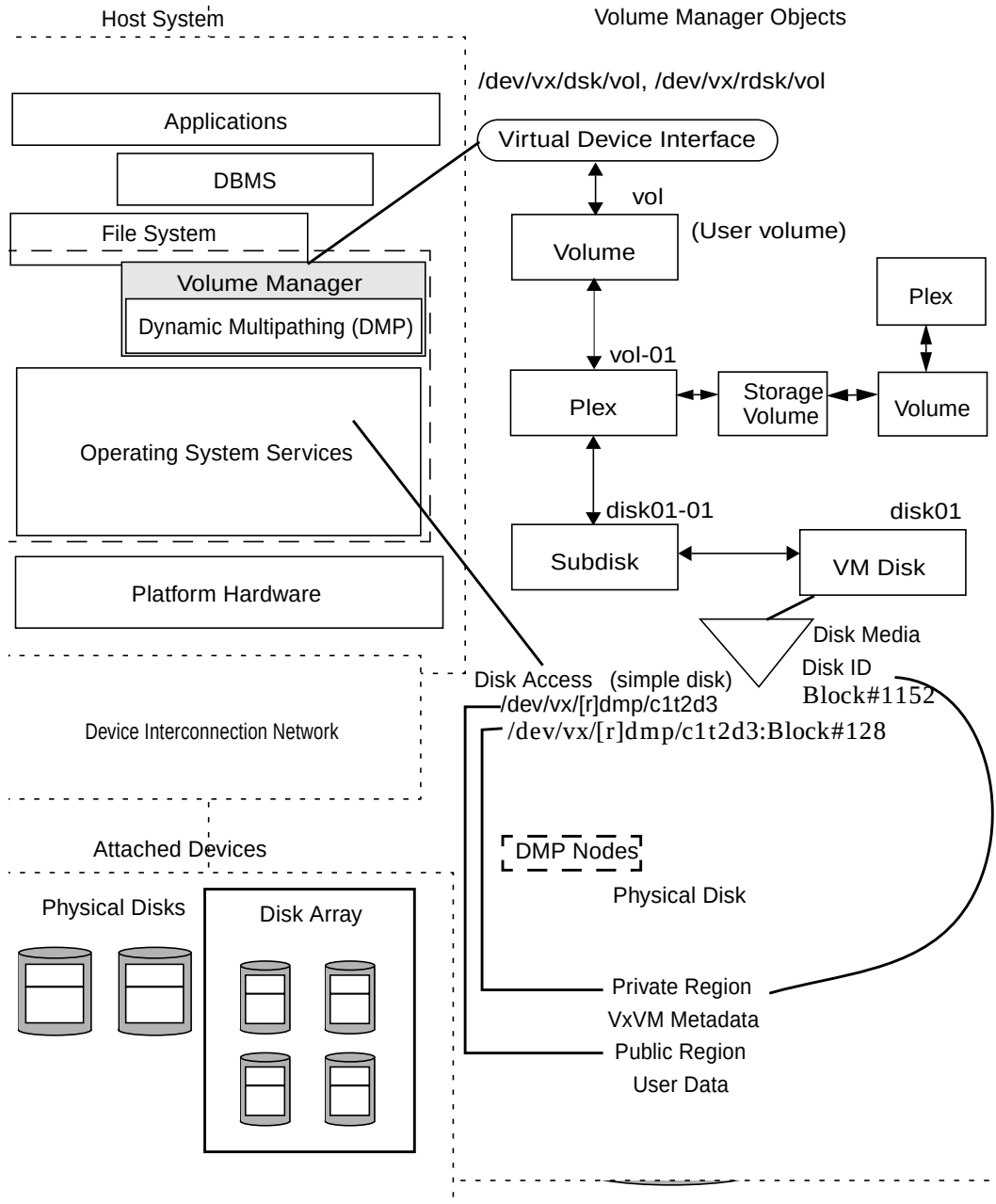
- space—allocation and use
- performance—by enhancing data delivery
- data availability—continuous operation and multisystem access
- device installation—centralized and optimized support
- system—multisystem support and monitoring of private/shared systems

Volume Manager Objects

Volume Manager is a storage management subsystem that allows you to manage physical disks as logical devices called **volumes**. The Volume Manager interfaces provide enhanced data access and storage management by using volumes. A volume is a logical device that appears to data management systems as a physical disk partition device. Volumes provide enhanced recovery, data availability, performance, and storage configuration options.

A Volume Manager volume is a logical **object**. Volume Manager creates other objects that you can operate, control, monitor, and query to optimize storage management. To configure and maintain a volume for use, Volume Manager places physical disks under its control and collects the disk space into **disk groups**. A disk group is a collection of claimed disks organized into logical volumes. Volume Manager then allocates the space on those disks to logical volumes. See Figure 1-20, Volume Manager System Concepts,.

Figure 1-20 Volume Manager System Concepts



After installing Volume Manager on a host system, perform the following procedure before you can configure and use Volume Manager objects:

- bring the contents of physical disks under Volume Manager control
bring the physical disk under VxVM control, the disk must not be under LVM control. For more information on LVM and VxVM disk co-existence or how to convert LVM disks to VxVM disks, see the *VERITAS Volume Manager Migration Guide*.

- collect the Volume Manager disks into disk groups
- allocate the disk group space to create logical volumes

Bringing the contents of physical disks under Volume Manager control is done only if:

- you allow Volume Manager to take control of the physical disks
- the disk is not under control of another storage manager

Volume Manager writes identification information on physical disks under Volume Manager control (claimed disks). Claimed disks can be identified even after physical disk disconnection or system outages. Volume Manager can then re-form disk groups and logical objects to provide failure detection and to speed system recovery.

Volume Manager and the Operating System

Volume Manager operates as a subsystem between your operating system and your data management systems, such as file systems and database management systems.

Before a disk can be brought under Volume Manager control, the disk must be accessible through the operating system device interface. Volume Manager is a subsystem layered on top of the operating system interface services. Therefore, Volume Manager is dependent upon how the operating system accesses physical disks.

Volume Manager is dependent upon the operating system for the following.

- operating system (disk) devices
- device handles
- VM disks
- Volume Manager dynamic multipathing (DMP) metadvice

Dynamic Multipathing (DMP)

NOTE

You may need an additional license to use this feature.

A multipathing condition can exist when a physical disk can be accessed by more than one operating system device handle. Each multipath operating system device handle permits data access and control through alternate host-to-device pathways.

Volume Manager is configured with its own DMP system to organize access to multipathed devices. Volume Manager detects multipath systems by using the Universal World-Wide-Device Identifiers (WWD IDs). The physical disk must provide unambiguous identification through its WWD ID for DMP to access the device.

If DMP cannot identify the physical disk through its WWD ID, identification is left to the Volume Manager device detection methods. Device detection depends on Volume Manager recognizing on-disk metadata identifiers.

Volume Manager DMP creates metanodes representing metadevices for each multipath target that it has detected. Each metanode is mapped to a set of operating system device handles and configured with an appropriate multipathing policy. Volume Manager DMP creates metanodes for all attached physical disks accessible through an operating system device handle.

Volume Manager DMP manages multipath targets, such as disk arrays, which define policies for using more than one path. Some disk arrays permit more than one path to be concurrently active (Active / Active). Some disk arrays permit only one path to be active, holding an alternate path as a spare in case of failure on the existing path (Active / Passive). Some disk arrays have more elaborate policies.

In general, Volume Manager is designed so that the VM disk is mapped to one Volume Manager DMP metanode. To simplify VxVM logical operations, each VM disk is mapped to a unique Volume Manager DMP metanode. The mapping occurs whether or not the physical disk device is connected in a multipathing configuration.

When using Volume Manager DMP, you should be aware of the layering of device recognition:

- How does the operating system view the paths?
- How does Volume Manager DMP view the paths?
- How does the Multipathing target deal with their paths?

References Additional information about DMP can be found in this document in the following sections:

- “Dynamic Multipathing (DMP)”
- “vxdctl Daemon”
- Chapter 4, Disk Tasks,
“Displaying Multipaths Under a VM Disk”

Related information also appears in the following sections of the *Volume Manager Reference Guide*:

- “*Volume Manager Commands*”
- “*DMP Error Messages*”
- “*Multipathed Disk Arrays*”

Volume Manager Layouts

A Volume Manager virtual device is defined by a volume. A volume has a layout defined by the association of a volume to one or more plexes, which in turn, each map to subdisks. The volume then presents a virtual device interface exposed to Volume Manager clients for data access. These logical building blocks re-map the volume address space through which I/O is re-directed at run-time.

Different volume layouts each provide different levels of storage service. A volume layout can be configured and re-configured to match particular levels of desired storage service.

In previous releases of Volume Manager, the subdisk was restricted to mapping directly to a VM disk. This allowed the subdisk to define a contiguous extent of storage space backed by the public region of a VM disk. When active, the VM disk is associated with an underlying physical disk, this is how Volume Manager logical objects map to physical objects, and stores data on stable storage.

The combination of a volume layout and the physical disks which provide backing store, therefore determine the storage service available from a given virtual device.

In the 3.0 or higher release of Volume Manager “layered volumes” can be constructed by permitting the subdisk to map either to a VM disk as before, or, to a new logical object called a **storage volume**. A storage volume provides a recursive level of mapping with layouts similar to the top-level volume. Eventually, the “bottom” of the mapping requires an association to a VM disk, and hence to attached physical storage.

Layered volumes allow for more combinations of logical compositions, some of which may be desirable for configuring a virtual device. Because permitting free use of layered volumes throughout the command level would have resulted in unwieldy administration, some ready-made layered volume configurations have been designed into the 3.0 release of the Volume Manager.

These ready-made configurations operate with built-in rules to automatically match desired levels of service within specified constraints. The automatic configuration is done on a “best-effort” basis for the current command invocation working against the current configuration.

To achieve the desired storage service from a set of virtual devices, it may be necessary to include an appropriate set of VM disks into a disk group, and to execute multiple configuration commands.

To the extent that it can, Volume Manager handles initial configuration and on-line re-configuration with its set of layouts and administration interface to make this job easier.

Volume Administration Tasks

Volume Manager can be used to perform system and configuration management tasks on its objects: disks, disk groups, subdisks, plexes, and volumes. Volumes contain two types of objects:

- subdisk—a region of a physical disk
- plex—a series of subdisks linked together in an address space

Disks and disk groups must be initialized and defined to the Volume Manager before volumes can be created. Volumes can be created in either of the following ways:

Automatically—you can create and manipulate volumes by using the `vxassist` command. The `vxassist` command creates required plexes and subdisks by using only the basic attributes of the desired volume as input. The `vxassist` command can also modify existing volumes. It automatically modifies any underlying or associated objects. The `vxassist` command uses default values for many volume attributes, unless you provide specific values.

Manually—you can create volumes by first creating the volume subdisks, then the plexes, then the volume itself (a bottom-up approach). This approach often requires detailed user input and an understanding of Volume Manager concepts. Creating volumes manually is useful for manipulating volumes with specific, non-default attributes. The manual approach does not use the `vxassist` command.

NOTE

This section focuses on how to create and modify volumes using the manual “bottom-up” approach. However, it can be more convenient to use the automated `vxassist` command approach to volume management.

Many of the concepts and procedures in this section are normally “hidden” from the casual user. Some procedures are managed automatically by higher level commands such as the `vxassist` command. Some of the commands and procedures described are not normally performed by system administrators.

The steps normally involved in creating volumes manually are:

- create subdisks
- create plexes
- associate subdisks and plexes
- create volumes
- associate volumes and plexes
- initialize volumes

Before you create volumes, you should determine which volume layout best suits your system needs.

2 **Initialization and Setup**

Introduction

This chapter briefly describes the steps needed to initialize Volume Manager and the daemons that must be running for Volume Manager to operate. This chapter also provides guidelines to help you set up a system with storage management.

See the *VERITAS Volume Manager 3.1 for HP-UX Release Notes* for detailed information on how to install and set up the Volume Manager and the Storage Administrator on HP-UX 11i systems.

NOTE

On HP-UX 11i Version 1.5 systems, initialization and setup is done automatically when the default operating system is installed.

The following topics are covered in this chapter:

- “Volume Manager Initialization”
- “Volume Manager Daemons”
- “System Setup”
- “System Setup Guidelines”
- “Protecting Your System”

Volume Manager Initialization

You initialize the Volume Manager by using the `vxinstall` program. The `vxinstall` program places specified disks under Volume Manager control. By default, these disks are placed in the `rootdg` disk group. You must use the `vxinstall` program to initialize at least one disk into `rootdg`. You can then use the `vxdiskadm` interface or the Storage Administrator to initialize additional disks into other disk groups.

Once you have completed the package installation, follow these steps to initialize the Volume Manager.

NOTE

Do not run `vxinstall` on HP-UX 11i Version 1.5 if you installed the default operating system, because VxVM initialization is done automatically when the default operating system is installed. If, however, you chose the VxFS whole disk option when you installed the operating system, then you will need to run `vxinstall`.

- Step 1.** Log in as superuser (or appropriate access level).
- Step 2.** Create a `disks.exclude` file if you want to exclude disks from Volume Manager control. The `vxinstall` program ignores any disks listed in this file. Place this file in `/etc/vx/disks.exclude`.

NOTE

The files `/etc/vx/cntrls.exclude` and `/etc/vx/disks.exclude` are used by the `vxinstall` and `vxdiskadm` utilities to automatically exclude controllers or disks so that these devices are not configured as Volume Manager devices. These files do not exclude controllers and disks from use by any other Volume Manager commands. See the `vxinstall(1M)` and `vxdiskadm(1M)` manual pages for more information.

- Step 3.** Create a `cntrls.exclude` file if you want to exclude all disks on a controller from Volume Manager control. Place this file in `/etc/vx/cntrls.exclude`.
- Step 4.** Start the `vxinstall` program by entering the following command:

vxinstall

The vxinstall program then does the following:

- examines and lists all controllers attached to the system
- describes the installation process: Quick or Custom installation

Quick installation gives you the option of initializing all disks. If you wish to initialize only some of the disks for VxVM, use the custom installation process.

Custom installation allows you to control which disks are added to Volume Manager control and how they are added. You can initialize all disks on a controller or only the disks you plan on using.

For details on how to use the Quick or Custom installation option, see the *VERITAS Volume Manager 3.1 for HP-UX Release Notes*.

Volume Manager Daemons

Two daemons must be running for the Volume Manager to operate properly:

- vxconfigd
- vxiod

Configuration Daemon vxconfigd

The Volume Manager configuration daemon (vxconfigd) maintains Volume Manager disk and disk group configurations. The vxconfigd daemon communicates configuration changes to the kernel and modifies configuration information stored on disk.

Starting the Volume Manager Configuration Daemon

Startup scripts invoke the vxconfigd daemon during the boot procedure.

To determine whether the volume daemon is enabled, use the following command:

```
# vxdctl mode
```

This message is displayed if the vxconfigd daemon is running and enabled:

```
mode: enabled
```

This message is displayed if vxconfigd is running, but not enabled:

```
mode: disabled
```

To enable the volume daemon, use the following command:

```
# vxdctl enable
```

This message is displayed if the vxconfigd daemon is not running:

```
mode: not-running
```

To start the vxconfigd daemon, use the following command:

```
# vxconfigd
```

Once started, the vxconfigd daemon automatically becomes a background process.

By default, the `vxconfigd` daemon issues errors to the console. However, the `vxconfigd` daemon can be configured to issue errors to a log file.

For more information, see the `vxconfigd(1M)` and `vxctl(1M)` manual pages.

Volume I/O Daemon `vxiod`

The volume extended I/O daemon (`vxiod`) allows for extended I/O operations without blocking calling processes.

For more information, see the `vxiod(1M)` manual page.

Starting the Volume I/O Daemon

The `vxiod` daemons are started at system boot time. There are typically several `vxiod` daemons running at all times. Rebooting after your initial installation starts the `vxiod` daemon.

Verify that `vxiod` daemons are running by entering this command:

```
# vxiod
```

Because the `vxiod` daemon is a kernel thread and is not visible to you through the `ps` command, this is the only way to see if any `vxiod` daemons are running.

If any `vxiod` daemons are running, the following message is displayed:

```
10 volume I/O daemons running
```

where 10 is the number of `vxiod` daemons currently running.

If no `vxiod` daemons are currently running, start some by entering this command:

```
# vxiod set 10
```

where 10 can be substituted by the desired number of `vxiod` daemons. It is recommended that at least one `vxiod` daemon exist for each CPU in the system.

System Setup

This section has information to help you set up your system for efficient storage management. For additional information on system setup tasks, see the *Volume Manager Storage Administrator Administrator's Guide*.

The following system setup sequence is typical and should be used as an example. Your system requirements may differ. The system setup guidelines provide helpful information for specific setup configurations.

Example System Setup Sequence

The following list describes typical activities you can use when setting up your storage management system.

Initial Setup

- Step 1.** Place disks under Volume Manager control.
- Step 2.** Create new disk groups (if you do not want to use rootdg or you want other disk groups).
- Step 3.** Create volumes.
- Step 4.** Put file system(s) in volumes.

Options

- Designate hot-relocation spare disks.
- Add mirrors to volumes if necessary.

Maintenance

- Resize volumes and file systems.
- Add more disks/disk groups.
- Create snapshots.

System Setup Guidelines

These general guidelines can help you to understand and plan an efficient storage management system. See the cross-references in each section for more information about the featured guideline.

Hot-Relocation Guidelines

NOTE

You may need an additional license to use this feature.

Follow these general guidelines when using hot-relocation. See “Hot-Relocation” for more information.

- The hot-relocation feature is enabled by default only if you have the license for this feature. Although it is possible to disable hot-relocation, it is advisable to leave it enabled.
- Although hot-relocation does not require you to designate disks as spares, you can designate at least one disk as a spare within each disk group. This gives you some control over which disks are used for relocation. If no spares exist, Volume Manager uses any available free space within the disk group. When free space is used for relocation purposes, it is possible to have performance degradation after the relocation.
- After hot-relocation occurs, you can designate one or more additional disks as spares to augment the spare space (some of the original spare space may be occupied by relocated subdisks).
- If a given disk group spans multiple controllers and has more than one spare disk, you can set up the spare disks on different controllers (in case one of the controllers fails).
- For a mirrored volume, the disk group must have at least one disk that does not already contain a mirror of the volume. This disk should either be a spare disk with some available space or a regular disk with some free space and the disk is not excluded from hot-relocation use.
- For a mirrored and striped volume, the disk group must have at least

one disk that does not already contain one of the mirrors of the volume or another subdisk in the striped plex. This disk should either be a spare disk with some available space or a regular disk with some free space and the disk is not excluded from hot-relocation use.

- For a RAID-5 volume, the disk group must have at least one disk that does not already contain the RAID-5 plex (or one of its log plexes) of the volume. This disk should either be a spare disk with some available space or a regular disk with some free space and the disk is not excluded from hot-relocation use.
- If a mirrored volume has a DRL log subdisk as part of its data plex, that plex cannot be relocated. You can place log subdisks in plexes that contain no data (log plexes).
- Hot-relocation does not guarantee that it preserves the original performance characteristics or data layout. You can examine the location(s) of the newly-relocated subdisk(s) and determine whether they should be relocated to more suitable disks to regain the original performance benefits.
- Hot-relocation is capable of creating a new mirror of the root disk if the root disk is mirrored and it fails. The `rootdg` disk group should therefore contain sufficient contiguous spare or free space to accommodate the volumes on the root disk (`rootvol` and `swapvol` require contiguous disk space).
- Although it is possible to build VxVM objects on spare disks (using `vxmake` or the Storage Administrator interface), it is preferable to use spare disks for hot-relocation only.

Striping Guidelines

Follow these general guidelines when using striping. See “Striping (RAID-0)” for more information.

- Do not place more than one column of a striped plex on the same physical disk.
- Calculate stripe unit sizes carefully. In general, a moderate stripe unit size (for example, 64 kilobytes, which is also the default used by the `vxassist` command) is recommended. If it is not feasible to set the stripe unit size to the track size, and you do not know the application I/O pattern, it is recommended that you use 64 kilobytes for the stripe unit size.

NOTE

Many modern disk drives have “variable geometry,” which means that the track size differs between cylinders (i.e., outer disk tracks have more sectors than inner tracks). It is therefore not always appropriate to use the track size as the stripe unit size. For these drives, use a moderate stripe unit size (such as 64 kilobytes), unless you know the I/O pattern of the application.

- Volumes with small stripe unit sizes can exhibit poor sequential I/O latency if the disks do not have synchronized spindles. Generally, striping over non-spindle-synched disks performs better if used with larger stripe unit sizes and multi-threaded, or largely asynchronous, random I/O streams.
- Typically, the greater the number of physical disks in the stripe, the greater the improvement in I/O performance; however, this reduces the effective mean time between failures of the volume. If this is an issue, striping can be combined with mirroring to provide a high-performance volume with improved reliability.
- If only one plex of a mirrored volume is striped, be sure to set the policy of the volume to `prefer` for the striped plex. (The default read policy, `select`, does this automatically.)
- If more than one plex of a mirrored volume is striped, make sure the stripe unit size is the same for each striped plex.
- Where possible, distribute the subdisks of a striped volume across drives connected to different controllers and buses.
- Avoid the use of controllers that do not support overlapped seeks (these are rare).

The `vxassist` command automatically applies and enforces many of these rules when it allocates space for striped plexes in a volume.

Mirroring Guidelines

NOTE

You must license the VERITAS Volume Manager product to use this feature.

Follow these general guidelines when using mirroring. See “Mirroring (RAID-1)” for more information.

- Do not place subdisks from different plexes of a mirrored volume on the same physical disk. This action compromises the availability benefits of mirroring and degrades performance. Use of the `vxassist` command precludes this from happening.
- To provide optimum performance improvements through the use of mirroring, at least 70 percent of the physical I/O operations should be read operations. A higher percentage of read operations results in a higher benefit of performance. Mirroring may not provide a performance increase or result in performance decrease in a write-intensive workload environment.

NOTE

The UNIX operating system implements a file system cache. Read requests can frequently be satisfied from the cache. This can cause the read/write ratio for physical I/O operations through the file system to be biased toward writing (when compared to the read/write ratio at the application level).

- Where feasible, use disks attached to different controllers when mirroring or striping. Most disk controllers support overlapped seeks that allow seeks to begin on two disks at once. Do not configure two plexes of the same volume on disks attached to a controller that does not support overlapped seeks. This is important for older controllers or SCSI disks that do not cache on the drive. It is less important for many newer SCSI disks and controllers used in most modern workstations and server machines. Mirroring across controllers can be of benefit because the system can survive a controller failure. The other controller can continue to provide data from the other mirror.
- A plex can exhibit superior performance due to being striped or concatenated across multiple disks, or because it is located on a much faster device. The read policy can then be set to prefer the “faster” plex. By default, a volume with one striped plex is configured with preferred reading of the striped plex.

Dirty Region Logging (DRL) Guidelines

NOTE

You must license the VERITAS Volume Manager product to use this feature.

Dirty Region Logging (DRL) can speed up recovery of mirrored volumes following a system crash. When DRL is enabled, Volume Manager keeps track of the regions within a volume that have changed as a result of writes to a plex. Volume Manager maintains a bitmap and stores this information in a **log subdisk**. Log subdisks are defined for and added to a volume to provide DRL. Log subdisks are independent of plexes, are ignored by plex policies, and are only used to hold the DRL information.

NOTE

Using Dirty Region Logging can impact system performance in a write-intensive environment.

Follow these guidelines when using DRL:

- For Dirty Region Logging to be in effect, the volume must be mirrored.
- At least one log subdisk must exist on the volume for DRL to work. However, only one log subdisk can exist per plex.
- The subdisk that is used as the log subdisk should not contain necessary data.
- It is possible to “mirror” log subdisks by having more than one log subdisk (but only one per plex) in the volume. This ensures that logging can continue, even if a disk failure causes one log subdisk to become inaccessible.
- Log subdisks must be configured with two or more sectors (preferably an even number, as the last sector in a log subdisk with an odd number of sectors is not used). The log subdisk size is normally proportional to the volume size. If a volume is less than 2 gigabytes, a log subdisk of 2 sectors is sufficient. The log subdisk size should then increase by 2 sectors for every additional 2 gigabytes of volume size. However, the `vxassist` command chooses reasonable sizes by default. In general, use of the default log subdisk length provided by

the `vxassist` command is recommended.

- The log subdisk should not be placed on a heavily-used disk, if possible.
- Persistent (non-volatile) storage disks must be used for log subdisks.

Mirroring and Striping Guidelines

NOTE

You may need an additional license to use this feature.

Follow these general guidelines when using mirroring and striping together. For more information, see “Mirroring Plus Striping (RAID-1 + RAID-0)”.

- Make sure that there are enough disks available for the striped and mirrored configuration. At least two disks are required for the striped plex and one or more *additional* disks are needed for the mirror.
- Never place subdisks from one plex on the same physical disk as subdisks from the other plex. Follow the striping guidelines described in “Striping Guidelines”.
- Follow the mirroring guidelines described in “Mirroring Guidelines”.

Striping and Mirroring Guidelines

NOTE

You may need an additional license to use this feature.

Follow these general guidelines when using striping and mirroring together. For more information, see “Mirroring Plus Striping (RAID-1 + RAID-0)”.

- Make sure that there are enough disks available for the striped and mirrored configuration. At least two disks are required for the striped plex and one or more *additional* disks are needed for the mirror.
- Never place subdisks from one plex on the same physical disk as subdisks from the other plex. Follow the striping guidelines described in “Striping Guidelines”.

- Follow the mirroring guidelines described in “Mirroring Guidelines”.

RAID-5 Guidelines

NOTE

You may need an additional license to use this feature.

Follow these general guidelines when using RAID-5. See “RAID-5” for more information.

In general, the guidelines for mirroring and striping together also apply to RAID-5. The following guidelines should also be observed with RAID-5:

- Only one RAID-5 plex can exist per RAID-5 volume (but there can be multiple log plexes).
- The RAID-5 plex must be derived from at least two subdisks on two or more physical disks. If any log plexes exist, they must belong to disks other than those used for the RAID-5 plex.
- RAID-5 logs can be mirrored and striped.
- If the volume length is not explicitly specified, it is set to the length of any RAID-5 plex associated with the volume; otherwise, it is set to zero. If the volume length is set explicitly, it must be a multiple of the stripe unit size of the associated RAID-5 plex, if any.
- If the log length is not explicitly specified, it is set to the length of the smallest RAID-5 log plex that is associated, if any. If no RAID-5 log plexes are associated, it is set to zero.
- Sparse RAID-5 log plexes are not valid.

Protecting Your System

Disk failures can cause two types of problems: loss of data on the failed disk and loss of access to your system. Loss of access can be due to the failure of a key disk (a disk used for system operations). The VERITAS Volume Manager can protect your system from these problems.

To maintain system availability, data important to running and booting your system must be mirrored. The data must be preserved so it can be used in case of failure.

Here are suggestions on how to protect your system and data:

- Use mirroring to protect data. By mirroring your data, you prevent data loss from a disk failure. To preserve data, create and use mirrored volumes that have at least two data plexes. The plexes must be on different disks. If a disk failure causes a plex to fail, the data in the mirrored volume still exists on the other disk.

If you use the `vxassist mirror` command to create mirrors, it locates the mirrors so the loss of one disk does not result in a loss of data. By default, the `vxassist` command does not create mirrored volumes; edit the file `/etc/default/vxassist` to set the default layout to mirrored. For information on the `vxassist` defaults file, see “`vxassist` Command”.

- Leave the Volume Manager hot-relocation feature enabled to automatically detect failures, notify you of the nature of the failures, attempt to relocate any affected subdisks that are redundant, and initiate recovery procedures. Provide at least one hot-relocation spare disk per disk group so sufficient space is available for relocation in the event of a failure.
- For mirrored volumes, take advantage of the Dirty Region Logging feature to speed up recovery of your mirrored volumes after a system crash. Make sure that each mirrored volume has at least one log subdisk. (Note that the `usr` volume cannot have Dirty Region Logging turned on.)
- For RAID-5 volumes, take advantage of logging to prevent corruption of recovery data. Make sure that each RAID-5 volume has at least one log plex.
- Perform regular backups to protect your data. Backups are necessary

if all copies of a volume are lost or corrupted in some way. For example, a power surge could damage several (or all) disks on your system. Also, typing a command in error can remove critical files or damage a file system directly.

3 **Volume Manager Operations**

Introduction

This chapter provides information about the VERITAS Volume Manager command line interface (CLI). The Volume Manager command set (for example, the `vxassist` command) ranges from commands requiring minimal user input to commands requiring detailed user input. Many of the Volume Manager commands require an understanding of Volume Manager concepts. For more information on Volume Manager concepts, see “Volume Manager Conceptual Overview”. Most Volume Manager commands require access privileges.

NOTE

Rootability or bringing the root disk under the VxVM control, is not supported for the HP-UX 11i release of Volume Manager. Rootability is supported on HP-UX 11i Version 1.5 systems.

See the chapters that follow for information on using Volume Manager commands to perform common tasks. The `vxintro(1M)` manual page also contains introductory information relating to Volume Manager tasks.

The following topics are covered in this chapter:

- “Displaying Disk Configuration Information”
- “Displaying Subdisk Information”
- “Creating Volumes”
- “Volume Manager Task Monitor”
- “Performing Online Backup”
- “Exiting the Volume Manager Support Tasks”
- “Online Relayout”
- “Hot-Relocation”
- “Volume Resynchronization”
- “Dirty Region Logging”
- “FastResync (Fast Mirror Resynchronization)”
- “Volume Manager Rootability”

- “Dynamic Multipathing (DMP)”
- “VxSmartSync Recovery Accelerator”
- “Common Volume Manager Commands”

NOTE

Your system can use a **device name** that differs from the examples. For more information on device names, see “Disk Devices”.

Displaying Disk Configuration Information

You can display disk configuration information from the command line. Output listings available include: available disks, Volume Manager objects, and free space in disk groups.

Displaying Disk Listings

To display a list of available disks, use the following command:

```
# vxdisk list
```

The output of the `vxdisk list` command lists the device name, the type of disk, the disk name, the disk group to which the disk belongs, and status of the disk. The following is an example of `vxdisk list` command output:

DEVICE	TYPE	DISK	GROUP	STATUS
c0t0d0	simple	disk01	rootdg	online
c0t1d0	simple	disk02	rootdg	LVM
c0t2d0	simple	disk03	rootdg	online invalid

Displaying Volume Manager Object Listings

The `vxprint` command displays detailed information on existing Volume Manager objects. To display detailed output for all currently existing objects, use the following command:

```
# vxprint -ht
```

The following are examples of `vxprint` command output:

```
Disk group: rootdg VDGNAME NCONFIG LOGMINOR SGROUP - ID
DMNAME DEVICETYPE PRIVLEN PUBLEN STATE
RVNAME RLINK_CNT KSTATE STATE PRIMARY DATA VOL SSRL
RLNAME RVGK STATE STATE REM_HOST REM_DG REM_RLNK
VNAME RVGK STATE STATE LENGTH HSETYPE PREF PLEX RDPOL
PLNAME VOLUME KSTATE STATE LENGTH LAYOUT NCOL/WIDMODE
SDNAME PLEX DISK DISK OFFS LENGTH [COL/] OFF DEVICEMODE
SVNAME PLEX VOL NAME VOLLAY RLENGTH [COL/] OFF FAM/NM MODE
```

```
dgrootdgdefaultdefault0962910960.1025.bass
dmdisk01c0t10d0simple10244444228 -
dmdisk02c0t11d0simple10244443310-
```

```
Disk group: newdg DGNAMENCONFIGNLOGMINORSGROUP-ID
DMNAMEDEVICETYPEPRIVLENPUBLENSTATE
RVNAMERLINK_CNTKSTATESTATEPRIMARYDATAVOLSSRL
RLNAMERVGKSTATESTATEEREM_HOSTREM_DGREM_RLNK
VNAMERVGKSTATESTATELENGTHHSETYPEPREFPLEXRDPOL
PLNAMEVOLUMEKSTATESTATELENGTHLAYOUTNCOL/WIDMODE
SDNAMEPLEXDISKDISKOFFSLENGTH[COL/]OFFDEVICEMODE
SVNAMEPLEXVOLNAMENVOLLAYRLENGTH[COL/]OFFFAM/NMMODE
dgnewdgdefaultdefault4963000963504895.1075.bass
dmnewdg01c0t12d0simple10244443310-
dmnewdg02c0t13d0simple10244443310-
```

In this output, **dg** is a disk group, **dm** is a disk, **v** is a volume, **pl** is a plex (mirror), and **sd** is a subdisk. The top few lines indicate the headers that match each type of output line that follows. Each volume is listed along with its associated plex(es) and subdisk(s).

Displaying Free Space in a Disk Group

Before you add volumes and file systems to your system, make sure you have enough free disk space to meet your needs. Use the Volume Manager to request a display of free space.

To display free space in the system, use the following command:

```
# vxvg free
```

The following is an example of **vxvg free** command output:

GROUP	DISK	DEVICE	TAG	OFFSET	LENGTH	FLAGS
rootdg	disk01	c0t10d0	c0t10d0	0	4444228	-
rootdg	disk02	c0t11d0	c0t11d0	0	4443310	-
newdg	newdg01	c0t12d0	c0t12d0	0	4443310	-
newdg	newdg02	c0t13d0	c0t13d0	0	4443310	-
oradg	oradg01	c0t14d0	c0t14d0	0	4443310	-

To display free space for a disk group, use the following command:

```
# vxvg -g diskgroup free
```

where **-g *diskgroup*** optionally specifies a disk group.

Volume Manager Operations

Displaying Disk Configuration Information

For example, to display the free space in the default disk group, rootdg, use the following command:

```
# vxdg -g rootdg free
```

The following is an example of `vxdg -g rootdg free` command output:

DISK FLAGS	DEVICE	TAG	OFFSET	LENGTH
disk01	c0t10d0	c0t10d0	0	4444228
-				
disk02	c0t11d0	c0t11d0	0	4443310
-				

The free space is measured in 1024-byte sectors.

Displaying Subdisk Information

The `vxprint` command displays information about Volume Manager objects. To display general information for all subdisks, use the following command:

```
# vxprint -st
```

The following is an example of `vxprint -st` command output:

Disk group: rootdg

SD NAME	PLEX	DISK	DISKOFFS	LENGTH	[COL/]OFF
DEVICE	MODE				
SV NAME	PLEX	VOLNAME	NVOLLAYR	LENGTH	[COL/]OFF
AM/NM	MODE				
sd disk01-01	vol1-01	disk01	0	102400	0
c0t10d0	ENA				
sd disk02-01	vol2-01	disk02	0	102400	0
c0t11d0	ENA				

The `-s` option specifies information about subdisks. The `-t` option prints a single-line output record that depends on the type of object being listed.

You can display complete information about a particular subdisk by using the following command:

```
# vxprint -l subdisk_name
```

For example, to obtain all information on a subdisk named `disk02-01`, use the following command:

```
# vxprint -l disk02-01
```

The following is an example of `vxprint -l disk02-01` command output:

Disk group: rootdg

```
Subdisk:  disk02-01
info:     disk=disk02 offset=0 len=102400
assoc:    vol=vol2 plex=vol2-01 (offset=0)
flags:    enabled
device:   device=c0t11d0 path=/dev/vx/dmp/c0t11d0
```

Volume Manager Operations
Displaying Subdisk Information

diskdev=102/8

Creating Volumes

Volumes are created to take advantage of the Volume Manager concept of virtual disks. Once a volume exists, a file system can be placed on the volume to organize the disk space with files and directories. Also, applications such as databases can be used to organize data on volumes.

Create volumes using either a basic or advanced approach, as described below:

- **Basic**—The basic approach takes information about what you want to accomplish and then performs the necessary underlying tasks. This approach requires only minimal input from you, but also permits more detailed specifications. Basic operations are performed primarily through the `vxassist` command.
- **Manual**—The advanced approach consists of a number of commands that typically require you to specify detailed input. These commands use a “building block” approach that requires you to have a detailed knowledge of the underlying structure and components to manually perform the commands necessary to accomplish a certain task. Advanced operations are performed through several Volume Manager commands.

The creation of a volume involves the creation of plex and subdisk components. With the basic approach to volume creation, you indicate the desired volume characteristics and the underlying plexes and subdisks are created automatically.

Volumes can be created with various layout types:

- **Concatenated**—A volume whose subdisks are arranged both sequentially and contiguously within a plex. Concatenation allows a volume to be created from multiple regions of one or more disks if there is not enough space for an entire volume on a single region of a disk.
- **Striped**—A volume with data spread evenly across multiple disks. **Stripes** are equal-sized fragments that are allocated alternately and evenly to the subdisks of a single plex. There must be at least two subdisks in a striped plex, each of which must exist on a different disk. Throughput increases with the number of disks across which a plex is striped. Striping helps to balance I/O load in cases where high

traffic areas exist on certain subdisks.

- **RAID-5**—A volume that uses striping to spread data and parity evenly across multiple disks in an array. Each stripe contains a parity stripe unit and data stripe units. Parity can be used to reconstruct data if one of the disks fails. In comparison to the performance of striped volumes, write throughput of RAID-5 volumes decreases since parity information needs to be updated each time data is accessed. However, in comparison to mirroring, the use of parity reduces the amount of space required.
- **Mirrored**—A volume with multiple plexes that duplicate the information contained in a volume. Although a volume can have a single plex, at least two are required for true mirroring (redundancy of data). Each of these plexes should contain disk space from different disks, for the redundancy to be useful.
- **Striped and Mirrored**—A volume with a striped plex and another plex that mirrors the striped one. This requires at least two disks for striping and one or more other disks for mirroring (depending on whether the plex is simple or striped). A striped and mirrored volume is advantageous because it both spreads data across multiple disks and provides redundancy of data.
- **Mirrored and Striped**—A volume with a plex that is mirrored and another plex that is striped. The mirrored and striped layout offers the benefits of spreading data across multiple disks (striping) while providing redundancy (mirroring) of data. The mirror and its striped plex are allocated from separate disks.
- **Layered Volume**—A volume built on top of volumes. Layered volumes can be constructed by mapping a subdisk to a VM disk or to a **storage** volume. A storage volume provides a recursive level of layout that is similar to the top-level volumes. Layered volumes allow for more combinations of logical layouts.

Volume Manager Task Monitor

The Volume Manager Task Monitor tracks the progress of system recovery by monitoring task creation, maintenance, and completion. The Task Monitor allows you to monitor task progress and to modify characteristics of tasks, such as pausing and recovery rate (for example, to reduce the impact on system performance). You can also monitor and modify the progress of the Online Relay layout feature.

Task Monitor Options

Command line option (-t) allows you to specify a **task tag** for any task.

You can use the Task Monitor option with the following utilities:

vxplex, vxsd, vxvol, vxrecover, vxreattach, vxresize, vxassist, and vxevac

Task monitor option -t takes the following form:

```
# utility [-t tasktag] ...
```

where:

utility—Volume Manager utilities that support the -t option.

tasktag—Assigns the given task tag to any tasks created by the utility. All tasks related to this operation are identified as a group.

For example, to execute a vxrecover command and track all resulting tasks as a group, specify a task tag to the vxrecover command, as follows:

```
# vxrecover -t myrecovery -b disk05
```

The vxrecover command creates a task to track all recovery jobs. To establish the task tag grouping, the vxrecover command specifies the tag to all utilities that it calls. Additionally, to establish a parent-child task relationship to any tasks those utilities execute, the vxrecover command passes its own task ID to those utilities.

For more information about the utilities, see their respective manual pages.

Performing Online Backup

Volume Manager provides snapshot backups of volume devices. This is done through the `vxassist` command and other commands. There are various procedures for doing backups, depending upon the requirements for integrity of the volume contents. These procedures have the same starting requirement: a plex that is large enough to store the complete contents of the volume. The plex can be larger than necessary, but if a plex that is too small is used, an incomplete copy results.

The recommended approach to volume backup is by using the `vxassist` command which is easy to use. The `vxassist snapstart`, `snapwait`, and `snapshot` tasks provide a way to do online backup of volumes with minimal disruption to users.

The `vxassist` snapshot procedure consists of two steps:

- Step 1.** Running `vxassist snapstart` to create a snapshot mirror
- Step 2.** Running `vxassist snapshot` to create a snapshot volume

You can use the `vxassist` command to create a snapshot of a RAID-5 volume by using the recommended approach to volume backup described in this section.

The `vxassist snapstart` step creates a write-only backup plex which gets attached to and synchronized with the volume. When synchronized with the volume, the backup plex is ready to be used as a snapshot mirror. The end of the update procedure is indicated by the new snapshot mirror changing its state to `SNAPDONE`. This change can be tracked by the `vxassist snapwait` task, which waits until at least one of the mirrors changes its state to `SNAPDONE`. If the attach process fails, the snapshot mirror is removed and its space is released.

Once the snapshot mirror is synchronized, it continues being updated until it is detached. You can then select a convenient time at which to create a snapshot volume as an image of the existing volume. You can also ask users to refrain from using the system during the brief time required to perform the snapshot (typically less than a minute). The amount of time involved in creating the snapshot mirror is long in contrast to the brief amount of time that it takes to create the snapshot volume.

The online backup procedure is completed by running the `vxassist snapshot` command on a volume with a `SNAPDONE` mirror. This task detaches the finished snapshot (which becomes a normal mirror), creates a new normal volume and attaches the snapshot mirror to the snapshot volume. The snapshot then becomes a normal, functioning mirror and the state of the snapshot is set to `ACTIVE`.

If the snapshot procedure is interrupted, the snapshot mirror is automatically removed when the volume is started.

Use the following steps to perform a complete backup using the `vxassist` command:

Step 1. Create a snapshot mirror for a volume with the following command:

```
# vxassist snapstart volume_name
```

Step 2. When the `snapstart` step is complete and the mirror is in a `SNAPDONE` state, choose a convenient time to complete the `snapshot` task. Inform users of the upcoming snapshot and ask them to save files and refrain from using the system briefly during that time.

Create a snapshot volume that reflects the original volume with this command:

```
# vxassist snapshot volume_name temp_volume_name
```

Step 3. Use `fsck` (or some utility appropriate for the application running on the volume) to clean the temporary volume's contents. For example, use this command:

```
# fsck -y /dev/vx/rdisk/temp_volume_name
```

Step 4. Copy the temporary volume to tape, or to some other appropriate backup media.

Step 5. Remove the new volume with this command:

```
# vxedit -rf rm temp_volume_name
```

Exiting the Volume Manager Support Tasks

When you have completed all of your disk administration activities, exit the Volume Manager Support Operations by selecting `q` from the `vxdiskadm` main menu. For a description of the `vxdiskadm` main menu and explanations of the menu options, see “`vxdiskadm` Main Menu” and “`vxdiskadm` Menu Description”.

Online Relayout

NOTE

You may need an additional license to use this feature.

Online Relayout allows you to convert any supported storage layout in the Volume Manager to any other, in place, with **uninterrupted data access**.

You usually change the storage layout in the Volume Manager to change the redundancy or performance characteristics of the storage. The Volume Manager adds redundancy to storage either by duplicating the address space (mirroring) or by adding parity (RAID-5). Performance characteristics of storage in the Volume Manager can be changed by changing the striping parameters which are the number of columns and the stripe width.

Layout changes can be classified into these types:

- RAID-5 to mirroring and mirroring to RAID-5
- adding or removing parity
- adding or removing columns
- changing stripe width

Storage Layout

Online Relayout currently supports these storage layouts:

- concatenation
- striped
- RAID-5
- mirroring (also supported where data is duplicated across different storage devices)
- striped-mirror
- concatenated-mirror

How Online Relayout Works

The VERITAS Online Relayout feature allows you to change storage layouts that you have already created in place, without disturbing data access. You can change the performance characteristics of a particular layout to suit changed requirements.

For example, if a striped layout with a 128K stripe unit size is not providing optimal performance, change the stripe unit size of the layout by using the Relayout feature. You can transform one layout to another by invoking a single command.

File systems mounted on the volumes do not need to be unmounted to achieve this transformation as long as the file system provides online shrink and grow operations.

Online Relayout reuses the existing storage space and has space allocation policies to address the needs of the new layout. The layout transformation process converts a given volume to the destination layout by using minimal temporary space.

The transformation is done by moving a portion-at-a-time of data in the source layout to the destination layout. Data is copied from the source volume to the temporary space, which removes data from the source volume storage area in portions. The source volume storage area is then transformed to the new layout and data saved in the temporary space is written back to the new layout. This operation is repeated until all the storage and data in the source volume has been transformed to the new layout.

You can use Online Relayout to change the number of columns, change stripe width, remove and add parity, and change RAID-5 to mirroring.

Types of Transformation

At least one of the following criteria must be met for an Online Relayout operation to be effective. You must perform one or more of the following operations:

- RAID-5 to or from mirroring
- change the number of columns
- change the stripe width
- remove or add parity

To be eligible for layout transformation, mirrored volume plexes must be identical in layout, with the same stripe width and number of columns. See Table 3-1, “Supported Layout Transformations.”

Table 3-1Supported Layout Transformations

From/To	Striped Mirrored	Concatenate d Mirrored	Regular Mirrored	RAID-5	Concatenate d	Striped
Striped Mirrored	Yes 1	Yes 2	No 3	Yes 4	Yes 5	Yes 6
Concatenated Mirrored	Yes 7	No 8	No 9	Yes 10	No 11	Yes 12
Regular Mirrored	Yes 13	Yes 14	No 15	Yes 16	No 17	No 18
RAID-5	Yes 4	Yes 10	No 19	Yes 20	Yes 21	Yes 22
Concatenated	Yes 5	No 11	No 17	Yes 21	No 23	Yes 24
Striped	Yes 6	Yes 12	No 18	Yes 22	Yes 24	Yes 25

Entries in Table 3-1, Supported Layout Transformations, include the following:

- *Yes*—indicates that an Online Relayout operation is possible.
- *No*—indicates that the operation *may* be possible but you cannot use Relayout.
- *Numbers*—indicate a brief description of the possible changes in that particular layout transformation. See the following “Number Descriptions”
- *Operations*—can be performed in both directions.

Number Descriptions

The numbers in Table 3-1, Supported Layout Transformations, describe the Relayout operation as follows:

1. Changes the stripe width or number of columns.
2. Removes all of the columns.

3. Not a Relayout, but a convert operation.
4. Changes mirroring to RAID-5 and/or stripe width/column changes.
5. Changes mirroring to RAID-5 and/or stripe width/column changes.
6. Changes stripe width/column and remove a mirror.
7. Adds columns.
8. Not a Relayout operation.
9. A convert operation.
10. Changes mirroring to RAID-5. See the `vxconvert` procedure.
11. Removes a mirror; not a Relayout operation.
12. Removes a mirror and add striping.
13. Changes an old mirrored volume to a stripe mirror. Relayout is valid only if there are changes to columns/stripe width; otherwise, this is a convert operation. See the `vxconvert` procedure.
14. Changes an old mirrored volume to a concatenated mirror. Relayout is valid only if there are changes to columns; otherwise, this is a convert operation.
15. No changes; not a Relayout operation.
16. Changes an old mirrored volume to RAID-5. Choose a plex in the old mirrored volume to use Relayout. The other plex is removed at the end of the Relayout operation.
17. Unless you choose a plex in the mirrored volume and change the column/stripe width, this is not a Relayout operation.
18. Unless you choose a plex in the mirrored volume and change the column/stripe width, this is not a Relayout operation.
19. Not a Relayout operation.
20. Changes stripe width/column.
21. Removes parity and all columns.
22. Removes parity.
23. No changes; not a Relayout operation.
24. Removes columns.
25. Changes stripe width/number of columns.

A striped mirror plex is a striped plex on top of a mirrored volume, resulting in a single plex that has both mirroring and striping. This combination forms a plex called a striped-mirror plex. A concatenated plex can be created in the same way. Online Relayout supports transformations to and from striped-mirror and concatenated-mirror plexes.

NOTE

Changing the number of mirrors during a transformation is not currently supported.

Transformation Characteristics

Transformation of data from one layout to another involves rearrangement of data in the existing layout to the new layout.

During the transformation, Online Relayout retains data redundancy by mirroring any temporary space used.

Read/write access to data is not interrupted during the transformation.

Data is not corrupted if the system fails during a transformation. The transformation continues after the system is restored and read/write access is maintained.

You can reverse the layout transformation process at any time, but the data may not be returned to the exact previous storage location. Any existing transformation in the volume should be stopped before doing a reversal.

You can determine the transformation direction by using the `vxrelayout status` command.

These transformations eliminate I/O failures as long as there is sufficient redundancy to move the data.

Transformations and Volume Length

Some layout transformations can cause the volume length to increase or decrease. If the layout transformation causes the volume length to increase or decrease, Online Relayout uses the `vxresize` command to shrink or grow a file system.

Sparse plexes are not transformed by Online Relayout, and no plex can

be rendered sparse by Online Relayout.

NOTE

Online Relayout can be used only with volumes created with the `vxassist` command.

Transformations Not Supported

Transformation of log plexes is not supported.

A snapshot of a volume when there is an Online Relayout operation running in the volume is not supported.

Hot-Relocation

NOTE

You may need an additional license to use this feature.

Hot-relocation allows a system to automatically react to I/O failures on redundant (mirrored or RAID-5) Volume Manager objects and restore redundancy and access to those objects. The Volume Manager detects I/O failures on objects and relocates the affected subdisks. The subdisks are relocated to disks designated as **spare disks** and/or free space within the disk group. The Volume Manager then reconstructs the objects that existed before the failure and makes them redundant and accessible again.

When a partial disk failure occurs (that is, a failure affecting only some subdisks on a disk), redundant data on the failed portion of the disk is relocated. Existing volumes on the unaffected portions of the disk remain accessible.

NOTE

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

How Hot-Relocation Works

The hot-relocation feature is enabled by default. No system administrator action is needed to start hot-relocation when a failure occurs.

The hot-relocation daemon, `vxrelocd`, monitors Volume Manager for events that affect redundancy and performs hot-relocation to restore redundancy. The `vxrelocd` daemon also notifies the system administrator (via electronic mail) of failures and any relocation and recovery actions. See the `vxrelocd(1M)` manual page for more information on `vxrelocd`.

The `vxrelocd` daemon starts during system startup and monitors the Volume Manager for failures involving disks, plexes, or RAID-5 subdisks.

When a failure occurs, it triggers a hot-relocation attempt.

A successful hot-relocation process involves:

- Step 1.** Detecting Volume Manager events resulting from the failure of a disk, plex, or RAID-5 subdisk.
- Step 2.** Notifying the system administrator (and other designated users) of the failure and identifying the affected Volume Manager objects. This is done through electronic mail.
- Step 3.** Determining which subdisks can be relocated, finding space for those subdisks in the disk group, and relocating the subdisks. Notifying the system administrator of these actions and their success or failure.
- Step 4.** Initiating any recovery procedures necessary to restore the volumes and data. Notifying the system administrator of the outcome of the recovery attempt.

NOTE

Hot-relocation does not guarantee the same layout of data or the same performance after relocation. The system administrator may make some configuration changes after hot-relocation occurs.

How Space is Chosen for Relocation

A spare disk must be initialized and placed in a disk group as a spare *before* it can be used for replacement purposes. If no disks have been designated as spares when a failure occurs, Volume Manager automatically uses any available free space in the disk group in which the failure occurs. If there is not enough spare disk space, a combination of spare space and free space is used.

The free space mentioned in hot-relocation is always the free space not excluded from hot-relocation use. Disks can be excluded from hot-relocation use by using the Storage Administrator interface: `vxdiskadm` or `vxedit`.

The system administrator can designate one or more disks as hot-relocation spares within each disk group. Disks can be designated as spares by using the Storage Administrator interface, `vxdiskadm`, or `vxedit`. Disks designated as spares do not participate in the free space model and should not have storage space allocated on them.

When selecting space for relocation, hot-relocation preserves the redundancy characteristics of the Volume Manager object that the relocated subdisk belongs to. For example, hot-relocation ensures that subdisks from a failed plex are not relocated to a disk containing a mirror of the failed plex. If redundancy cannot be preserved using any available spare disks and/or free space, hot-relocation does not take place. If relocation is not possible, the system administrator is notified and no further action is taken.

From the eligible disks, hot-relocation attempts to use the disk that is “closest” to the failed disk. The value of “closeness” depends on the controller, target, and disk number of the failed disk. A disk on the same controller as the failed disk is closer than a disk on a different controller; a disk under the same target as the failed disk is closer than one on a different target.

Hot-relocation tries to move all subdisks from a failing drive to the same destination disk, if possible.

When hot-relocation takes place, the failed subdisk is removed from the configuration database and Volume Manager ensures that the disk space used by the failed subdisk is not recycled as free space.

For information on hot-relocation, see Chapter 4, “Disk Tasks.”

Modifying the vxrelocd Command

Hot-relocation is turned on as long as the vxrelocd procedure is running. Leave hot-relocation turned on so that you can take advantage of this feature if a failure occurs. Disabling this feature (because you do not want the free space on some of your disks used for relocation) prevents the vxrelocd procedure from starting at system startup time.

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).

See “vxrelocd Command” for more information.

vxunreloc Unrelocate Utility

VxVM hot-relocation allows the system to automatically react to I/O failures on a redundant VxVM object at the subdisk level and take necessary action to make the object available again. This mechanism detects I/O failures in a subdisk, relocates the subdisk, and recovers the plex associated with the subdisk. After the disk has been replaced,

Volume Manager provides the `vxunreloc` utility, which can be used to restore the system to the same configuration that existed before the disk failure. The `vxunreloc` utility allows you to move the hot-relocated subdisks back onto a disk that was replaced due to a disk failure.

When the `vxunreloc` utility is invoked, you must specify the disk media name where the hot-relocated subdisks originally resided so that when `vxunreloc` moves the subdisks, it moves them to the original offsets. If you try to unrelocate to a disk that is smaller than the original disk that failed, the `vxunreloc` utility does nothing except return an error.

The `vxunreloc` program provides an option to move the subdisks to a different disk from where they were originally relocated. It also provides an option to unrelocate subdisks to a different offsets if the destination disk is large enough to accommodate all of the subdisks.

If the `vxunreloc` utility cannot replace the subdisks back to the same original offsets, a `force` option is available that allows you to move the subdisks to a specified disk without using the original offsets. See the `vxunreloc(1M)` manual page for more information.

The following examples demonstrate the use of the `vxunreloc` program.

Example 1:

Assume that `disk01` failed and all the subdisks were relocated. After `disk01` is replaced, use `vxunreloc` to move all the hot-relocated subdisks back to `disk01`, as follows:

```
# vxunreloc -g newdg disk01
```

Example 2:

The `vxunreloc` utility provides the `-n` option to move the subdisks to a different disk from where they were originally relocated.

Assume that `disk01` failed, and that all of the subdisks that resided on it were hot-relocated to other disks. After the disk is repaired, it is added back to the disk group using a different name, e.g, `disk05`. If you want to move all the hot-relocated subdisks back to the new disk, use the following command:

```
# vxunreloc -g newdg -n disk05 disk01
```

Example 3:

Assume that `disk01` failed and the subdisks were relocated and that you

want to move the hot-relocated subdisks to disk05 where some subdisks already reside. Use the force option to move the hot-relocated subdisks to disk05, but not to the exact offsets:

```
# vxunreloc -g newdg -f -n disk05 disk01
```

Example 4:

If a subdisk was hot-relocated more than once due to multiple disk failures, it can still be unrelocated back to its original location. For instance, if disk01 failed and a subdisk named disk01-01 was moved to disk02, and then disk02 experienced disk failure, all of the subdisks residing on it, including the one which was hot-relocated to it, will be moved again. When disk02 is replaced, a vxunreloc operation for disk02 does nothing to the hot-relocated subdisk disk01-01. However, a replacement of disk01, followed by a vxunreloc operation, moves disk01-01 back to disk01 if the vxunreloc utility is run immediately after the replacement.

After the disk that experienced the failure is fixed or replaced, the vxunreloc program can be used to move all the hot-relocated subdisks back to the disk. When a subdisk is hot-relocated, its original disk media name and the offset into the disk, are saved in the configuration database. When a subdisk is moved back to the original disk or to a new disk using the vxunreloc utility, the information is erased. The original dm name and the original offset are saved in the subdisk records. To print all of the subdisks that were hot-relocated from disk01 in the rootdg disk group, use the following command:

```
# vxprint -g rootdg -se 'sd_orig_dmname="disk01"'
```

To move all the subdisks that were hot-relocated from disk01 back to the original disk, use the following command:

```
# vxunreloc -g rootdg disk01
```

The vxunreloc utility provides -n option to move the subdisks to a different disk from where they were originally relocated. For example, when disk01 failed, all the subdisks that resided on it were hot-relocated to other disks. After the disk is repaired, it is added back to the disk group using a different name, for example, disk05. To move all the hot-relocated subdisks to the new disk, use the following command:

```
# vxunreloc -g rootdg -n disk05 disk01
```

The destination disk should have at least as much storage capacity as was in use on the original disk. If there is not enough space, the

unrelocate operation will fail and none of the subdisks will be moved.

When the `vxunreloc` program moves the hot-relocated subdisks, it moves them to the original offsets. However, if there some subdisks existed which occupied part or all of the area on the destination disk, the `vxunreloc` utility will fail. If failure occurs, you have two choices: (1) move the existing subdisks somewhere else, and then re-run the `vxunreloc` utility, or (2) use the `-f` option provided by the `vxunreloc` program to move the subdisks to the destination disk, but allow the `vxunreloc` utility to find the space on the disk. As long as the destination disk is large enough so that the region of the disk for storing subdisks can accommodate all subdisks, all the hot-relocated subdisks will be “unrelocated” without using the original offsets.

A subdisk that was hot-relocated more than once due to multiple disk failures will still be able to be unrelocated back to its original location. For instance, if `disk01` failed and a subdisk named `disk01-01` was moved to `disk02`, and then `disk02` experienced disk failure, all the subdisks residing on it, including the one which was hot-relocated to it, will be moved again. When `disk02` is replaced, an unrelocate operation for `disk02` will not do anything to the hot-relocated subdisk `disk01-01`. However, a replacement of `disk01` followed by the unrelocate operation moves `disk01-01` back to `disk01` when the `vxunreloc` program is run, immediately after the replacement.

Restarting the vxunreloc Utility After Errors

Internally, the `vxunreloc` program moves the subdisks in three phases. The first phase is creates as many subdisks on the specified destination disk as there are the number of the subdisks to be unrelocated. When the subdisks are made, the `vxunreloc` program fills in the comment field in the subdisk record with the string `UNRELOC` as an identification. The second phase is the actual data moving. If all the subdisk moves are successful, the third phase proceeds to clean up the comment field of the subdisk records.

Making the subdisk is a all-or-none operation. If the `vxunreloc` program cannot make all the subdisks successfully, no subdisk is made and the `vxunreloc` program exits. The operation of the subdisks move is not all-or-none. One subdisk move is independent of another, and as a result, if one subdisk move fails, the `vxunreloc` utility prints an error message and then exits. But, all of the subsequent subdisks remain on the disk where they were hot-relocated and will not be moved back. For subdisks that made their way back home, the comment field in their subdisk

records is still marked as UNRELOC because the cleanup phase is never executed.

If the system goes down after the new subdisks are made on the destination, but before they are moved back, the vxunreloc program can be executed again after the system comes back. As described above, when a new subdisk is created, the vxunreloc program sets the comment field of the subdisk as UNRELOC. When the vxunreloc utility is re-executed, it checks the offset, the len, and the comment fields of the existing subdisks on the destination disk to determine if it was left on the disk at a previous execution of the vxunreloc program, which then uses it as it sees fit.

Do not manually modify the string UNRELOC in the comment field.

If one out of a series of subdisk moves fails, the vxunreloc program exits. Under this circumstance, you should check the error that caused the subdisk move to fail and determine if the unrelocation can proceed. When you re-execute the vxunreloc utility to resume the subdisk moves, it uses the subdisks created at a previous run.

The cleanup phase is done with one transaction. The vxunreloc program resets the comment field to a NULL string for all the subdisks marked as UNRELOC that reside on the destination disk. This includes clean up for those subdisks that were unrelocated in any previous invocation of the vxunreloc program, that was not successfully completed.

Volume Resynchronization

When storing data redundantly, using mirrored or RAID-5 volumes, the Volume Manager ensures that all copies of the data match exactly. However, under certain conditions (usually due to complete system failures), some redundant data on a volume can become inconsistent or *unsynchronized*. The mirrored data is not exactly the same as the original data. Except for normal configuration changes (such as detaching and reattaching a plex), this can only occur when a system crashes while data is being written to a volume.

Data is written to the mirrors of a volume in parallel, as is the data and parity in a RAID-5 volume. If a system crash occurs before all the individual writes complete, it is possible for some writes to complete while others do not. This can result in the data becoming unsynchronized. For mirrored volumes, it can cause two reads from the same region of the volume to return different results, if different mirrors are used to satisfy the read request. In the case of RAID-5 volumes, it can lead to parity corruption and incorrect data reconstruction.

The Volume Manager needs to ensure that all mirrors contain exactly the same data and that the data and parity in RAID-5 volumes agree. This process is called **volume resynchronization**. For volumes that are part of disk groups that are automatically imported at boot time (such as rootdg), the resynchronization process takes place when the system reboots.

Not all volumes require resynchronization after a system failure. Volumes that were never written or that were quiescent (that is, had no active I/O) when the system failure occurred could not have had outstanding writes and do not require resynchronization.

The Volume Manager records when a volume is first written to and marks it as **dirty**. When a volume is closed by all processes or stopped cleanly by the administrator, all writes have been completed and the Volume Manager removes the dirty flag for the volume. Only volumes that are marked dirty when the system reboots require resynchronization.

The process of resynchronization depends on the type of volume. RAID-5 volumes that contain RAID-5 logs can “replay” those logs. If no logs are available, the volume is placed in reconstruct-recovery mode and all parity is regenerated. For mirrored volumes, resynchronization is done

by placing the volume in recovery mode (also called **read-writeback recovery mode**). Resynchronization of data in the volume is done in the background. This allows the volume to be available for use while recovery is taking place.

The process of resynchronization can be expensive and can impact system performance. The recovery process reduces some of this impact by spreading the recoveries to avoid stressing a specific disk or controller.

For large volumes or for a large number of volumes, the resynchronization process can take time. These effects can be addressed by using Dirty Region Logging for mirrored volumes, or by ensuring that RAID-5 volumes have valid RAID-5 logs. For volumes used by database applications, the VxSmartSync™ Recovery Accelerator can be used (see “VxSmartSync Recovery Accelerator”).

Dirty Region Logging

NOTE

You must license the VERITAS Volume Manager product to use this feature.

Dirty Region Logging (DRL) is an optional property of a volume, used to provide a speedy recovery of mirrored volumes after a system failure. DRL keeps track of the regions that have changed due to I/O writes to a mirrored volume. DRL uses this information to recover only the portions of the volume that need to be recovered.

If DRL is not used and a system failure occurs, all mirrors of the volumes must be restored to a consistent state. Restoration is done by copying the full contents of the volume between its mirrors. This process can be lengthy and I/O intensive. It may also be necessary to recover the areas of volumes that are already consistent.

DRL logically divides a volume into a set of consecutive regions. It keeps track of volume regions that are being written to. A dirty region log is maintained that contains a status bit representing each region of the volume. For any write operation to the volume, the regions being written are marked dirty in the log before the data is written. 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 that are marked as dirty in the dirty region log.

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; multiple log subdisks can be used to mirror the dirty region log. Each log subdisk is associated with one plex of the volume. Only one log subdisk can exist per plex. If the plex contains only a log subdisk and no data subdisks, that plex can be referred to as a **log plex**.

The log subdisk can also be associated with a regular plex containing data subdisks. In that case, the log subdisk risks becoming unavailable if the plex must be detached due to the failure of one of its data subdisks.

If the `vxassist` command is used to create a dirty region log, it creates a log plex containing a single log subdisk by default. A dirty region log can

also be created manually by creating a log subdisk and associating it with a plex. Then the plex can contain both a log subdisk and data subdisks.

Only a limited number of bits can be marked dirty in the log at any time. The dirty bit for a region is not cleared immediately after writing the data to the region. Instead, it remains marked as dirty until the corresponding volume region becomes the least recently used. If a bit for a given region is already marked dirty and another write to the same region occurs, it is not necessary to write the log to the disk before the write operation can occur.

Some volumes, such as those used for Oracle replay logs, are written sequentially and do not benefit from this lazy cleaning of the DRL bits. For these volumes, *sequential DRL* can be used to further restrict the number of dirty bits and speed up recovery. The number of dirty bits allowed for sequential DRL is restricted by the tunable `voldr1_max_dirty`. Using sequential DRL on volumes that are written sequentially may severely impact I/O throughput.

NOTE

DRL adds a small I/O overhead for most write access patterns.

FastResync (Fast Mirror Resynchronization)

NOTE

You may need an additional license to use this feature.

The FastResync (also called Fast Mirror Resynchronization, which is abbreviated as FMR) feature performs quick and efficient resynchronization of stale mirrors by increasing the efficiency of the VxVM snapshot mechanism to better support operations such as backup and decision support. Typically, these operations require that the data store volume is quiescent and/or secondary access to the store not affect or impede primary access (for example, throughput, updates, consistency, and so on).

To achieve these goals, VxVM currently provides a *snapshot* mechanism that creates an exact copy of a primary volume at a particular instance in time. After a snapshot is taken, it can be accessed independently of the volume from which it was taken. In a shared/clustered VxVM environment, it is possible to eliminate the resource contention and overhead of using the snapshot simply by accessing it from a different machine.

Fast Mirror Resynchronization overcomes certain drawbacks that exist in an earlier version of the snapshot mechanism. They are:

- After a snapshot is taken, the primary volume and snapshot can diverge. They are no longer consistent. As a result, the snapshot must be discarded when it is no longer useful, and a new snapshot must be taken in order to obtain a current copy of the primary data.
- Snapshot setup time can limit the usefulness of the snapshot feature. This is because the period in which a snapshot can be created is directly proportional to the size of the volume. For large, enterprise class volumes, this period can dictate off-line policies in unacceptable ways.

FMR Components

FMR provides two fundamental enhancements to VxVM. The first is to optimize the mirror resynchronization process (**Fast Mirror**

Resynchronization), and the second is to extend the snapshot model (**Fast Mirror Reconnect**) to provide a method by which snapshots can be refreshed and re-used rather than discarded.

Fast Mirror Resynchronization Component

The former enhancement, Fast Mirror Resynchronization (FMR), requires keeping track of data store updates missed by mirrors that were unavailable at the time the updates were applied. When a mirror returns to service, it must re-apply only the updates missed by that mirror. With FMR, this process requires an amount of restoration far less than the current method whereby the entire data store is copied to the returning mirror.

An unavailable mirror is one that has been *detached* from its volume either automatically (by VxVM, as the result of an error), or directly by an administrator (via a VxVM utility such as the `vxplex` or `vxassist` commands). A **returning mirror** is a mirror that was previously detached and is in the process of being re-attached to its original volume as the result of the `vxrecover` or `vxplex att` operation.

FMR will not alter the current mirror failure and repair administrative model. The only visible effect is that typical mirror repair operations conclude expediently.

The resynchronization enhancement allows the administrator to enable/disable FMR on a per-volume basis, and to check FMR status.

Fast Mirror Reconnect Component

Fast Mirror Reconnect augments the existing snapshot usage model. Without Fast Mirror Resynchronization, an independent copy of a volume is created via the snapshot mechanism. The original volume and the snap volume are completely independent of each other and, their data may diverge.

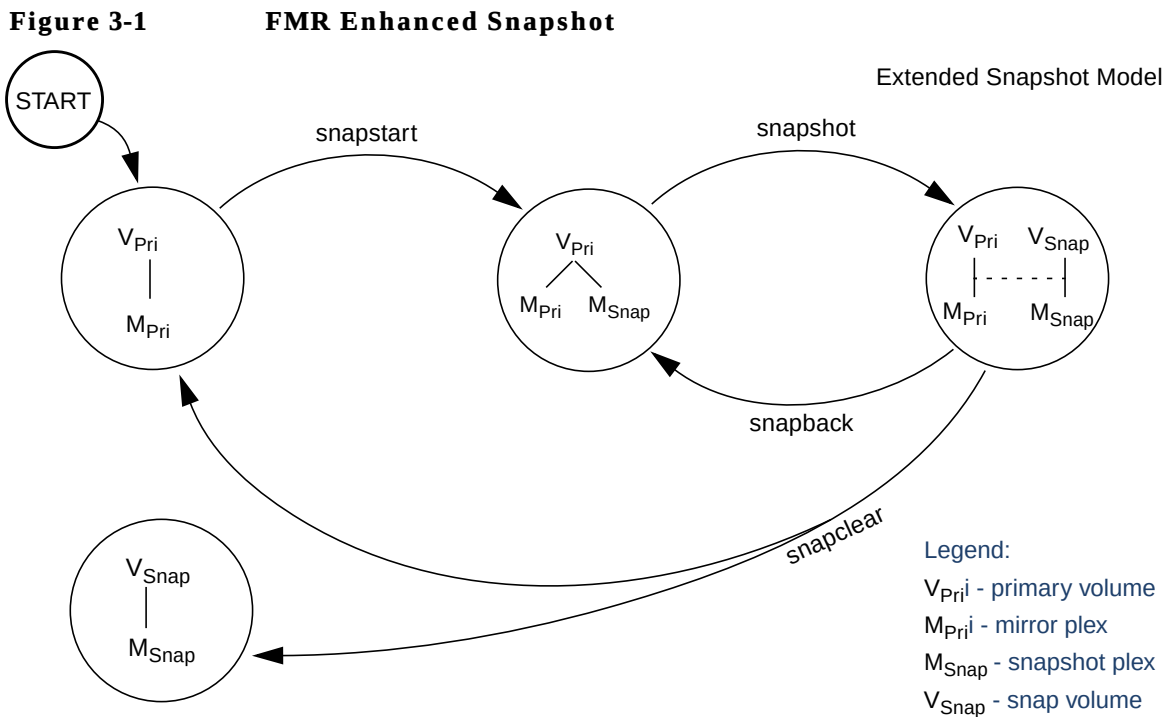
The *Fast Mirror Reconnect* snapshot enhancement makes it possible to re-associate a snapshot volume with its original peer for the express purpose of reducing the work-load required to perform cyclic operations which rely heavily upon the VxVM snapshot functionality.

FMR Enhancements to VxVM Snapshot Functionality

The FMR snapshot enhancements to Release 3.1 extend the snapshot model as shown in Figure 3-1, FMR Enhanced Snapshot,. Beginning

with Release 3.1, the `snapshot` command behaves as before with the exception that it creates an association between the original volume and the snap volume. A new command, `vxassist snapback`, leverages this association to expediently return the snapshot plex (M_{Snap}) to the volume from which it was snapped (in this example, V_{Pri}).

Figure 3-1, FMR Enhanced Snapshot, depicts the extended transitions of the snapshot model as introduced by the `snapback` and `snapclear` commands.



Additionally, a new command, `vxassist snapclear`, relieves a volume of the administrative overhead of tracking a snapshot by permanently destroying the association created by the `snapshot` command. This capability is useful in situations where it is known that a snapshot will never return to the volume from which it was created.

Theory of Operations

The basis of FMR lies in change tracking. Keeping track of updates

missed when a mirror is offline/detached/shapshotted, and then applying only those updates when the mirror returns, considerably reduces the time to resynchronize the volume.

The basis for this change tracking is the use of a bitmap. Each bit in the bitmap represents a contiguous region (an extent) of a volume's address space. This contiguous region is called the **region size**. Typically, this region size is one block; each block of the volume is represented by one bit in the bitmap. However, a tunable called `vol_fmr_logsz` is provided, which can be used to limit the maximum size (in blocks) of the FMR map. When computing the size of the map, the algorithm starts with a region size of one, and if the resulting map size is less than the `vol_fmr_logsz`, then the computed value becomes the map size. If the size is larger than `vol_fmr_logsz`, an attempt is made to accommodate `vol_fmr_logsz` with a region size of two, and so on until the map size is less than the `vol_fmr_logsz` tunable.

For example:

volume size = 1G

`vol_fmr_logsz` = 4

on a system with a block size of 1024 bytes, that is $4 * 1024 = 4096$ bytes or $4096 * 8 = 32768$ bits.

Thus, for a 1G volume, a region size of one is 4096 bits, which is less than four blocks, so the map size is 4096 bits or 512 bytes.

Note that if the size of the volume increases, this computation is redone to ensure the map size does not exceed the `vol_fmr_logsz` tunable.

Persistent Versus Non-Persistent Tracking

For VxVM 3.1, the FMR maps are allocated in memory. They do not reside on disk or persistent store, unlike a DRL. Therefore, if the system crashes, this information is lost and the entire length of the volume must be synced up. One advantage of this approach is that the FMR updates (updates to this map) do not cost anything in terms of performance, as no disk updates must be done. However, if the system crashes, this information is lost and full resynchronization of mirrors is once again necessary.

Snapshot(s) and FMR

To take advantage of the FMR delta tracking when using snapshots, use the new snapshot option. After a snapshot is taken, the snapshot option

is used to reattach the snapshot plex. If FMR is enabled before the snapshot is taken and is not disabled at any time before the snapshot is complete, then the FMR delta changes reflected in the FMR bitmap are used to resynchronize the volume during the snapback. To make it easier to create snapshots of several volumes at the same time, the snapshot option has been enhanced to accept more than one volume and a naming scheme has been added. By default, each replica volume is named **SNAP-*<original volume>***. This default can be overridden with options on the command line. Only volumes in the same disk group can take a snapshot of more than one volume at a time.

To make it easier to snapshot all the volumes in a single disk group, the option **-o resyncfromreplica** has been added to **vxassist**. However, it fails if any of the volumes in the disk group do not have a complete snapshot plex. It is possible to take several snapshots of the same volume. A new FMR bitmap is produced for each snapshot taken, and the resynchronization time for each snapshot is minimized.

The snapshot plex can be chosen as the preferred set of data when performing a snapback. Adding **-o resyncfromreplica** to the snapback option copies the data on the snapshot (replica) plex onto all the mirrors attached to the original volume. By default, the data on the original volume is preferred and copied onto the snapshot plex.

It is possible to grow the replica volume, or the original volume, and still use FMR. Growing the volume extends the bitmap FMR uses to track the delta changes. This may change the size of the bitmap or its region size. In either case, the part of the bitmap that corresponds to the grown area of the volume is marked as “dirty” so that this area is resynchronized. The snapshot operation fails if the snapshot attempts to create an incomplete snapshot plex. In such cases, it is necessary to grow the replica volume, or the original volume, before the snapback option is run. Growing the two volumes separately can lead to a snapshot that shares physical disks with another mirror in the volume. To prevent this, grow the volume after the snapback command is complete.

Any operation that changes the layout of the replica volume can mark the FMR map for that snapshot “dirty” and require a full resynchronization during the snapback. Operations that cause this include subdisk split, subdisk move, and online relay of the replica. It is safe to perform these operations after the snapshot is completed. For more information, see the **vxvol (1M)**, **vxassist (1M)**, and **vplex (1M)** manual pages.

FMR and Writable Snapshots

One of two options is used to track changes to a writable snapshot, as follows:

- create a separate map that tracks changes to a snapshot volume
- update the map of the parent of the snapshot volume. Use this shortcut method only if there are few updates to the snapshot volume, such as in the backup and DSS (decision support systems) applications

For VxVM 3.1, the latter method is implemented; i.e., the map of the parent of the snapshot volume is updated when writing to the snapshot.

Caveats and Limitations

FMR is not supported on RAID-5 volumes.

When a subdisk is relocated, the entire plex is marked “dirty” and a full resynchronization becomes necessary.

FMR Enhancements to VxVM Snapshot Functionality

The FMR snapshot enhancements to Release 3.1 extend the snapshot model as shown in the diagram below. Beginning with Release 3.1, the snapshot command behaves as before with the exception that it creates an association between the original volume and the snap volume. A new command, vxassist snapback leverages this association to expediently return the snap mirror (M_{Snap}) to the volume from which it was snapped (in this example, V_{Pri}).

Additionally, a new command relieves a volume of the administrative overhead of tracking a snapshot (it would permanently destroy the association created by the snapshot command). This capability is useful in situations where it is known that a snapshot will never return to the volume from which it was created. To achieve this result, the command vxassist snapclear has been added to the family of snapshot commands.

Volume Manager Rootability

Rootability is the term used to indicate that the logical volumes containing the root file system and the system swap area are under Volume Manager control. Normally the Volume Manager is started following a successful boot after the operating system has passed control to the initial user mode process. However, when the volume containing the root file system is under Volume Manager control, portions of the Volume Manager must be started early from the operating system kernel before the operating system starts the first user process. Thus, the Volume Manager code that enables rootability is contained within the operating system kernel.

An HP-UX boot disk is set up to contain a Logical Interchange format (LIF) area. Part of the LIF structure is a LIF LABEL record, which contains information about the starting block number and length of the volumes containing the stand and root filesystems as well as the volume containing the system swap area. Part of the procedure for making a disk VxVM-rootable is to initialize the LIF LABEL record with volume extent information for the stand, root, swap, and, optionally, dump volumes. This is done with *vxbootsetup* (1M).

Booting With Root Volumes

Before the kernel mounts the root file system, it determines if the boot disk was a rootable VxVM disk. If so, then the kernel passes control to the kernel Volume Manager rootability code. The kernel rootability code extracts the starting block number and length of the root and swap volumes from the LIF LABEL record and builds “fake” volume and disk configuration objects for these volumes and then loads this fake configuration into the VxVM kernel driver. At this point, I/O can proceed on these fake root and swap volumes by simply referencing the device number that was set up by the rootability code.

Once the kernel passes control to the initial user procedure (*pre_init_rc()*), the Volume Manager daemon starts and reads the configuration of the volumes in the root disk group and loads them into the kernel. At this time the fake root and swap volume hierarchy can be discarded, as further I/O to these volumes will be done through the normal configuration objects just loaded into the kernel.

Boot Time Volume Restrictions

The volumes that need to be available at boot time have some very specific restrictions on their configuration. These restrictions include their names, the disk group they are in, their volume usage types, and they must be single subdisk, contiguous volumes. These restrictions are detailed below:

- Disk Group

All volumes on the boot disk must be in the `rootdg` disk group.

- Names

The names of the volumes that will have entries in the LIF LABEL record must be `rootvol`, `standvol`, and `swapvol`. If there is an optional dump volume to be added to the LIF LABEL, then its name must be `dumpvol`.

- Usage Types

The `rootvol` and `swapvol` volumes have specific volume usage types named usage type `root` and `swap` respectively.

- Contiguous Volumes

Any volume that will have an entry in the LIF LABEL record must be contiguous. It can have only one subdisk and it cannot span to another disk.

- Mirrored Root Volumes

All the volumes on the boot disk can be mirrored. If you want the mirror of the boot disk also to be bootable, then the above restrictions apply to the mirror disk as well. A VxVM-rootable boot disk can be mirrored with `vxrootmir` (1M). You can set up mirrors of selected volumes on the boot disk for enhanced performance (for example, striped or spanned), but the resultant mirrors will not be bootable.

Dynamic Multipathing (DMP)

NOTE

You may need an additional license to use this feature.

On some systems, the Volume Manager supports multiported disk arrays. It automatically recognizes multiple I/O paths to a particular disk device within the disk array. The Dynamic Multipathing feature of the Volume Manager provides greater reliability by providing a path failover mechanism. In the event of a loss of one connection to a disk, the system continues to access the critical data over the other sound connections to the disk. DMP also provides greater I/O throughput by balancing the I/O load uniformly across multiple I/O paths to the disk device.

In the Volume Manager, all the physical disks connected to the system are represented as metadevices with one or more physical access paths. A single physical disk connected to the system is represented by a metadevice with one path. A disk that is part of a disk array is represented by a metadevice that has two physical access paths. You can use the Volume Manager administrative utilities such as the `vxdisk` utility to display all the paths of a metadevice and status information of the various paths.

Path Failover Mechanism

DMP enhances system reliability when used with multiported disk arrays. In the event of the loss of one connection to the disk array, DMP automatically selects the next I/O paths for the I/O requests dynamically without action from the administrator.

DMP allows the administrator to indicate to the DMP subsystem in the Volume Manager whether the connection is repaired or restored. This is called DMP reconfiguration. The reconfiguration procedure also allows the detection of newly added devices, as well as devices that are removed after the system is fully booted (if the operating system detects them properly).

Load Balancing

To provide load balancing across paths, DMP follows the **balanced path mechanism** for active/active disk arrays. Load balancing makes sure that I/O throughput can be increased by utilizing the full bandwidth of all paths to the maximum.

Sequential I/Os starting within a certain range will be sent down the same path in order to optimize I/O throughput by utilizing the effect of disk track caches. However, large sequential I/Os that do not fall within this range will be distributed across paths to take the advantage of I/O load balancing. For active/passive disk arrays, I/Os are sent down the primary path until it fails. Once the primary path fails, I/Os are then switched over to the other available primary paths or secondary paths. Load balancing across paths is not done for active/passive disk arrays to avoid the continuous transfer of ownership of LUNs from one controller to another, thus resulting in severe I/O slowdown.

Disabling and Enabling DMP

NOTE

After either disabling or enabling DMP, you must reboot the system for the changes to take effect.

Disabling DMP on an HP System: Step 1. Stop the vxconfigd daemon using the following command:

```
# vxdctl stop
```

Step 2. Save the /stand/system file as /stand/system.vxdmp by executing the command:

```
# cp /stand/system /stand/system.vxdmp
```

Step 3. Save the /stand/vmunix file as /stand/vmunix.vxdmp by executing the command:

```
# cp /stand/vmunix /stand/vmunix.vxdmp
```

Step 4. Edit the /stand/system file and remove the vxdmp entry.

Step 5. Run the following script

```
# /etc/vx/bin/vxdmpdis
```

If all the above steps complete successfully, reboot the system. When the system comes up, DMP should be removed completely from the system. Verify that DMP was removed by running the `vxdmpadm` command. The following message is displayed:

```
vxvm:vxdmpadm: ERROR: vxdmp module is not loaded on the  
system. Command invalid.
```

Also the command `vxdisk list <da_name>` should not display any multipathing information.

Enabling DMP on a HP System (Where it is Currently Disabled): Step 1. Stop the `vxconfigd` daemon using the following command:

```
# vxdctl stop
```

Step 2. Save the `/stand/system` file as `/stand/system.vxdmpdis` by executing the command:

```
# cp /stand/system /stand/system.vxdmpdis
```

Step 3. Save the `/stand/vmunix` file as `/stand/vmunix.vxdmpdis` by executing the command:

```
# cp /stand/vmunix /stand/vmunix.vxdmpdis
```

Step 4. Edit the `/stand/system` file and add the `vxdmp` entry to it (after the `vxvm` entry).

Step 5. Run the following script:

```
# /etc/vx/bin/vxdmpen
```

After the above steps complete successfully, reboot the system. When the system comes up, DMP should be enabled on the system. Verify this by running the `vxdmpadm` command. This command shows multipathing information. Also, the command `vxdisk list <da_name>` shows multipathing information.

Input/Output (I/O) Controllers

This feature allows the administrator to turn off I/Os to a host I/O controller to perform administrative operations. It can be used for

maintenance of controllers attached to the host or a disk array supported by the Volume Manager. I/O operations to the host I/O controller can be turned on after the maintenance task is completed. This once again enables I/Os to go through this controller. This operation can be accomplished using the `vxddmpadm(1M)` utility provided with Volume Manager.

For active/active type disk arrays, Volume Manager uses the **balanced path mechanism** to schedule I/Os to a disk with multiple paths to it. As a result, I/Os may go through any path at any given point in time.

For active/passive type disk arrays, I/Os will be scheduled by Volume Manager to the primary path until a failure is encountered. Therefore, to change an interface card on the disk array or a card on the host (when possible) that is connected to the disk array, the I/O operations to the host I/O controller(s) should be disabled. This allows all I/Os to be shifted over to an active secondary path or an active primary path on another I/O controller before the hardware is changed.

After the operation is over, the paths through these controller(s) can be put back into action by using the enabled option of the `vxddmpadm(1M)` command.

Volume Manager does not allow you to disable the last active path to the rootdisk.

Displaying DMP Database Information

The `vxddmpadm(1M)` utility can be used to list DMP database information and perform other administrative tasks. This command allows the user to list all the controllers on the systems (connected to disks) and other related information stored in the DMP database. This information can be used to locate system hardware and also make a decision regarding which controllers to enable/disable.

It also provides you with other useful information such as disk array serial number and the list of DMP devices (disks) that are connected to the disk array, the list of paths that go through a particular controllers, etc.

VxSmartSync Recovery Accelerator

The VxSmartSync™ Recovery Accelerator is available for some systems. VxSmartSync for Mirrored Oracle® Databases is a collection of features that speed up the resynchronization process (known as **resilvering**) for volumes used in with the Oracle Universal Database™. These features use an extended interface between Volume Manager volumes and the database software so they can avoid unnecessary work during mirror resynchronization. These extensions can result in an order of magnitude improvement in volume recovery times.

Oracle automatically takes advantage of SmartSync when it is available.

The system administrator must configure the volumes correctly to use VxSmartSync. For Volume Manager, there are two types of volumes used by the database:

- *redo log volumes* contain redo logs of the database.
- *data volumes* are all other volumes used by the database (control files and tablespace files).

VxSmartSync works with these two types of volumes differently, and they must be configured correctly to take full advantage of the extended interfaces. The only difference between the two types of volumes is that redo log volumes should have dirty region logs, while data volumes should not.

Data Volume Configuration

The improvement in recovery time for data volumes is achieved by letting the database software decide which portions of the volume require recovery. The database keeps logs of changes to the data in the database and can determine which portions of the volume require recovery. By reducing the amount of space that requires recovery and allowing the database to control the recovery process, the overall recovery time is reduced.

Also, the recovery takes place when the database software is started, not at system startup. This reduces the overall impact of recovery when the system reboots. Because the recovery is controlled by the database, the recovery time for the volume is the resilvering time for the database (that is, the time required to replay the redo logs).

Because the database keeps its own logs, it is not necessary for Volume Manager to do logging. Data volumes should therefore be configured as mirrored volumes *without* dirty region logs. In addition to improving recovery time, this avoids any run-time I/O overhead due to DRL, which improves normal database write access.

Redo Log Volume Configuration

A **redo log** is a log of changes to the database data. No logs of the changes to the redo logs are kept by the database, so the database itself cannot provide information about which sections require resilvering. Redo logs are also written sequentially, and since traditional dirty region logs are most useful with randomly-written data, they are of minimal use for reducing recovery time for redo logs. However, Volume Manager can reduce the number of dirty regions by modifying the behavior of its Dirty Region Logging feature to take advantage of sequential access patterns. This decreases the amount of data needing recovery and reduces recovery time impact on the system.

The enhanced interfaces for redo logs allow the database software to inform Volume Manager when a volume is to be used as a redo log. This allows Volume Manager to modify the DRL behavior of the volume to take advantage of the access patterns. Since the improved recovery time depends on dirty region logs, redo log volumes should be configured as mirrored volumes *with* dirty region logs.

Common Volume Manager Commands

vxedit Command

The common command used to remove or rename Volume Manager objects (volumes, plexes, and subdisks) is the `vxedit` command. The `vxedit` command has two functions:

- it allows you to modify certain records in the volume management databases. Only fields that are not volume usage-type-dependent can be modified.
- it can remove or rename Volume Manager objects.

Volume Manager objects that are associated with other objects are not removable by the `vxedit` command. This means that the `vxedit` command cannot remove:

- a subdisk that is associated with a plex
- a plex that is associated with a volume

NOTE

Using the recursive suboption (`-r`) to the removal option of the `vxedit` command removes all objects from the specified object downward. In this way, a plex and its associated subdisks, or a volume and its associated plexes and their associated subdisks, can be removed by a single `vxedit` command.

For detailed information, see the `vxedit(1M)` manual page.

vxtask Utility

The `vxtask` utility performs basic administrative operations on Volume Manager tasks that are running on the system. Operations include listing tasks, modifying the state of a task (pausing, resuming, aborting) and modifying the rate of progress of a task. See the `vxtask(1M)` manual page for more information.

Volume Manager **tasks** represent long-term operations in progress on the system. Every task gives information on the time the operation

started, the size and progress of the operation, and the state and rate of progress of the operation. The administrator can change the state of a task, giving coarse-grained control over the progress of the operation. For those operations that support it, the rate of progress of the task can be changed, giving more fine-grained control over the task.

Every task is given a unique **task identifier**. This is a numeric identifier for the task that can be specified to the vxtask utility to specifically identify a single task. For most utilities, the tag is specified with the -t tag option.

Operations The vxtask utility supports the following operations:

list

List tasks running on the system in one-line summaries. The -l option prints tasks in long format. The -h option prints tasks hierarchically, with child tasks following the parent tasks. By default, all tasks running on the system are printed. If a taskid argument is supplied, the output is limited to those tasks whose taskid or task tag match taskid. The remaining arguments are used to filter tasks and limit the tasks actually listed.

monitor

The monitor operation causes information about a task or group of tasks to be printed continuously as task information changes. This allows the administrator to track progress on an ongoing basis. Specifying -l causes a long listing to be printed. By default, short one-line listings are printed. In addition to printing task information when a task state changes, output is also generated when the task completes. When this occurs, the state of the task is printed as EXITED (see “Output”).

pause

resume

abort

These three operations request that the specified task change its state. The pause operation puts a running task in the paused state, causing it to suspend operation. The resume operation causes a paused task to continue operation. The abort operation causes the specified task to cease operation. In most cases, the operations “back out” as if an I/O error occurred, reversing to the extent possible what had been done so

far.

set

The **set** operation is used to change modifiable parameters of a task. Currently, there is only one modifiable parameter for tasks: the *slow* attribute, which represents a throttle on the task progress. The larger the slow value, the slower the progress of the task and the fewer system resources it consumes in a given time. (The slow attribute is the same attribute that many commands, such as **vxplex(1M)**, **vxvol(1M)** and **vxrecover(1M)** accept on command lines.)

Output There are two output formats printed by the **vxtask** command: a short, one-line summary format per task, and a long task listing. The short listing provides the most used task information for quick perusal.

The long output format prints all available information for a task, spanning multiple lines. If more than one task is printed, the output for different tasks is separated by a single blank line. Each line in the long output format contains a title for the line, followed by a colon (:), followed by the information.

Examples To list all tasks currently running on the system, use the following command:

```
# vxtask list
```

The **vxtask list** command produces the following output:

```
TASKID  PTID TYPE/STATE    PCT   PROGRESS
    162  RDWRBACK/R 01.10% 0/1048576/11520 VOLSTART
mirrvol-L01
```

To print tasks hierarchically, with child tasks following the parent tasks, use the **-h** option, as follows:

```
# vxtask -h list
```

Use of the **-h** option produces the following output:

```
TASKID  PTID TYPE/STATE    PCT   PROGRESS
    167  ATCOPY/R 06.79% 0/1048576/71232 PLXATT mirrvol
mirrvol-02
```

To trace all tasks in the diskgroup **foodg** that are currently paused, as well as any tasks with the tag **sysstart**, use the following command:

```
# vxtask -G foodg -p -i sysstart list
```

To list all tasks on the system that are currently paused, use the following command:

```
# vxtask -p list
```

The vxtask -p list command is used as follows:

```
# vxtask pause 167
```

```
# vxtask -p list
```

```
TASKID  PTID TYPE/STATE      PCT    PROGRESS
      167  ATCOPY/P 27.82% 0/1048576/291712 PLXATT
mirrvol(mirrvol-02)
```

```
# vxtask resume 167
```

```
# vxtask -p list
```

```
TASKID  PTID TYPE/STATE      PCT    PROGRESS
```

To monitor all tasks with the tag myoperation, use the following command:

```
# vxtask monitor myoperation
```

To cause all tasks tagged with recovall to exit, use the following command:

```
# vxtask abort recovall
```

vxassist Command

You can use the vxassist command to create and manage volumes from the command line. The vxassist command requires minimal input from you and performs many underlying tasks for you. You can use the vxassist command to easily create, mirror, grow, shrink, remove, and back up volumes. The vxassist command is capable of performing many operations that would otherwise require the use of a series of more complicated Volume Manager commands. The vxassist command creates and manipulates volumes based on a set of established defaults, but also allows you to supply preferences for each task.

The vxassist command can perform tasks that would otherwise require the use of several other Volume Manager commands. The vxassist command automatically performs all underlying and related tasks that would otherwise be done by the user (in the form of other commands).

The vxassist command does not conflict with existing Volume Manager commands or preclude their use. Objects created by the vxassist command are compatible and inter-operable with objects created by other VM commands and interfaces.

The `vxassist` command typically takes the following form:

```
# vxassist keyword volume_name [attributes...]
```

Select the specific action to perform by specifying an operation keyword as the first argument on the command line. For example, the keyword for creating a new volume is `make`. You can create a new volume by entering:

```
# vxassist make volume_name length
```

The first argument after any `vxassist` command keyword is always a volume name. Follow the volume name with a set of attributes. Use these attributes to specify where to allocate space and whether you want mirroring or striping to be used.

You can select the disks on which the volumes are to be created by specifying the disk names at the end of the command line. For example, to create a 30 megabyte striped volume on three specific disks (`disk03`, `disk04`, and `disk05`), enter:

```
# vxassist make stripevol 30m layout=stripe disk03  
disk04\disk05
```

The `vxassist` command defaults are listed in the file `/etc/default/vxassist`. The defaults listed in this file take effect if there is no overriding default specified on the `vxassist` command line.

The `vxassist` command performs the following tasks:

- finds space for and creates volumes
- finds space for and creates mirrors for existing volumes
- finds space for and extends existing volumes
- shrinks existing volumes and returns unused space
- provides facilities for the online backup of existing volumes
- provides an estimate of the maximum size for a new or existing volume

For more information, see the `vxassist(1M)` manual page.

Advantages of the `vxassist` Command **NOTE**The `vxassist` command is *required* for `vxrelayout`.

Some of the advantages of using the `vxassist` command include:

The use of the `vxassist` command involves only one step (command) on the part of the user.

You are required to specify only minimal information to the `vxassist` command, yet you can optionally specify additional parameters to modify its actions.

The `vxassist` command tasks result in a set of configuration changes that either succeed or fail as a group, rather than individually. Most `vxassist` tasks work so that system crashes or other interruptions do not leave intermediate states to be cleaned up. If the `vxassist` command finds an error or an exceptional condition, it exits without leaving partially-changed configurations. The system is left in the same state as it was prior to the attempted `vxassist` task.

How the `vxassist` Command Works The `vxassist` command allows you to create and modify volumes. You specify the basic volume creation or modification requirements and the `vxassist` command performs the necessary tasks.

The `vxassist` command obtains most of the information it needs from sources other than your input. The `vxassist` command obtains information about the existing objects and their layouts from the objects themselves.

For tasks requiring new disk space, the `vxassist` command seeks out available disk space and allocates it in the configuration that conforms to the layout specifications and that offers the best use of free space.

The `vxassist` command typically takes this form:

```
# vxassist keyword volume_name [attributes...]
```

where **keyword** selects the task to perform. The first argument after a `vxassist` command keyword is a volume name, which is followed by a set of attributes. For details on available `vxassist` keywords and attributes, see the `vxassist(1M)` manual page.

The `vxassist` command creates and manipulates volumes based on a set of established defaults, but also allows you to supply preferences for each task.

`vxassist` Command Defaults The `vxassist` command uses a set of tunable parameters that can be specified in the defaults files or at the command line. The tunable parameters default to reasonable values if

they are not listed on the command line. Tunables listed on the command line override those specified elsewhere.

Tunable parameters are as follows:

- Internal defaults—The built-in defaults are used when the value for a particular tunable is not specified elsewhere (on the command line or in a defaults file).
- System-wide defaults file—The system-wide defaults file contains default values that you can alter. These values are used for tunables that are not specified on the command line or in an alternate defaults file.
- Alternate defaults file—A non-standard defaults file, specified with the command `vxassist -d alt_defaults_file`.
- Command line—The tunable values specified on the command line override any values specified internally or in defaults files.

Defaults File The default behavior of the `vxassist` command is controlled by the tunables specified in the `/etc/default/vxassist` file. The format of the defaults file is a list of *attribute=value* pairs separated by new lines. These *attribute=value* pairs are the same as those specified as options on the `vxassist` command line (see the `vxassist(1M)` manual page for details).

The following is a sample `vxassist` defaults file:

```
# by default:
# create unmirrored, unstriped volumes
# allow allocations to span drives
# with RAID-5 create a log, with mirroring don't create a log
# align allocations on cylinder boundaries
layout=nomirror,nostripe,span,nocontig,raid5log,noregionlog,
diskalign
# use the fsgen usage type, except when creating RAID-5 volumes
usetype=fsgen
# allow only root access to a volume
mode=u=rw,g=o=
user=root
group=root
# when mirroring, create two mirrors
nmirror=2
# for regular striping, by default create between 2 and 8 stripe
# columns
max_nstripe=8
```

```
min_nstripe=2
# for RAID-5, by default create between 3 and 8 stripe columns
max_nraid5stripe=8
min_nraid5stripe=3
# create 1 log copy for both mirroring and RAID-5 volumes, by
default
nregionlog=1
nraid5log=1
# by default, limit mirroring log lengths to 32Kbytes
max_regionloglen=32k
# use 64K as the default stripe unit size for regular volumes
stripe_stwid=64k
# use 16K as the default stripe unit size for RAID-5 volumes
raid5_stwid=16k
```

vxctl Daemon

The volume configuration daemon (vxconfigd) is the interface between the Volume Manager commands and the kernel device drivers (vol, vols, DMP). The config device is a special device file created by the Volume Manager that interacts with vxctl to make system configuration changes.

Some vxctl tasks involve modifications to the volboot file that indicates the location of some root configuration copies.

The vxctl daemon is the interface to vxconfigd and is used for:

- performing tasks related to the state of the vxconfigd daemon
- managing boot information and Volume Manager root configuration initialization
- manipulating the contents of the volboot file that contains a list of disks containing root configuration databases (this is not normally necessary, as Volume Manager automatically locates all disks on the system)

reconfiguring the DMP database (if used on your system) to reflect new disk devices attached to the system, and removal of any disk devices from the system

- creating DMP (if used on your system) device nodes in the device directories /dev/vx/dmp and /dev/vx/rdmp
- reflecting the change in path type into the DMP database for

active/passive disk arrays. You can change the path type from primary to secondary and vice-versa through the utilities provided by disk array vendors

For more information, see the `vxctl` (1M) manual page.

vxmake Command

You can use the `vxmake` command to add a new volume, plex, or subdisk to the set of objects managed by Volume Manager. The `vxmake` command adds a new record for that object to the Volume Manager configuration database. You can create records from parameters specified on the command line or by using a description file.

Specify operands on the command line as shown in this example:

```
# vxmake -Uusage_type vol volume_name len=length  
plex=plex_name, ..
```

where:

- the first operand (keyword) determines the kind of object to be created
- the second operand is the name given to that object
- additional operands specify attributes for the object

If no operands are specified on the command line, then a description file is used to specify the records to create.

A **description file** is a file that contains plain text that describes the objects to be created with `vxmake`. A description file can contain several commands, and can be edited to perform a list of tasks. The description file is read from standard input, unless the `-d description_file` option specifies a filename.

The following is a sample description file:

```
#rectyp #name      #options  
sd      disk3-01    disk=disk3 offset=0 len=10000  
sd      disk3-02    disk=disk3 offset=25000 len=10480  
sd      disk4-01    disk=disk4 offset=0 len=8000  
sd      disk4-02    disk=disk4 offset=15000 len=8000  
sd      disk4-03    disk=disk4 offset=30000 len=4480  
plex    db-01      layout=STRIPE ncolumn=2 stwidth=16k
```

```
sd=disk3-01:0/0,disk3-02:0/10000,disk4-01:1/0,\
```



```
disk4-02:1/8000, disk4-03:1/16000
sd      ramd1-01      disk=ramd1 len=640
                        comment="Hot spot for dbvol
plex    db-02          sd=ramd1-01:40320
vol     db              usetype=gen plex=db-01,db-02
                        readpol=prefer prefname=db-02
                        comment="Uses mem1 for hot spot in last 5m
```

This description file specifies a volume with two plexes. The first plex has five subdisks on physical disks. The second plex is preferred and has one subdisk on a volatile memory disk. For more information, see the `vxmake` (1M) manual page.

vxplex Command

The `vxplex` command performs Volume Manager tasks on a plex or on volume-and-plex combinations. The first operand is a keyword that specifies the task to perform. The remaining operands specify the objects to which the task is to be applied.

Use the `vxplex` command to:

- Attach or detach a plex and a volume. A detached plex does not share in I/O activity to the volume, but remains associated with the volume. A detached plex is reattached when a volume is next started.
- Dissociate a plex from the associated volume. When a plex is dissociated, its relationship to the volume is broken. The plex is then available for other uses and can be associated with a different volume. This is useful as part of a backup procedure.
- Copy the contents of the specified volume to the named plex(es). This task makes a copy of a volume (for backup purposes) without mirroring the volume in advance.
- Move the contents of one plex to a new plex. This is useful for moving a plex on one disk to a different location.

For more information, see the `vxplex` (1M) manual page.

vxsd Command

The `vxsd` command maintains subdisk-mirror associations. Use the `vxsd` command to:

- associate or dissociate a subdisk from its associated mirror

- move the contents of a subdisk to another subdisk
- split one subdisk into two subdisks that occupy the same space as the original
- join two contiguous subdisks into one.

NOTE

Some vxsd tasks can take a large amount of time to complete.

For more information, see the vxsd (1M) manual page.

vxmend Command

The vxmend command performs Volume Manager usage-type-specific tasks on volumes, plexes, and subdisks. These tasks fix simple problems in configuration records (such as clearing utility fields, changing volume or plex states, and offlining or onlining volumes or plexes).

The vxmend command is used primarily to escape from a state that was accidentally reached. The offline and online functions can be performed with disk-related commands.

For more information, see the vxmend (1M) manual page.

vxprint Command

The vxprint command displays information from records in a Volume Manager configuration database. You can use this command to display partial or complete information about any or all Volume Manager objects. The format can be hierarchical to clarify relationships between Volume Manager objects. UNIX system utilities such as awk, sed, or grep can also use vxprint output.

For more information, see the vxprint (1M) manual page.

vxrelocd Command

Hot-relocation is turned on as long as the vxrelocd program is running. It is advisable to leave hot-relocation turned on so that you can take advantage of this feature if a failure occurs. However, if you disable this feature because you do not want the free space on some of your disks used for relocation, you must prevent the vxrelocd program from

starting at system startup time.

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 the `vxrelocd` program behaves by either editing the `vxrelocd` line in the startup file that invokes the `vxrelocd` program (`/sbin/rc2.d/S095vxvm-recover`) or killing the existing `vxrelocd` process and restarting it with different options. After making changes to the way the `vxrelocd` program is invoked in the startup file, reboot the system so that the changes take effect. If you 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.

The `vxrelocd` command can be altered as follows:

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

```
# vxrelocd root user_name1 user_name2 &
```

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

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

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

Options

Command options include:

- `-O`

This option is used to revert to an older version. Specifying `VxVM_version -O` tells the `vxrelocd` command to use the relocation scheme in that version.

- `-s`

Before the `vxrelocd` program attempts relocation, a snapshot of the current configuration is saved in `/etc/vx/saveconfig.d`. This option specifies the maximum number of configurations to keep for each disk group. The default is 32.

vxstat Command

The `vxstat` command prints statistics about Volume Manager objects and block devices under Volume Manager control. The `vxstat` command reads the summary statistics for Volume Manager objects and formats them to the standard output. These statistics represent Volume Manager activity from the time the system initially booted or from the last time statistics were cleared. If no Volume Manager object name is specified, statistics from all volumes in the configuration database are reported.

For more information, see the `vxstat(1M)` manual page and Chapter 9, Performance Monitoring,.

vxtrace Command

The `vxtrace` command prints kernel I/O error or I/O error trace event records on the standard output or writes them to a file in binary format. Binary trace records written to a file are also read back and formatted by the `vxtrace` command.

If no operands are given, then either all error trace data or all I/O trace data on all virtual disk devices are reported. With error trace data, you can select all accumulated error trace data, wait for new error trace data, or both (the default). Selection can be limited to a specific disk group, to specific Volume Manager kernel I/O object types, or to particular named objects or devices.

For more information, see the `vxtrace(1M)` manual page and Chapter 9, Performance Monitoring,.

vxunrelocate Command

After the disk that experienced the failure is fixed or replaced, the `vxunreloc` command can be used to move all the hot-relocated subdisks back to the disk. When a subdisk is hot-relocated, its original disk media name and the offset into the disk, are saved in the configuration database. When a subdisk is moved back to the original disk or to a new disk using the `vxunreloc` command, the information is erased. The

original dm name and the original offset are saved in the subdisk records. To print all of the subdisks that were hot-relocated from disk01 in the rootdg disk group, use the following command:

```
# vxprint -g rootdg -se 'sd_orig_dmname="disk01"'
```

To move all the subdisks that were hot-relocated from disk01 back to the original disk, use the following command:

```
# vxunreloc -g rootdg disk01
```

The vxunreloc command provides the -n option to move the subdisks to a different disk from where they were originally relocated. For example, when disk01 fails, all the subdisks that reside on it are hot-relocated to other disks. After the disk is repaired, it is added back to the disk group using a different name, for example, disk05. To move all the hot-relocated subdisks to the new disk, use the following command:

```
# vxunreloc -g rootdg -n disk05 disk01
```

The destination disk should have at least as much storage capacity as was in use on the original disk. If there is not enough space, the unrelocate operation fails and none of the subdisks are moved.

The vxunreloc command moves the hot-relocated subdisks to the original offsets. However, if some subdisks occupy part or all of the area on the destination disk, the vxunreloc command fails. If vxunreloc command failure occurs, select one of the following two options:

- move the existing subdisks elsewhere, and then re-run the vxunreloc command
- use the -f option provided by the vxunreloc command to move the subdisks to the destination disk, but allow the vxunreloc command to locate space on the disk. As long as the destination disk is large enough for the subdisk storage region to accommodate all subdisks, all hot-relocated subdisks are “unrelocated” without using the original offsets.

A subdisk that is hot-relocated more than once due to multiple disk failures can still be unrelocated back to its original location. For example, if disk01 fails, a subdisk named disk01-01 is moved to disk02. If disk02 then fails, all subdisks that reside on it, including the subdisk that was hot-relocated to it, are moved again. When disk02 is replaced, an unrelocate operation for disk02 does not affect hot-relocated subdisk disk01-01. However, a replacement of disk01, followed by the unrelocate operation, moves disk01-01 back to disk01 when the

`vxunreloc` command is run immediately after the replacement.

Restarting the vxunreloc Program After Errors Internally, the `vxunreloc` program moves subdisks in three phases. The first phase creates as many subdisks on the specified destination disk as there are subdisks to be unrelocated. When the subdisks are made, the `vxunreloc` program fills in the `comment` field in the subdisk record with the string `UNRELOC` for identification. In the second phase, the data is moved. If all the subdisk moves are successful, the third phase clears the `comment` field of the subdisk records.

Making the subdisk is an all-or-none operation. If the `vxunreloc` program cannot make all the subdisks successfully, no subdisks are made and the `vxunreloc` program exits. The operation of the subdisks move is not all-or-none. One subdisk move is independent of another, and as a result, if one subdisk move fails, the `vxunreloc` utility prints an error message and then exits. However, all of the subsequent subdisks remain on the disk where they were hot-relocated and are not moved back. Subdisks that are returned home retain the `UNRELOC` string in the `comment` field of the subdisk records because the cleanup phase does not execute.

If the system goes down after the new subdisks are made on the destination, but before they are moved back, the `unrelocate` utility can be executed again after the system is brought back up. As described above, when a new subdisk is created, the `vxunreloc` command sets the `comment` field of the subdisk to `UNRELOC`. Re-execution of the `vxunreloc` command checks the `offset`, the `len`, and the `comment` fields of the existing subdisks on the destination disk to determine if the subdisks were left on the disk during a previous execution of the `vxunreloc` command.

NOTE

Do not manually modify the string `UNRELOC` in the `comment` field.

If one out of a series of subdisk moves fails, the `vxunreloc` program exits. Check the error that caused the subdisk move to fail and determine if the unrelocation can proceed. When you re-execute the `vxunreloc` command to resume the subdisk moves, it uses the subdisks created at a previous run.

The cleanup phase is performed with one transaction. The `vxunreloc` command resets the comment field to a NULL string for all subdisks marked as UNRELOC that reside on the destination disk. This includes clean-up for those subdisks that were unrelocated in any previous invocation of the `vxunreloc` command.

vxvol Command

The `vxvol` command performs Volume Manager tasks on volumes. Use the `vxvol` command to:

- initialize a volume
- start a volume
- stop a volume
- establish the read policy for a volume

Starting a volume changes its kernel state from DISABLED or DETACHED to ENABLED. Stopping a volume changes its state from ENABLED or DETACHED to DISABLED. For more information about volume states, see Chapter 8, Recovery,.

Use the `vxvol` command to specify one of these read policies:

- `round`

This option reads each plex in turn in “round-robin” fashion for each nonsequential I/O detected. Sequential access causes only one plex to be accessed. This takes advantage of the drive or controller read-ahead caching policies.

- `prefer`

This option reads first from a plex that has been named as the preferred plex.

- `select`

This option chooses a default policy based on plex associations to the volume. If the volume has an enabled striped plex, the `select` option defaults to preferring that plex; otherwise, it defaults to round-robin.

For more information, see the `vxvol (1M)` manual page.

4 **Disk Tasks**

Introduction

This chapter describes the operations for managing disks used by the Volume Manager.

NOTE

Most Volume Manager commands require superuser or other appropriate privileges.

The following topics are covered in this chapter:

- “Disk Devices”
- “Disk Utilities”
- “vxdiskadm Main Menu”
- “Initializing Disks”
- “Adding a Disk to Volume Manager”
- “Placing Disks Under Volume Manager Control”
- “Moving Disks”
- “Enabling a Physical Disk”
- “Disabling a Disk”
- “Replacing a Disk”
- “Removing Disks”
- “Taking a Disk Offline”
- “Adding a Disk to a Disk Group”
- “Adding a VM Disk to the Hot-Relocation Pool”
- “Removing a VM Disk From the Hot-Relocation Pool”
- “Excluding a Disk from Hot-Relocation Use”
- “Including a Disk for Hot-Relocation Use”
- “Reinitializing a Disk”
- “Renaming a Disk”

- “Reserving Disks”
- “Displaying Disk Information”

Disk Devices

Two classes of disk devices can be used with the Volume Manager: standard devices and special devices. In Volume Manager, special devices are considered physical disks connected to the system that are represented 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. Use the `vxdisk` utility to display the paths of a metadevice, and to display the status of each path (for example, enabled or disabled).

NOTE

Using special devices applies only to systems with the Dynamic Multipathing (DMP) feature.

When performing disk administration, it is important that you recognize the difference between a device name and a disk name.

The **device name** (sometimes referred to as **devname** or **disk access name**) is the location of the disk. The syntax of a typical device name is `c#t#d#`, where:

- **c#**—The number of the controller to which the disk drive is attached (used on HP-UX systems)
- **t#** and **d#**—The target ID and device number that constitute the address of the disk drive on the controller (used on HP-UX systems)
- **s2**—The HP-UX partition number on the boot disk drive (used on HP-UX 11i Version 1.5 systems)

On HP-UX 11i Version 1.5 systems, the HP-UX partition number on the boot disk drive is `c#t#d#s2`, where **s2** is the slice or partition number.

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 area (with the public area following the private area).
- *nopriv*—there is no private region (only a public region for allocating subdisks).

NOTE

For HP-UX 11i, disks (except the root disk) are treated and accessed as entire physical disks, so a device name of the form `c#t#d#` is used. On HP-UX 11i Version 1.5, the root disk is on partition 2, `c#t#d#s2`.

The full pathname of a device is `/dev/vx/dmp/devicename`. In this document, only the device name is listed and `/dev/vx/dmp` is assumed.

The **disk name** (sometimes referred to as **disk media name**) is an administrative name for the disk, such as `disk01`. If you do not assign a disk name, the disk name defaults to `disk##` if the disk is being added to `rootdg` (where `##` is a sequence number). Otherwise, the default disk name is `groupname##`, where *groupname* is the name of the disk group to which the disk is added. Your system may use a device name that differs from the examples.

Disk Utilities

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

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

Utilities discussed in this chapter include:

- `vxdiskadm`—the `vxdiskadm` utility is the Volume Manager Support Operations menu interface. The `vxdiskadm` 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.
- `vxdiskadd`—the `vxdiskadd` utility is used to add standard disks to the Volume Manager. The `vxdiskadd` utility leads you through the process of initializing a new disk by displaying information and prompts. See the `vxdiskadd(1M)` manual page for more information.
- `vxdisk`—the `vxdisk` utility is the command-line utility for administering disk devices. The `vxdisk` utility 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 more information.

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. 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 `disk01`, enter:

```
# vxdisk list disk01
```

- `vxunreloc`—VxVM Hot-relocation allows the system to automatically react to I/O failures on a redundant VxVM object at the

subdisk level and then take necessary action to make the object available again. This mechanism detects I/O failures in a subdisk, relocates the subdisk, and recovers the plex associated with the subdisk. After the disk has been replaced, Volume Manager provides a utility, `vxunreloc`, that allows you to restore the system back to the configuration that existed before the disk failure. The `vxunreloc` utility allows you to move the hot-relocated subdisks back onto a disk that was replaced due to a failure.

When the `vxunreloc` utility is invoked, you must specify the disk media name where the hot-relocated subdisks originally resided. When the `vxunreloc` utility moves the subdisks, it moves them to the original offsets. If you try to unrellocate to a disk that is smaller than the original disk that failed, the `vxunreloc` utility does nothing except return an error.

The `vxunreloc` utility provides an option to move the subdisks to a different disk from where they were originally relocated. It also provides an option to unrellocate subdisks to a different offset as long as the destination disk is large enough to accommodate all the subdisks. For more information on using the `vxunreloc` utility, see “`vxunrelocate` Command”.

- `vxdiskadd`—the `vxdiskadd` utility and most `vxdiskadm` operations can be used only with standard disk devices.

vxdiskadm Main Menu

The vxdiskadm main menu is as follows:

Volume Manager Support Operations

Menu: VolumeManager/Disk

- 1 Add or initialize one or more disks
- 2 Remove a disk
- 3 Remove a disk for replacement
- 4 Replace a failed or removed disk
- 5 Mirror volumes on a disk
- 6 Move volumes from a disk
- 7 Enable access to (import) a disk group
- 8 Remove access to (deport) a disk group
- 9 Enable (online) a disk device
- 10 Disable (offline) a disk device
- 11 Mark a disk as a spare for a disk group
- 12 Turn off the spare flag on a disk
- 13 Remove (deport) and destroy a disk group
- 14 Unrelocate subdisks back to a disk
- 15 Exclude a disk from hot-relocation use
- 16 Make a disk available for hot-relocation use

list List disk information

- ? Display help about menu
- ?? Display help about the menuing system
- q Exit from menus

Select an operation to perform

- **?** can be entered at any time to provide help in using the menu. The output of **?** is a list of operations and a definition of each.
- **??** lists inputs that can be used at any prompt.
- **q** returns you to the main menu if you need to restart a process; however, using **q** at the main menu level exits the Volume Manager Support Operations.

vxdiskadm Menu Description

The vxdiskadm menu provides access to the following tasks. The numbers correspond to the items listed in the main menu:

1. Add or initialize one or more disks.

You can add formatted disks to the system. SCSI disks are already formatted. For other disks, see the manufacturer's documentation for formatting instructions. You are prompted for the disk device(s). You can specify the disk group to which the disk(s) should be added; if none is selected, the disk is held as a spare to be used for future operations or disk replacements without needing to be initialized at that time. You can also specify that selected disks be marked as hot-relocation spares for a disk group. If the disk has not been initialized already, the disk is initialized for use with the Volume Manager.

2. Remove a disk.

You can remove a disk from a disk group. You are prompted for the name of a disk to remove. You cannot remove a disk if any volumes use storage on that disk. If any volumes are using storage on the disk, you have the option of asking the Volume Manager to move that storage to other disks in the disk group.

NOTE

You cannot remove the last disk in a disk group using this task. To use all remaining disks in a disk group, disable (deport) the disk group. You can then reuse the disks. However, the rootdg cannot be deported.

3. Remove a disk for replacement.

You can remove a physical disk from a disk group, while retaining the disk name. This changes the state for the named disk to removed. If there are any initialized disks that are not part of a disk group, you are given the option of using one of these disks as a replacement.

4. Replace a failed or removed disk.

You can specify a replacement disk for a disk that you removed with the Remove a disk for replacement menu entry, or one that failed during use. You are prompted for a disk name to replace and a disk

device to use as a replacement. You can choose an uninitialized disk, in which case the disk will be initialized, or you can choose a disk that you have already initialized using the `Add` or `initialize a disk` menu operation.

5. Mirror volumes on a disk.

You can mirror volumes on a disk. These volumes can be mirrored to another disk with available space. Creating mirror copies of volumes in this way protects against data loss in case of disk failure. Volumes that are already mirrored or that are comprised of more than one subdisk will not be mirrored with this task. Mirroring volumes from the boot disk will produce a disk that can be used as an alternate boot disk.

6. Move volumes from a disk.

You can move any volumes (or parts of a volume) that are using a disk onto other disks. Use this menu task immediately prior to removing a disk, either permanently or for replacement.

NOTE

Simply moving volumes off a disk, without also removing the disk, does not prevent other volumes from being moved onto the disk by future operations.

7. Enable access to (import) a disk group.

You can enable access by this system to a disk group. If you wish to move a disk group from one system to another, you must first disable (deport) it on the original system. Then, move the disks from the deported disk group to the other system and enable (import) the disk group there. You are prompted for the disk group name.

8. Disable access to (deport) a disk group

You can disable access to a disk group that is currently enabled (imported) by this system. Deport a disk group if you intend to move the disks in a disk group to another system. Also, deport a disk group if you want to use all of the disks remaining in a disk group for some new purpose.

You are prompted for the name of a disk group. You are asked if the disks should be disabled (offlined). For removable disk devices on some systems, it is important to disable all access to the disk before

removing the disk.

9. Enable (online) a disk device.

If you move a disk from one system to another during normal system operation, the Volume Manager will not recognize the disk automatically. Use this menu task to tell the Volume Manager to scan the disk to identify it, and to determine if this disk is part of a disk group. Also, use this task to re-enable access to a disk that was disabled by either the disk group deport task or the disk device disable (offline) operation.

10. Disable (offline) a disk device.

You can disable all access to a disk device through the Volume Manager. This task can be applied only to disks that are not currently in a disk group. Use this task if you intend to remove a disk from a system without rebooting.

Note that some systems do not support disks that can be removed from a system during normal operation. On those systems, the offline operation is seldom useful.

11. Mark a disk as a spare for a disk group.

You can reserve a disk as an automatic replacement disk (for hot-relocation) in case another disk in the disk group should fail.

12. Turn off the spare flag on a disk.

You can free hot-relocation spare disks for use as regular VM disks.

You can display a list of disks attached to your system. This also lists removed or failed disks. You can also use this task to list detailed information for a particular disk. This information includes the disk group of which the disk is a member, even if that disk group is currently disabled.

13. Remove (deport) and destroy a disk group.

You can remove access to and destroy a disk group that is currently enabled (imported) by this system. Destroy a disk group if you intend to use the disks for some new purpose.

You are prompted for the name of a disk group. You are also asked if the disks should be disabled (offlined). For removable disk devices on some systems, it is important to disable all access to the disk before removing the disk.

14. Unrelocate subdisks back to a disk.

VxVM hot-relocation allows the system to automatically react to IO failures on a redundant VxVM object at the subdisk level and take necessary action to make the object available again. This mechanism detects I/O failures in a subdisk, relocates the subdisk, and recovers the plex associated with the subdisk. After the disk has been replaced, Volume Manager provides the vxunreloc utility, which can be used to restore the system to the same configuration that existed before the disk failure. vxunreloc allows you to move the hot-relocated subdisks back onto a disk that was replaced due to a disk failure.

15. Exclude a disk from hot-relocation use.

Excludes disks in the free pool (non-spares) to be used by hot-relocation.

16. Make a disk available for hot-relocation use.

Makes disk in the free pool (non_spares) available for hot-relocation.

Initializing Disks

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

- Step 1.** Formatting of the disk media itself. This must be done outside of the Volume Manager.
- Step 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 command (on some systems, `diskadd`) to do a media format of any disk.

NOTE

SCSI disks are usually preformatted.

For more information, see the `formatting(1M)` manual page.

Volume Manager Disk Installation

Use either the `vxdiskadm` menus or the `vxdiskadd` command for disk initialization. This section describes how to use the `vxdiskadd` command.

Use the `vxdiskadd` command to initialize a specific disk. For example, to initialize the second disk on the first controller, use the following command:

```
# vxdiskadd c0t1d2
```

The `vxdiskadd` command examines your disk to determine whether it has been initialized and displays prompts based on what it finds. The `vxdiskadd` command checks for 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 the `vxdiskadd` command. Ignore these messages. These messages should not appear after the disk has been fully initialized; the `vxdiskadd` command 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**

When the vxdiskadm program prompts whether to exclude this disk from hot-relocation use, enter **n** (or press Return).

Exclude disk from hot-relocation use? [y,n,q,?] (default: 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:

c0t1d0

Initializing device c0t1d0.

Adding disk device c0t1d0 to disk group rootdg with disk name

disk01.

Goodbye.

Adding a Disk to Volume Manager

You must place a disk under Volume Manager control, or add it to a disk group, before you can use the disk space for volumes. If the disk was previously in use, but not under Volume Manager control, you can preserve existing data on the disk while still letting the Volume Manager take control of the disk. If the user wants to bring all the non LVM disks under control, they are considered as fresh disks. LVM volume groups can only be converted to VxVM disk groups. If the disk was previously not under Volume Manager control, but no data is required to be preserved, it should be initialized.

NOTE

See the *VERITAS Volume Manager Migration Guide* for more information on conversion.

To add a disk, use the following command:

```
# vxdiskadd devname
```

where *devname* is the device name of the disk to be added.

To add the device `c1t0d0` to Volume Manager control, perform the following steps:

Step 1. Enter the following to start the `vxdiskadd` program:

```
# vxdiskadd c1t0d0
```

Step 2. To continue with the task, enter **y** (or press Return) at the following prompt:

```
Add or initialize disks
Menu: VolumeManager/Disk/AddDisks
Here is the disk selected.  Output format: [Device_Name]
c1t0d0
Continue operation? [y,n,q,?] (default: y) y
```

Step 3. At the following prompt, specify the disk group to which the disk should be added or press Return to accept `rootdg`:

```
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)

- Step 4.** At the following prompt, either press Return to accept the default disk name or enter a disk name:

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

- Step 5.** When prompted as to whether this disk should become a hot-relocation spare, enter **n** (or press Return):

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

- Step 6.** When the vxdiskadm program prompts whether to exclude this disk from hot-relocation use, enter **n** (or press Return).

Exclude disk from hot-relocation use? [y,n,q,?]
(default: n)

- Step 7.** To continue with the task, enter **y** (or press Return) at the following prompt:

The selected disks will be added to the disk group rootdg with default disk names.
c1t0d0
Continue with operation? [y,n,q,?] (default: y) **y**

- Step 8.** Messages similar to the following are displayed while the disk device is initialized and added to the disk group selected by you:

c1t0d0
Initializing device c1t0d0.
Adding disk device c0t1d0 to disk group rootdg with disk name
disk01.
Goodbye.

Placing Disks Under Volume Manager Control

When you add a disk to a system that is running Volume Manager, you need to put the disk under Volume Manager control so that the Volume Manager can control the space allocation on the disk. Unless another disk group is specified, Volume Manager places new disks in the default disk group, `rootdg`. When you are asked to name a disk group, enter **none** instead of selecting `rootdg` or typing in a disk group name. The disk is then initialized as before, but is reserved for use at a later time. It cannot be used until it is added to a disk group. Note that this type of “spare disk” should not be confused with a hot-relocation spare disk.

The method by which you place a disk under Volume Manager control depends on the circumstances:

- If the disk is new, it must be initialized and placed under Volume Manager control.
- If the disk is not needed immediately, it can be initialized (but not added to a disk group) and reserved for future use.
- If the disk was previously initialized for future Volume Manager use, it can be reinitialized and placed under Volume Manager control.
- If the disk was previously used for a file system, Volume Manager gives you an option before destroying the file system.
- If the disk was previously in use, but not under Volume Manager control, you can preserve existing data on the disk while still letting Volume Manager take control of the disk. This is accomplished using conversion. With conversion, the virtual layout of the data is fully converted to Volume Manager's control (see the *VERITAS Volume Manager Migration Guide*).
- Multiple disks on one or more controllers can be placed under Volume Manager control simultaneously. Depending on the circumstances, all of the disks may not be processed the same way.

When initializing multiple disks at once, it is possible to exclude certain disks or certain controllers. To exclude disks, list the names of the disks to be excluded in the file `/etc/vx/disks.exclude` before the initialization. Similarly, you can exclude all disks on specific controllers from initialization by listing those controllers in the file `/etc/vx/cntrls.exclude`.

The sections that follow provide detailed examples of how to use the `vxdiskadm` utility to place disks under Volume Manager control in various ways and circumstances.

NOTE

A disk must be formatted (using the `mediainit` command, for example) or added to the system (using the `diskadd` command) before it can be placed under Volume Manager control.

If you attempt to place an unformatted disk under Volume Manager control through the `vxdiskadm` utility, the initialization begins as normal, but quits with a message that the disk does not appear to be valid and may not be formatted. If you receive this message, format the disk properly and then attempt to place the disk under Volume Manager control again.

Placing a Formatted Disk Under VM Control

A formatted disk can be new or previously used outside Volume Manager. Initialization does not preserve data on disks. Initialize a single disk for Volume Manager use as follows:

Step 1. Select menu item 1 (Add or initialize one or more disks) from the `vxdiskadm` main menu.

Step 2. At the following prompt, enter the disk device name of the disk to be added to Volume Manager control (or enter **list** for a list of disks):

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

Use this operation to add one or more disks to a disk group.
You can add the selected disks to an existing disk group or to a new disk group that will be created as a part of the operation. The selected disks may also be added to a disk group as spares. The selected disks may also be initialized without adding them to a disk group leaving the disks available for use as replacement disks.

More than one disk or pattern may be entered at the prompt.

Here are some disk selection examples:

all:all disks

c3 c4t2:all disks on both controller 3 and controller 4, target 2

c3t4d0: a single disk

Select disk devices to add:

[<pattern-list>,all,list,q,?] **list**

DEVICE	DISK	GROUP	STATUS
c2t4d0	-	-	LVM
c2t5d0	-	-	LVM
c2t6d0	-	-	LVM
c3t0d0	disk01	rootdg	online
c3t1d0	disk03	rootdg	online
c3t2d0	disk04	rootdg	online
c3t3d0	disk05	rootdg	online
c3t8d0	disk06	rootdg	online
c3t9d0	disk07	rootdg	online
c3t10d0	disk02	rootdg	online
c4t1d0	disk08	rootdg	online
c4t2d0	TCd1-18238	TCg1-18238	online
c4t13d0	-	-	online
c4t14d0	-	-	online

<pattern-list> can be a single disk, or a series of disks and/or controllers (with optional targets). If <pattern-list> consists of multiple items, those items must be separated by white space.

If you enter **list** at the prompt, the vxdiskadm program displays a list of the disks available to the system, followed by a prompt at which you should enter the device name of the disk to be added:

Select disk devices to add:

[<pattern-list>,all,list,q,?] **c1t0d1**

All disks attached to the system are recognized by the Volume Manager and displayed here.

The phrase online invalid in the STATUS line tells you that a disk has not

yet been added to Volume Manager control. These disks may or may not have been initialized before. The disks that are listed with a disk name and disk group cannot be used for this task, as they are already under Volume Manager control.

- Step 3.** To continue with the operation, enter **y** (or press Return) at the following prompt:

```
Here is the disk selected.  Output format: [Device_Name]
c1t2d0
Continue operation? [y,n,q,?] (default: y) y
```

- Step 4.** At the following prompt, specify the disk group to which the disk should be added or press Return to accept rootdg:

```
You can choose to add these disks to an existing disk
group, a new
disk group, or you can leave these disks 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
disks available for future use, specify a disk group
name of
"none".
Which disk group [<group>,none,list,q,?] (default:
rootdg)
```

- Step 5.** At the following prompt, either press Return to accept the default disk name or enter a disk name:

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

- Step 6.** When the vxdiskadm program asks whether this disk should become a hot-relocation spare, enter **n** (or press Return):

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

- Step 7.** When the vxdiskadm program prompts whether to exclude this disk from hot-relocation use, enter **n** (or press Return).

```
Exclude disk from hot-relocation use? [y,n,q,?]
(default: n) n
```

Step 8. To continue with the operation, enter **y** (or press Return) at the following prompt:

```
The selected disks will be added to the disk group rootdg
with
default disk names.
c1t2d0
Continue with operation? [y,n,q,?] (default: y) y
```

Step 9. If the disk was used for the file system earlier, then the `vxdiskadm` program gives you the following choices:

```
The following disk device appears to contain a currently
unmounted
file system.
c1t2d0
Are you sure you want to destroy this file system
[y,n,q,?] (default: n)
```

Messages similar to the following should now confirm that disk `c1t2d0` is being placed under Volume Manager control.

```
Initializing device c1t2d0.
```

```
Adding disk device c1t2d0 to disk group rootdg with disk
name disk39.
```

Step 10. At the following prompt, indicate whether you want to continue to initialize more disks (**y**) or return to the `vxdiskadm` main menu (**n**):

```
Add or initialize other disks? [y,n,q,?] (default: n)
```

Placing Multiple Disks Under Volume Manager Control

This section describes how to place multiple disks under Volume Manager control simultaneously. The set of disks can consist of all disks on the system, all disks on a controller, selected disks, or a combination thereof.

NOTE

Initialization does not preserve data on disks.

When initializing multiple disks at one time, it is possible to exclude certain disks or certain controllers. To exclude disks, list the names of the disks to be excluded in the file `/etc/vx/disks.exclude` before the initialization. You can exclude all disks on specific controllers from initialization by listing those controllers in the file `/etc/vx/cntrls.exclude`.

Place multiple disks under Volume Manager control at one time as follows:

- Step 1.** Select menu item 1 (Add or initialize one or more disks) from the `vxdiskadm` main menu.
- Step 2.** At the following prompt, enter the *pattern-list* for the disks to be added to Volume Manager control. In this case, enter **c3** to indicate all disks on controller 3:

```
Add or initialize disks
Menu: VolumeManager/Disk/AddDisks
Use this operation to add one or more disks to a disk
group.
You can add the selected disks to an existing disk group
or to
a new disk group that will be created as a part of the
operation. The selected disks may also be added to a disk
group
as spares. The selected disks may also be initialized
without
adding them to a disk group leaving the disks available
for use
as replacement disks.
More than one disk or pattern may be entered at the
prompt. Here
are some disk selection examples:
all:all disks
c3 c4t2:all disks on both controller 3 and controller
  4,target 2
c3t4d0:a single disk
Select disk devices to add:
[<pattern-list>,all,list,q,?] c3
```

where *<pattern-list>* can be a single disk, or a series of disks and/or controllers (with optional targets). If *<pattern-list>* consists of multiple items, those items must be separated by white space.

If you do not know the address (device name) of the disk to be added, enter `list` at the prompt for a complete listing of available disks.

- Step 3.** To continue the operation, enter **y** (or press Return) at the following prompt:

```
Here are the disks selected.  Output format:
[Device_Name]
c3t0d0 c3t1d0 c3t2d0 c3t3d0
Continue operation? [y,n,q,?] (default: y) y
```

- Step 4.** To add these disks to the default disk group, `rootdg`, enter **y** (or press Return) at the following prompt:

```
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) y
```

- Step 5.** To allow the `vxdiskadm` program to use default disk names for each of the disks, enter **y** (or Press Return) at the following prompt:

```
Use default disk names for these disks? [y,n,q,?]
(default: y) y
```

- Step 6.** At the following prompt, enter **n** to indicate that these disks should not be used as hot-relocation spares:

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

- Step 7.** When the `vxdiskadm` program prompts whether to exclude this disk from hot-relocation use, enter **n** (or press Return).

```
Exclude disk from hot-relocation use? [y,n,q,?]
(default: n) n
```

- Step 8.** To continue with the operation, enter **y** (or press Return) at the following prompt:

```
The selected disks will be added to the disk group rootdg
with
default disk names.
c3t0d0 c3t1d0 c3t2d0 c3t3d0
Continue with operation? [y,n,q,?] (default: y) y
```

- Step 9.** If the disk was used for the file system earlier, then the vxdiskadm program gives you the following choices:

```
The following disk device appears to contain a currently
unmounted file system.
c3t1d0 c3t2d0 c3t3d0
Are you sure you want to destroy this file system
[y,n,q,?] (default: n)
```

The vxdiskadm program now confirms those disks that are being initialized and added to Volume Manager control with messages similar to the following:

```
Initializing device c3t1d0.
```

```
Initializing device c3t2d0.
```

```
Initializing device c3t3d0.
```

```
Adding disk device c3t1d0 to disk group rootdg with disk
name disk33.
```

```
Adding disk device c3t2d0 to disk group rootdg with disk
name disk34.
```

```
Adding disk device c3t3d0 to disk group rootdg with disk
name disk35.
```

In addition to the output displayed above, you may see prompts that give you the option of performing surface analysis.

- Step 10.** At the following prompt, indicate whether you want to continue to initialize more disks (**y**) or return to the vxdiskadm main menu (**n**):

```
Add or initialize other disks? [y,n,q,?] (default: n)
```

NOTE

To bring the LVM disks (HP-UX platform) under Volume Manager control, use the Migration Utilities. See the *VERITAS Volume Manager*

Migration Guide—HP-UX for details.

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 mkt dg, use the following commands:

```
# vxdg rmdisk disk04
# vxdg -g mkt dg adddisk mkt dg02=c0
t3d0
```

NOTE

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

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

Enabling a Physical Disk

If you move a disk from one system to another during normal system operation, the Volume Manager does not recognize the disk automatically. The enable disk task enables Volume Manager to identify the disk and to determine if this disk is part of a disk group. Also, this task re-enables access to a disk that was disabled by either the disk group deport task or the disk device disable (offline) task.

To enable a disk, use the following procedure:

- Step 1.** Select menu item 9 (Enable (online) a disk device) from the vxdiskadm main menu.
- Step 2.** At the following prompt, enter the device name of the disk to be enabled (or enter list for a list of devices):

```
Enable (online) a disk device
Menu: VolumeManager/Disk/OnlineDisk
Use this operation to enable access to a disk that was
disabled
with the "Disable (offline) a disk device" operation.
You can also use this operation to re-scan a disk that
may have
been changed outside of the Volume Manager. For example,
if a
disk is shared between two systems, the Volume Manager
running
on the other system may have changed the disk. If so, you
can
use this operation to re-scan the disk.
NOTE:Many vxdiskadm operations re-scan disks without
user
intervention. This will eliminate the need to online a
disk directly, except when the disk is directly
offlined.
Select a disk device to enable [<address>,list,q,?] c1
t1d0
```

vxdiskadm enables the specified device.

- Step 3.** At the following prompt, indicate whether you want to enable another

Enabling a Physical Disk

device (**y**) or return to the vxdiskadm main menu (**n**):

Enable another device? [y,n,q,?] (default: n)

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, mail information is sent as shown in the following display:

```
To: root
Subject: Volume Manager failures on host teal
Failures have been detected by the VERITAS Volume
Manager:
failed plexes:
home-02
src-02
```

To determine which disk is causing the failures in the above example message, enter the following command:

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

A typical output display is as follows:

TYP	NAME	FAILED	
		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 the following 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 a Disk”).

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, mail information is sent as shown in the following 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 mktimg-01 are also detached because of the disk failure.

Again, the problem can be a cabling error. If the problem is not a cabling error, replace the disk.

Disabling a Disk

You can take a disk offline. If the disk is corrupted, you need to take it offline and remove it. You may be moving the physical disk device to another location to be connected to another system. To take a disk offline, first remove it from its disk group, and then use the following procedure:

- Step 1.** Select menu item 10 (Disable (offline) a disk device) from the vxdiskadm main menu.
- Step 2.** At the following prompt, enter the address of the disk you want to disable:

```
Disable (offline) a disk device
Menu: VolumeManager/Disk/OfflineDisk
Use this menu operation to disable all access to a disk
device
by the Volume Manager. This operation can be applied
only to
disks that are not currently in a disk group. Use this
operation
if you intend to remove a disk from a system without
rebooting.
NOTE:Many systems do not support disks that can be
removed from
a system during normal operation. On such systems, the
offline operation is seldom useful.
Select a disk device to disable [<address>,list,q,?]
c1t1d0
```

The vxdiskadm program disables the specified disk.

- Step 3.** At the following prompt, indicate whether you want to disable another device (**y**) or return to the vxdiskadm main menu (**n**):

```
Disable another device? [y,n,q,?] (default: n)
```

Replacing a Disk

If a disk fails, you need to replace that disk with another. This task requires disabling and removing the failed disk and installing a new disk in its place.

If a disk was replaced due to a disk failure and you wish to move hot-relocate subdisks back to this replaced disk, see Chapter 6, Volume Tasks, for information on moving hot-relocate subdisks.

To replace a disk, use the following procedure:

- Step 1.** Select menu item 3 (Remove a disk for replacement) from the vxdiskadm main menu.
- Step 2.** At the following prompt, enter the name of the disk to be replaced (or enter **list** for a list of disks):

```
Remove a disk for replacement
Menu: VolumeManager/Disk/RemoveForReplace
Use this menu operation to remove a physical disk from a
disk
group, while retaining the disk name. This changes the
state
for the disk name to a removed disk. If there are any
initialized disks that are not part of a disk group, you
will be
given the option of using one of these disks as a
replacement.
Enter disk name [<disk>, list, q, ?] disk02
```

Additional displays show any volumes associated with the disk you wish to remove. You must decide whether to keep the data associated with the volumes or to allow that data to be lost when the disk is replaced. Answer any prompts accordingly.

- Step 3.** At the following prompt, either select the device name of the replacement disk (from the list provided) or press Return to choose the default disk:

```
The following devices are available as replacements:
c1t1d0
You can choose one of these disks now, to replace disk02.
Select "none" if you do not wish to select a replacement
```

```
disk.  
Choose a device, or select "none"  
[<device>,none,q,?] (default: c1  
t1d0)
```

Step 4. At the following prompt, press Return to continue:

```
The requested operation is to use the initialized device  
c1t0d0 to replace the removed or failed disk disk02 in  
disk  
group rootdg.  
Continue with operation? [y,n,q,?] (default: y)
```

The vxdiskadm program displays the following success messages:

```
Replacement of disk disk02 in group rootdg with disk  
device  
c1t0d0 completed successfully.
```

Step 5. At the following prompt, indicate whether you want to remove another disk (**y**) or return to the vxdiskadm main menu (**n**):

```
Remove another disk? [y,n,q,?] (default: n)
```

Replacing a Failed or Removed Disk

Disks can be removed and then replaced later. To remove a disk, use the following procedure:

Step 1. Select menu item 4 (Replace a failed or removed disk) from the vxdiskadm main menu.

Step 2. Select the disk name of the disk to be replaced:

```
Replace a failed or removed disk  
Menu: VolumeManager/Disk/ReplaceDisk  
Use this menu operation to specify a replacement disk  
for a disk  
that you removed with the "Remove a disk for  
replacement" menu  
operation, or that failed during use. You will be  
prompted for  
a disk name to replace and a disk device to use as a  
replacement.
```

You can choose an uninitialized disk, in which case the disk will be initialized, or you can choose a disk that you have already initialized using the Add or initialize a disk menu operation.

Select a removed or failed disk [<disk>,list,q,?] **disk02**

- Step 3.** The vxdiskadm program displays the device names of the disk devices available for use as replacement disks. Your system may use a device name that differs from the examples.

Enter the device name of the device of your choice or press Return to select the default device:

The following devices are available as replacements:
c1t0d0 c1t1d0

You can choose one of these disks to replace disk02. Choose "none" to initialize another disk to replace disk02.

Choose a device, or select "none"
[<device>,none,q,?] (default: c1t0d0)pa

- Step 4.** At the following prompt, press Return to replace the disk:

The requested operation is to use the initialized device c1t0d0 to replace the removed or failed disk disk02 in disk group rootdg.

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

The vxdiskadm program displays the following success message:

Replacement of disk disk02 in group rootdg with disk device c1t0d0 completed successfully.

- Step 5.** At the following prompt, indicate whether you want to replace another disk (**y**) or return to the vxdiskadm main menu (**n**):

Replace another disk? [y,n,q,?] (default: n)

Removing Disks

You can remove a disk from a system and move it to another system if the disk is failing or has failed. Before removing the disk from the current system, you must:

- Step 1.** Unmount any file systems on the volumes.
- Step 2.** Stop the volumes on the disk.
- Step 3.** Move the volumes to other disks or back up the volumes. To move a volume, mirror the volume on one or more other disks, then remove the original copy of the volume. If the volumes are no longer needed, they can be removed instead of moved. Before removing a disk, make sure it contains no data, all data no longer needed, or the data can be moved to other disks. Then remove the disk using the `vxdiskadm` utility:
- Step 4.** Select menu item
2 (Remove a disk) from the `vxdiskadm` main menu.

NOTE

You must disable the disk group before you can remove the last disk in that group.

- Step 5.** At the following prompt, enter the disk name of the disk to be removed:

```
Remove a disk
Menu: VolumeManager/Disk/RemoveDisk
Use this operation to remove a disk from a disk group.
This
operation takes a disk name as input. This is the same
name
that you gave to the disk when you added the disk to the
disk
group.
Enter disk name [<disk>, list, q, ?] disk01
```

- Step 6.** If there are any volumes on the disk, the Volume Manager asks you whether they should be evacuated from the disk. If you wish to keep the volumes, answer **y**. Otherwise, answer **n**.

Step 7. At the following verification prompt, press Return to continue:

```
Requested operation is to remove disk disk01 from group  
rootdg.
```

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

The `vxdiskadm` utility removes the disk from the disk group and displays the following success message:

```
Removal of disk disk01 is complete.
```

You can now remove the disk or leave it on your system as a replacement.

Step 8. At the following prompt, indicate whether you want to remove other disks (**y**) or return to the `vxdiskadm` main menu (**n**):

```
Remove another disk? [y,n,q,?] (default: n)
```

Removing a Disk with No Subdisks

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

```
# vxdbg  
[-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 the following command:

```
# vxdbg rmdisk disk02
```

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

```
vxdbg:Disk diskname is used by one or more subdisks
```

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

NOTE

Use of the `-k` option to the `vxdbg` command 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
```

To remove `c1t0d0` from Volume Manager control, use one of the following commands. On systems without a bus, use the command:

```
# vxdisk rm c1t0d0
```

On systems with a bus, use the command:

```
# vxdisk rm c1b0t0d0
```

Removing a Disk with Subdisks

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 the `vxdiskadm` program to remove a disk, you can choose to move volumes off that disk. To do this, run the `vxdiskadm` program and select item 3 (Remove a disk) from the main menu.

If the disk is used by some subdisks, the following 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 are not movable. A subdisk may not be movable for one of the following reasons:

- 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 the `vxdiskadm` program cannot move some subdisks, remove some plexes from some disks to free more space before proceeding with the disk removal operation. See Chapter 6, Volume Tasks, for information on how to remove volumes and plexes.

Removing a Disk as a Hot-Relocation Spare

While a disk is designated as a spare, the space on that disk is not used

as free space for the creation of Volume Manager objects within its disk group. If necessary, you can free a spare disk for general use by removing it from the pool of hot-relocation disks.

To determine which disks are currently designated as spares, use the following command:

```
# vxdisk list
```

The output of this command lists any spare disks with the `spare` flag.

To remove a spare from the hot-relocation pool, use the following command:

```
# vxedit set spare=off diskname
```

For example, to make `disk01` available for normal use, use the following command:

```
# vxedit set spare=off disk01
```

Taking a Disk Offline

There are instances when you must take a disk offline. If a disk is corrupted, you must disable it and remove it. You must also disable a disk before moving the physical disk device to another location to be connected to another system.

To take a physical disk offline, first remove the disk from its disk group (see “Removing Disks”), and then place the disk in an “offline” state with the following command:

```
# vxdisk offline device_name
```

For example, to take the device c1t1d0 off line, use the following command:

```
# vxdisk offline c1t1d0
```

NOTE

The device name is used here because the disk is no longer in a disk group, so it does not have an administrative name.

Adding a Disk to a Disk Group

You can add a new disk to an already established disk group. For example, the current disks have insufficient space for the application or work group requirements, especially if these requirements have changed.

To add an initialized disk to a disk group, use the following command:

```
# vxdiskadd devname
```

For example, to add device `c1t1d0` to `rootdg`, use the following procedure:

Step 1. Enter this command to start the `vxdiskadd` program:

```
# vxdiskadd c1t1d0
```

The `vxdiskadd` program displays the following message. Enter **y** to continue the process.

```
Add or initialize disks
Menu: VolumeManager/Disk/AddDisks
Here is the disk selected.  Output format: [Device_Name]
c1t1d0
Continue operation? [y,n,q,?] (default: y) y
```

Step 2. At the following prompt, specify the disk group to which the disk should be added or press Return to accept `rootdg`:

```
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)
```

- Step 3.** At the following prompt, either press Return to accept the default disk name or enter a disk name:

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

- Step 4.** When the vxdiskadd program asks whether this disk should become a hot-relocation spare, enter **n** (or press Return):

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

- Step 5.** When the vxdiskadd program prompts whether to exclude this disk from hot-relocation use, enter **n** (or press Return).

```
Exclude disk from hot-relocation use? [y,n,q,?]
(default: n) n
```

- Step 6.** To continue with the task, enter **y** (or press Return) at the following prompt:

```
The selected disks will be added to the disk group rootdg
with
default disk names.
c1t1d0
Continue with operation? [y,n,q,?] (default: y) y
```

- Step 7.** The following prompt indicates that this disk has been previously initialized for future Volume Manager use; enter **y** to confirm that you now want to use this disk:

```
The following disk device appears to have been
initialized
already. The disk is currently available as a
replacement disk.
Output format: [Device_Name]
c1t1d0
Use this device? [y,n,q,?] (default: y) y
```

- Step 8.** To reinitialize the disk, enter **y** (or press Return) at the following prompt:

```
The following disk you selected for use appears to
already have
been initialized for the Volume Manager. If you are
certain
the disk has already been initialized for the Volume
```

Manager,
then you do not need to reinitialize the disk device.
Output format: [Device_Name]

c1t1d0

Reinitialize this device? [y,n,q,?] (default: y) **y**

Messages similar to the following now confirm that this disk is being reinitialized for Volume Manager use. You may also be given the option of performing surface analysis on some systems.

Initializing device c1t1d0.

Perform surface analysis (highly recommended)
[y,n,q,?] (default: y) **n**

Adding disk device c1t1d0 to disk group rootdg with disk name
disk03.

NOTE

Output can vary depending on the previous usage of the disk.

To confirm that the disk has been added to the disk group, enter the following command:

vxdisk list

The Volume Manager returns a listing similar to the following:

DEVICE	TYPE	DISK	GROUP	STATUS
c0t0d0	simple	disk04	rootdg	online
c1t0d0	simple	disk01	rootdg	online
c1t1d0	simple	disk03	rootdg	online

Adding a VM Disk to the Hot-Relocation Pool

NOTE

You may need an additional license to use this feature.

Hot-relocation allows the system to automatically react to I/O failure by relocating redundant subdisks to other disks. Hot-relocation then restores the affected Volume Manager objects and data. If a disk has already been designated as a spare in the disk group, the subdisks from the failed disk are relocated to the spare disk. Otherwise, any suitable free space in the disk group is used.

Designating a Disk from the Command Line

To designate a disk as a hot-relocation spare, enter the following command:

```
# vxedit set spare=on diskname
```

For example, to designate disk01 as a spare, enter the following command:

```
# vxedit set spare=on disk01
```

You can use the `vxdisk list` command to confirm that this disk is now a spare; disk01 should be listed with a `spare` flag.

Any VM disk in this disk group can now use this disk as a spare in the event of a failure. If a disk fails, hot-relocation automatically occurs (if possible). You are notified of the failure and relocation through electronic mail. After successful relocation, you may want to replace the failed disk.

Designating a Disk with `vxdiskadm`

To designate a disk as a spare through the `vxdiskadm` main menu, use the following procedure:

- Step 1.** Select menu item 11 (Mark a disk as a spare for a disk group) from the `vxdiskadm` main menu.
- Step 2.** At the following prompt, enter a disk name (such as disk01):

Menu: VolumeManager/Disk/MarkSpareDisk

Use this operation to mark a disk as a spare for a disk group.

This operation takes, as input, a disk name. This is the same name that you gave to the disk when you added the disk to the disk group.

Enter disk name [<disk>, list, q, ?] **disk01**

- Step 3.** At the following prompt, indicate whether you want to add more disks as spares (**y**) or return to the vxdiskadm main menu (**n**):

Mark another disk as a spare? [y,n,q,?] (default: n)

Any VM disk in this disk group can now use this disk as a spare in the event of a failure. If a disk fails, hot-relocation should automatically occur (if possible). You should be notified of the failure and relocation through electronic mail. After successful relocation, you may want to replace the failed disk.

Removing a VM Disk From the Hot-Relocation Pool

NOTE

You may need an additional license to use this feature.

While a disk is designated as a spare, the space on that disk is not used as free space for the creation of Volume Manager objects within its disk group. If necessary, you can free a spare disk for general use by removing it from the pool of hot-relocation disks.

To verify which disks are currently designated as spares, select the **list** menu item from the **vxdiskadm** main menu. Disks that are spares are listed with the **spare** flag.

Removing a Disk with **vxdiskadm**

To remove a disk from the hot-relocation pool, do the following:

Step 1. Select menu item 12 (Turn off the spare flag on a disk) from the **vxdiskadm** main menu.

Step 2. At the following prompt, enter the name of a spare disk (such as **disk01**):

Menu: VolumeManager/Disk/UnmarkSpareDisk

Use this operation to turn off the spare flag on a disk. This operation takes, as input, a disk name. This is the same

name that you gave to the disk when you added the disk to the disk group.

Enter disk name [<disk>,list,q,?] **disk01**

The **vxdiskadm** program displays the following confirmation:

Disk **disk01** in **rootdg** no longer marked as a spare disk.

Step 3. At the following prompt, indicate whether you want to disable more spare disks (**y**) or return to the **vxdiskadm** main menu (**n**):

Removing a VM Disk From the Hot-Relocation Pool

Turn-off spare flag on another disk? [y,n,q,?] (default:
n)

Excluding a Disk from Hot-Relocation Use

NOTE

You may need an additional license to use this feature.

To exclude a disk available from hot-relocation use, use the following command:

```
# vxedit -g disk_group set nohotuse=on disk_name
```

Alternatively, from the vxdiskadm main menu, use the following procedure:

Step 1. Select menu item 15 (Exclude a disk from hot-relocation use) from the vxdiskadm main menu.

Step 2. At the following prompt, enter the disk name (such as **disk01**):

```
Menu: VolumeManager/Disk/UnmarkSpareDisk
```

```
Use this operation to exclude a disk from hot-relocation use.
```

```
This operation takes, as input, a disk name. This is the same name that you gave to the disk when you added the disk to the disk group.
```

```
Enter disk name [<disk>,list,q,?] disk01
```

The vxdiskadm program displays the following confirmation:

```
Excluding disk01 in rootdg from hot-relocation use is complete.
```

Step 3. At the following prompt, indicate whether you want to add more disks to be excluded from hot-relocation (**y**) or return to the vxdiskadm main menu (**n**):

```
Exclude another disk from hot-relocation use? [y,n,q,?]  
(default: n)
```

Including a Disk for Hot-Relocation Use

NOTE

You may need an additional license to use this feature.

Free space is used automatically by hot-relocation on case spare space is not sufficient to relocate failed subdisks. The user can limit this free space usage by hot-relocation by specifying which free disks should not be touched by hot-relocation. If a disk was previously excluded from hot-relocation use, you can undo the exclusion and add the disk back to the hot-relocation pool.

To make a disk available for hot-relocation use, use the following command:

```
# vxedit -g disk_group set nohotuse=off disk_name
```

Alternatively, from the vxdiskadm main menu, use the following procedure:

- Step 1.** Select menu item 16 (Make a disk available for hot-relocation use) from the vxdiskadm main menu.
- Step 2.** At the following prompt, enter the disk name (such as **disk01**):

```
Menu: VolumeManager/Disk/UnmarkSpareDisk
Use this operation to amke a disk available for
hot-relocation
use. This only applies to disks that were previously
excluded
from hot-relocation use. This operation takes, as input,
a disk
name. This is the same name that you gave to the disk
when you
added the disk to the disk group.
Enter disk name [<disk>,list,q,?] disk01
```

The vxdiskadm program displays the following confirmation:

```
Excluding disk01 in rootdg from hot-relocation use is
complete.
```

- Step 3.** At the following prompt, indicate whether you want to add more disks to be excluded from hot-relocation (**y**) or return to the `vxdiskadm` main menu (**n**):

```
Make another disk available for hot-relocation use?  
[y,n,q,?]  
(default: n)
```

Reinitializing a Disk

This section describes how to reinitialize a disk that has previously been initialized for Volume Manager use.

If the disk you want to add has been used before, but not with Volume Manager, use *one* of the following procedures:

- Convert the LVM disk and preserve its information (see the *VERITAS Volume Manager Migration Guide* for more details.)
- Reinitialize the disk, allowing the Volume Manager to configure the disk for Volume Manager. Note that reinitialization does not preserve data on the disk. If you wish to have the disk reinitialized, make sure that the disk does not contain data that should be preserved.

To reinitialize a disk for Volume Manager, use the following procedure:

- Step 1.** Select menu item 1 (Add or initialize one or more disks) from the vxdiskadm main menu.
- Step 2.** At the following prompt, enter the disk device name of the disk to be added to Volume Manager control:

```
Add or initialize disks
Menu: VolumeManager/Disk/AddDisks
Use this operation to add one or more disks to a disk
group.
You can add the selected disks to an existing disk group
or to
a new disk group that will be created as a part of the
operation. The selected disks may also be added to a disk
group
as spares. The selected disks may also be initialized
without
adding them to a disk group leaving the disks available
for use
as replacement disks.
More than one disk or pattern may be entered at the
prompt. Here
are some disk selection examples:
all:all disks
c3 c4t2:all disks on both controller 3 and controller
```

Disk Tasks

Reinitializing a Disk

```
4,target 2
c3t4d0:a single disk
Select disk devices to add:
[<pattern-list>,all,list,q,?] c1t3d0
```

Where *<pattern-list>* can be a single disk, or a series of disks and/or controllers (with optional targets). If *<pattern-list>* consists of multiple items, those items must be separated by white space.

If you do not know the address (device name) of the disk to be added, enter **l** or **list** at the prompt for a complete listing of available disks.

- Step 3.** To continue with the operation, enter **y** (or press Return) at the following prompt:

```
Here is the disk selected.  Output format: [Device_Name]
c1t3d0
Continue operation? [y,n,q,?] (default: y) y
```

- Step 4.** At the following prompt, specify the disk group to which the disk should be added or press Return to accept **rootdg**:

```
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)
```

- Step 5.** At the following prompt, either press Return to accept the default disk name or enter a disk name:

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

- Step 6.** When the **vxdiskadm** program asks whether this disk should become a hot-relocation spare, enter **n** (or press Return):

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

n) **n**

- Step 7.** When the `vxdiskadm` program prompts whether to exclude this disk from hot-relocation use, enter **n** (or press Return).

Exclude disk from hot-relocation use? [y,n,q,?]
(default: n) **n**

- Step 8.** To continue with the operation, enter **y** (or press Return) at the following prompt:

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

`c1t3d0`

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

- Step 9.** If the disk was used for the file system earlier, then the `vxdiskadm` program gives you the following choices:

The following disk device appears to contain a currently
unmounted file system.

`c1`
`t3d0`

Are you sure you want to destroy this file system
[y,n,q,?] (default: n)

- Step 10.** To reinitialize the disk, enter **y** (or press Return) at the following prompt:

Reinitialize this device? [y,n,q,?] (default: y) **y**

Messages similar to the following now confirm that this disk is being
reinitialized for Volume Manager use:

Initializing device `c1t3d0`.
Adding disk device `c1t3d0` to disk group `rootdg` with disk
name `disk40`.

- Step 11.** At the following prompt, indicate whether you want to continue to initialize more disks (**y**) or return to the `vxdiskadm` main menu (**n**):

Add or initialize other disks? [y,n,q,?] (default: n)

Renaming a Disk

If you do not specify a Volume Manager name for a disk, the Volume Manager gives the disk a default name when you add the disk to Volume Manager control. The Volume Manager name is used by the Volume Manager to identify the location of the disk or the disk type. To change the disk name to reflect a change of use or ownership, use the following command:

```
# vxedit rename old_diskname new_diskname
```

To rename disk01 to disk03, use the following command:

```
# vxedit rename disk01 disk03
```

To confirm that the name change took place, use the following command:

```
# vxdisk list
```

The Volume Manager returns the following display:

DEVICE	TYPE	DISK	GROUP	STATUS
c0t0d0	simple	disk04	rootdg	online
c1t0d0	simple	disk03	rootdg	online
c1t1d0	simple	-	-	online

NOTE

By default, Volume Manager names subdisk objects after the VM disk on which they are located. Renaming a VM disk does not automatically rename the subdisks on that disk.

Reserving Disks

By default, the `vxassist` command allocates space from any disk that has free space. You can reserve a set of disks for special purposes, such as to avoid general use of a particularly slow or a particularly fast disk.

To reserve a disk for special purposes, use the following command:

```
# vxedit set reserve=on diskname
```

After you enter this command, the `vxassist` program does not allocate space from the selected disk unless that disk is specifically mentioned on the `vxassist` command line. For example, if `disk03` is reserved, use the following command:

```
# vxassist make vol03 20m disk03
```

The `vxassist` command overrides the reservation and creates a 20 megabyte volume on `disk03`. However, the command:

```
# vxassist make vol04 20m
```

does not use `disk03`, even if there is no free space on any other disk.

To turn off reservation of a disk, use the following command:

```
# vxedit set reserve=off  
  disk_name
```

See `vxedit (1M)` Special Attribute Values for Disk Media.

Displaying Disk Information

Before you use a disk, you need to know if it has been initialized and placed under Volume Manager control. You also need to know if the disk is part of a disk group, because you cannot create volumes on a disk that is not part of a disk group. The `vxdisk list` command displays device names for all recognized disks, the disk names, the disk group names associated with each disk, and the status of each disk.

To display information on all disks that are defined to the Volume Manager, use the following command:

```
# vxdisk list
```

To display details on a particular disk defined to the Volume Manager, use the following command:

```
# vxdisk list (disk_name)
```

Displaying Multipaths Under a VM Disk

The `vxdisk` command is used to display the multipathing information for a particular metadvice. The metadvice is a device representation of a particular physical disk having multiple physical paths from the I/O controller of the system. In the Volume Manager, all the physical disks in the system are represented as metadevices with one or more physical paths.

To view multipathing information for a particular metadvice, use the following command :

```
# vxdisk list (device_name)
```

For example, to view multipathing information for `c1t0d3`, use the following command:

```
# vxdisk list c1t0d3
```

The output is as follows:

```
Device:      c1t0d3
devicetag:   c1t0d3
type:        simple
hostid:      zort
```

```
disk:      name=disk04 id=962923652.362193.zort
timeout:   30
group:     name=rootdg id=962212937.1025.zort
info:      privoffset=128
flags:     online ready private autoconfig autoimport
imported
pubpaths:  block=/dev/vx/dmp/c1t0d3
char=/dev/vx/rdmp/c1t0d3
version:   2.1
iosize:    min=1024 (bytes) max=64 (blocks)
public:    slice=0 offset=1152 len=4101723
private:   slice=0 offset=128 len=1024
update:    time=962923719 seqno=0.7
headers:   0 248
configs:   count=1 len=727
logs:      count=1 len=110
Defined regions:
  config   priv 000017-000247[000231]: copy=01
offset=000000
disabled
  config   priv 000249-000744[000496]: copy=01
offset=000231
disabled
  log      priv 000745-000854[000110]: copy=01
offset=000000
disabled
  lockrgn  priv 000855-000919[000065]: part=00
offset=000000
Multipathing information:
numpaths:  2
c1t0d3  state=enabled  type=secondary
c4t1d3  state=enabled  type=primary
```

Displaying Multipathing Information

To display details on a particular disk defined to the Volume Manager, use the following command:

```
# vxdisk list (device_name)
```

For example, to view multipathing information for c8t15d0, use the following command:

```
# vxdisk list c8t15d0
```

The output is as follows:

```
Device:      c8t15d0
devicetag:   c8t15d0
type:        simple
hostid:      copyy
disk:         name=disk06 id=963453688.1097.copyy
timeout:     30
group:        name=rootdg id=963453659.1025.copyy
info:         privoffset=128
flags:        online ready private autoimport imported
pubpaths:     block=/dev/vx/dmp/c8t15d0
char=/dev/vx/rdmp/c8t15d0
version:     2.1
iosize:       min=1024 (bytes) max=64 (blocks)
public:       slice=0 offset=1152 len=8885610
private:      slice=0 offset=128 len=1024
update:       time=963481049 seqno=0.449
headers:      0 248
configs:      count=1 len=727
logs:         count=1 len=110
Defined regions:
  config      priv 000017-000247[000231]: copy=01
offset=000000
disabled
  config      priv 000249-000744[000496]: copy=01
offset=000231
disabled
  log         priv 000745-000854[000110]: copy=01
offset=000000
disabled
  lockrgn     priv 000855-000919[000065]: part=00
offset=000000
Multipathing information:
numpaths:     2
c8t15d0 state=enabled
c10t15d0      state=enabled
```

Additional information in the form of *type* is shown for disks on active/passive type disk arrays. This information indicates the *primary* and *secondary* paths to the disk; for example, Nike.

The type information is not present for disks on active/active type disk arrays because there is no concept of primary and secondary paths to disks on these disk arrays.

Displaying Disk Information with the vxdiskadm Program

Displaying disk information shows you which disks are initialized, to which disk groups they belong, and the disk status. The `list` command displays device names for all recognized disks, the disk names, the disk group names associated with each disk, and the status of each disk.

To display disk information, use the following procedure:

Step 1. Start the vxdiskadm program with the following command:

```
# vxdiskadm
```

The vxdiskadm main menu is displayed.

Step 2. Select `list` (List disk information) from the vxdiskadm main menu.

Step 3. At the following display, enter the address of the disk you want to see, or enter **all** for a list of all disks:

```
List disk information
Menu: VolumeManager/Disk/ListDisk
Use this menu operation to display a list of disks. You
can
also choose to list detailed information about the disk
at
a specific disk device address.
Enter disk device or "all" [<address>,all,q,?] (default:
all)
```

- If you enter **all**, the Volume Manager displays the device name, diskname, group, and status.
- If you enter the address of the device for which you want information, complete disk information (including the device name, the type of disk, and information about the public and private areas of the disk) is displayed.

Once you have examined this information, press Return to return to the

main menu.

5 **Disk Group Tasks**

Introduction

This chapter describes the operations for managing disks groups.

NOTE

Most Volume Manager commands require superuser or other appropriate privileges.

The following topics are covered in this chapter:

- “Disk Groups”
- “Disk Group Utilities”
- “Creating a Disk Group”
- “Renaming a Disk Group”
- “Importing a Disk Group”
- “Deporting a Disk Group”
- “Upgrading a Disk Group”
- “Moving a Disk Group”
- “Moving Disk Groups Between Systems”
- “Using Disk Groups”
- “Removing a Disk Group”
- “Destroying a Disk Group”
- “Reserving Minor Numbers for Disk Groups”
- “Displaying Disk Group Information”

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 must be added to `rootdg` for you to do the Volume Manager installation procedures.

Even though the `rootdg` is the default disk group, it is not the root disk group. In the current release the root volume group is always under LVM control.

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 `vxkg list diskgroupname`.

You may wish to add a new disk to an already established disk group. For example, the current disks may have insufficient space for the project or work group requirements, especially if these requirements have

changed. You can add a disk to a disk group by following the steps required to add a disk. See Chapter 4, Disk Tasks,.

Disk Group Utilities

The Volume Manager provides a menu interface, `vxdiskadm`, and a command line utility, `vx dg`, to manage disk groups. These utilities are described as follows:

- The `vx diskadm` utility 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.
- The `vx dg` utility is the command-line utility for operating on disk groups. The `vx dg` utility creates new disk groups, adds and removes disks from disk groups, and enables (imports) or disables (deports) access to disk groups. See the `vx dg(1M)` manual page for more information.

Creating a Disk Group

Data related to a particular set of applications or a particular group of users may need to be made accessible on another system. Examples of this are:

- A system has failed and its data needs to be moved to other systems.
- The work load must be balanced across a number of systems.

It is important that the data related to particular application(s) or users be located on an identifiable set of disk, so that when these disks are moved, all data of the application(s) or group of users, and no other data, is moved.

Disks need to be in disk groups before the Volume Manager can use the disks for volumes. Volume Manager always has the root disk group, but you can add more disk groups if needed.

NOTE

The Volume Manager supports a default disk group, `rootdg`, in which all volumes are created if no further specification is given. All commands default to `rootdg` as well.

A disk group can only be created along with a disk. A disk group must have at least one disk associated with it. A disk that does not belong to any disk group must be available when you create a disk group.

To create a disk group in addition to `rootdg`, use the following procedure:

Step 1. Select menu item 1 (Add or initialize one or more disks) from the `vxdiskadm` main menu.

Step 2. At the following prompt, enter the disk device name of the disk to be added to Volume Manager control:

Add or initialize disks

Menu: VolumeManager/Disk/AddDisks

Use this operation to add one or more disks to a disk group.

You can add the selected disks to an existing disk group or
to a new disk group that will be created as a part of

the operation. The selected disks may also be added to a disk group as spares. The selected disks may also be initialized without adding them to a disk group leaving the disks available for use as replacement disks. More than one disk or pattern may be entered at the prompt. Here are some disk selection examples:
all:all disks
c3 c4t2:all disks on both controller 3 and controller 4, target 2
c3t4d2:a single disk
Select disk devices to add:
[<pattern-list>,all,list,q,?] c1
t2d0

Where *<pattern-list>* can be a single disk, or a series of disks and/or controllers (with optional targets). If *<pattern-list>* consists of multiple items, those items must be separated by white space.

If you do not know the address (device name) of the disk to be added, enter **l** or **list** at the prompt for a listing of all disks.

- Step 3.** To continue with the operation, enter **y** (or press Return) at the following prompt:

Here is the disk selected. Output format:
[Device_Name]
c1t2d0
Continue operation? [y,n,q,?] (default: y) **y**

- Step 4.** At the following prompt, specify the disk group to which the disk should be added (in this example, **anotherdg**):

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 Group Tasks

Creating a Disk Group

disk available for future use, specify a disk group name of

"none".

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

anotherdg

- Step 5.** The vxdiskadm utility confirms that no active disk group currently exists with the same name and prompts for confirmation that you really want to create this new disk group:

There is no active disk group named anotherdg.

Create a new group named anotherdg? [y,n,q,?] (default: y) **y**

- Step 6.** At the following prompt, either press Return to accept the default disk name or enter a disk name:

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

- Step 7.** When the vxdiskadm utility asks whether this disk should become a hot-relocation spare, enter **n** (or press Return):

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

- Step 8.** When the vxdiskadm utility prompts whether to exclude this disk from hot-relocation use, enter **n** (or press Return).

Exclude disk from hot-relocation use? [y,n,q,?] (default: n) **n**

- Step 9.** To continue with the operation, enter **y** (or press Return) at the following prompt:

A new disk group will be created named anotherdg and the

selected disks will be added to the disk group with default disk

names.

c1t2d0

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

- Step 10.** If the disk was used for the file system earlier, then the vxdiskadm

utility gives you the following choices:

The following disk device appears to have already been initialized for vxvm use.

c1t2d0

Are you sure you want to re-initialize this disk
[y,n,q,?] (default: n)

Messages similar to the following confirm that this disk is being reinitialized for Volume Manager use:

Initializing device c1t2d0.

Creating a new disk group named anotherdg containing the disk

device c1t2d0 with the name another01.

Step 11. At the following prompt, indicate whether you want to continue to initialize more disks (**y**) or return to the vxdiskadm main menu (**n**):

Add or initialize other disks? [y,n,q,?] (default: n)

Renaming a Disk Group

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 rename a disk group during import, do the following:

```
# vxdbg [-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.

To rename a disk group during deport, use the following command:

```
# vxdbg [-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.

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

- Step 1.** On the original host, identify the disk group ID of the root disk group to be imported with the following command:

```
# vxdisk -s list
```

This command results in output that includes the following.

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

- Step 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.

Step 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.

Importing a Disk Group

Use this menu task to enable access by this system to a disk group. To move a disk group from one system to another, first disable (deport) the disk group on the original system, and then move the disk between systems and enable (import) the disk group.

To import a disk group, use the following procedure:

- Step 1.** Select menu item 7 (Enable access to (import) a disk group) from the vxdiskadm main menu.
- Step 2.** At the following prompt, enter the name of the disk group to import (in this example, **newdg**):

```
Enable access to (import) a disk group
Menu: VolumeManager/Disk/EnableDiskGroup
Use this operation to enable access to a disk group. This
can be
    used as the final part of moving a disk group from one
system to
    another. The first part of moving a disk group is to use
the
    "Remove access to (deport) a disk group" operation on
the
original host.
A disk group can be imported from another host that
failed
    without first deporting the disk group. Be sure that all
disks
    in the disk group are moved between hosts.
If two hosts share a SCSI bus, be very careful to ensure
that
    the other host really has failed or has deported the
disk group.
    If two active hosts import a disk group at the same
time, the
    disk group will be corrupted and will become unusable.
Select disk group to import [<group>,list,q,?] (default:
list)
newdg
```

Once the import is complete, the `vxdiskadm` utility displays the following success message:

```
The import of newdg was successful.
```

Step 3. At the following prompt, indicate whether you want to import another disk group (**y**) or return to the `vxdiskadm` main menu (**n**):

```
Select another disk group? [y,n,q,?] (default: n)
```

Deporting a Disk Group

Use the deport disk group task to disable access to a disk group that is currently enabled (imported) by this system. Deport a disk group if you intend to move the disks in a disk group to another system. Also, deport a disk group if you want to use all of the disks remaining in a disk group for a new purpose.

To deport a disk group, use the following procedure:

- Step 1.** Select menu item 8 (Remove access to (deport) a disk group) from the vxdiskadm main menu.
- Step 2.** At the following prompt, enter the name of the disk group to be deported (in this example, **newdg**):

```
Remove access to (deport) a disk group
Menu: VolumeManager/Disk/DeportDiskGroup
```

```
Use this menu operation to remove access to a disk group
that is
currently enabled (imported) by this system. Deport a
disk
group if you intend to move the disks in a disk group to
another
system. Also, deport a disk group if you want to use all
of the
disks remaining in a disk group for some new purpose.
```

```
You will be prompted for the name of a disk group. You
will also
be asked if the disks should be disabled (offlined). For
removable disk devices on some systems, it is important
to
disable all access to the disk before removing the disk.
Enter name of disk group [<group>,list,q,?] (default:
list)
newdg
```

- Step 3.** At the following prompt, enter **y** if you intend to remove the disks in this disk group:

The requested operation is to disable access to the removable disk group named newdg. This disk group is stored on the following disks:
newdg01 on device c1t1d0
You can choose to disable access to (also known as "offline") these disks. This may be necessary to prevent errors if you actually remove any of the disks from the system.
Disable (offline) the indicated disks? [y,n,q,?]
(default: n) **y**

Step 4. At the following prompt, press Return to continue with the operation:

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

Once the disk group is deported, the vxdiskadm utility displays the following message:

Removal of disk group newdg was successful.

Step 5. At the following prompt, indicate whether you want to disable another disk group (**y**) or return to the vxdiskadm main menu (**n**):

Disable another disk group? [y,n,q,?] (default: n)

Upgrading a Disk Group

Prior to the release of Volume Manager 3.0, the disk group version was automatically upgraded (if needed) when the disk group was imported. Upgrading the disk group makes it incompatible with previous Volume Manager releases.

The Volume Manager 3.0 disk group upgrade feature separates the two operations of importing a disk group and upgrading its version. You can import a disk group of down-level version and use it without upgrading it.

When you want to use new features, the disk group can be upgraded. The disk group upgrade is an explicit operation (as opposed to earlier VxVM versions, where the disk group upgrade was performed when the disk group was imported). Once the upgrade occurs, the disk group becomes incompatible with earlier releases of VxVM that do not support the new disk group version.

NOTE

This information is not applicable for platforms whose first release is Volume Manager 3.0. However, it will be applicable with subsequent releases

Moving a Disk Group

A disk group can be moved between systems, along with its Volume Manager objects (except for `rootdg`). This relocates the disk group configuration to a new system.

To move a disk group across systems, use the following procedure:

Step 1. Unmount and stop all volumes in the disk group on the first system.

Step 2. Deport (disable local access to) the disk group to be moved with the following command:

```
# vxdg deport diskgroup
```

Step 3. Import (enable local access to) the disk group and its disks from the second system with the following command:

```
# vxdg import diskgroup
```

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

```
# vxrecover -g diskgroup -sb
```

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.

To move a disk group between systems, use the following procedure:

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

```
# vxdg  
deport diskgroup
```

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

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

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

```
# vxdg import diskgroup
```

CAUTION

It is very important that all the disks in the disk group be moved to the other system. If they are not moved, the import will fail.

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

```
# vxrecover -g diskgroup -sb
```

You can also 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.

CAUTION

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.

When 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:

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

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

```
# vxldisk clearimport devicename...
```

To clear the locks during import, use the following command:

```
# vxldg -C import diskgroup
```

NOTE

Be careful when using the `vxldisk clearimport` or `vxldg -C import` command on systems that have dual-ported disks. Clearing the locks allows those disks to be accessed at the same time from multiple hosts and can result in corrupted data.

You may want to import a disk group when some disks are not available. The `import` operation fails if some disks for the disk group cannot be found among the disk drives attached to the system. When the `import` operation fails, one of several error messages is displayed.

The following message indicates a fatal error that requires hardware repair or the creation of a new disk group, and recovery of the disk group configuration and data.

```
vxldg: Disk group groupname: import failed: Disk group  
has no valid
```

Disk Group Tasks

Moving Disk Groups Between Systems

configuration copies

The following message indicates a recoverable error.

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

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

```
# vxldg -f import diskgroup
```

NOTE

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 the `vxldiskadm` utility. To deport a disk group by using `vxldiskadm`, 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 `vxldiskadm` 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.

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`, use the following 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, to create a volume on disk `mktdg01` without specifying the disk group name, use the following command:

```
# 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`. Remember to 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 the following 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, or imported to use on other systems.

Destroying a Disk Group

The `vx dg` command provides a `destroy` option that removes a disk group from the system and frees the disks in that disk group for reinitialization so they can be used in other disk groups. Remove disk groups that are not needed with the `vx dg destroy` command so that the disks can be used by other disk groups. To remove unnecessary disk groups, use the following command:

```
# vx dg destroy diskgroup
```

The `vx dg deport` command can still be used to make disks inaccessible. The Volume Manager prevents disks in a deported disk group from being used in other disk groups.

Reserving Minor Numbers for Disk Groups

Disk groups can be moved between systems. When you allocate volume device numbers in separate ranges for each disk group, 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.

To set a base volume device minor number for a disk group, use the following command:

```
# vxdbg 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 `vxdbg 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, see the `vxdbg(1M)` manual page.

Displaying Disk Group Information

To use disk groups, you need to know their names and what disks belong to each group.

To display information on existing disk groups, use the following command:

```
# vxdg list
```

The Volume Manager returns the following listing of current disk groups:

NAME	STATE	ID
rootdg	enabled	730344554.1025.tweety
newdg	enabled	731118794.1213.tweety

To display more detailed information on a specific disk group (such as rootdg), use the following command:

```
# vxdg list rootdg
```

The output is as follows:

```
Group:      rootdg
dgid:      962910960.1025.bass
import-id: 0.1
flags:
version:   70
local-activation: read-write
detach-policy: local
copies:    nconfig=default nlog=default
config:    seqno=0.1183 permrlen=727 free=722 templen=2
loglen=110
config disk c0t10d0 copy 1 len=727 state=clean online
config disk c0t11d0 copy 1 len=727 state=clean online
log disk c0t10d0 copy 1 len=110
log disk c0t11d0 copy 1 len=110
```

To verify the disk group ID and name associated with a specific disk (for example, to import the disk group), use the following command:

```
# vxdisk -s list <diskmedianame>
```

This command provides output that includes the following information for the specified disk. For example, output for disk c0t12d0 is as follows:

Disk Group Tasks
Displaying Disk Group Information

```
Disk:    c0t12d0
type:    simple
flags:    online ready private autoconfig autoimport
imported
diskid:   963504891.1070.bass
dgname:   newdg
dgid:     963504895.1075.bass
hostid:   bass
info:     privoffset=128
```


6 **Volume Tasks**

Introduction

This chapter describes how to create and maintain a system configuration under Volume Manager control. It includes information about creating, removing, and maintaining Volume Manager objects. Volume Manager objects include:

- Volumes

Volumes are logical device that appear to data management systems as physical disk partition devices. Volumes provide enhanced recovery, data availability, performance, and storage configuration options. Volume tasks are used to protect data in the event of device failures.

- Plexes

Plexes are logical groupings of subdisks that create an area of disk space independent of physical disk size or other restrictions. Replication (mirroring) of disk data is done by creating multiple plexes for a single volume. Each plex contains an identical copy of the volume data. Because each plex must reside on different disks, the replication provided by mirroring prevents data loss in the event of a single-point disk-subsystem failure. Multiple plexes also provide increased data integrity and reliability.

- Subdisks

Subdisks are low-level building blocks in a Volume Manager configuration.

Volume, Plex, and Subdisk Tasks

This chapter provides information on the following tasks:

Volume Tasks

- “Creating a Volume”
- “Starting a Volume”
- “Stopping a Volume”
- “Resizing a Volume”

- “Removing a Volume”
- “Mirroring a Volume”
- “Displaying Volume Configuration Information”
- “Preparing a Volume to Restore From Backup”
- “Recovering a Volume”
- “Moving Volumes from a VM Disk”
- “Adding a RAID-5 Log”
- “Removing a RAID-5 Log”
- “Adding a DRL Log”
- “Removing a DRL Log”

Plex Tasks

- “Creating Plexes”
- “Associating Plexes”
- “Dissociating and Removing Plexes”
- “Displaying Plex Information”
- “Changing Plex Attributes”
- “Changing Plex Status: Detaching and Attaching Plexes”
- “Moving Plexes”
- “Copying Plexes”

Subdisk Tasks

- “Creating Subdisks”
- “Removing Subdisks”
- “Moving Subdisks”
- “Splitting Subdisks”
- “Joining Subdisks”
- “Associating Subdisks”
- “Dissociating Subdisks”

- “Changing Subdisk Attributes”

Performing Online Backup

- “FastResync (Fast Mirror Resynchronization)”
- “Mirroring Volumes on a VM Disk”
- “Mirroring Volumes on a VM Disk”

NOTE

Some Volume Manager commands require superuser or other appropriate privileges.

Creating a Volume

Volumes are created to take advantage of the Volume Manager concept of virtual disks. A file system can be placed on the volume to organize the disk space with files and directories. In addition, applications such as databases can be used to organize data on volumes.

The Volume Manager allows you to create volumes with the following layout types:

- Concatenated
- Striped
- RAID-5
- Mirrored
- Striped and Mirrored
- Mirrored and Striped
- Layered Volume

NOTE

You may need an additional license to use this feature.

The `vxassist` command provides the simplest way to create new volumes. To create a volume, use the following command:

```
# vxassist make volume_name length [attributes]
```

where `make` is the keyword for volume creation, *volume_name* is a name you give to the volume, and *length* specifies the number of sectors (by default) in the volume. The length can be specified in kilobytes, megabytes, or gigabytes by using a suffix character of `k`, `m`, or `g`, respectively. See the `vxintro(1M)` manual page for more information on specifying units of length when creating volumes. Additional attributes can be specified, as appropriate.

By default, the `vxassist` command creates volumes in the `rootdg` disk group. Another disk group can be specified by including `-g diskgroup` in the `vxassist` command line.

Creating a volume in the disk group `rootdg` creates two device node files

that can be used to access the volume:

- `/dev/vx/dsk/volume_name` (the block device node for the volume)
- `/dev/vx/rdisk/volume_name` (the raw device node for the volume)

For volumes in `rootdg` and disk groups other than `rootdg`, these names include the disk group name, as follows:

- `/dev/vx/dsk/diskgroup_name/volume_name`
- `/dev/vx/rdisk/diskgroup_name/volume_name`

The following section, “Creating a Concatenated Volume” describes the simplest way to create a (default) volume. Later sections describe how to create volumes with specific attributes.

Creating a Concatenated Volume

By default, the `vxassist` command creates a concatenated volume that uses one or more sections of disk space. On a fragmented disk, this allows you to put together a volume larger than any individual section of free disk space available.

If there is not enough space on a single disk, `vxassist` creates a spanned volume. A spanned volume is a concatenated volume with sections of disk space spread across more than one disk. A spanned volume can be larger than the single largest disk, since it takes space from more than one disk.

Creating a Concatenated Volume on Any Disk

If no disk is specified, the Volume Manager selects a disk on which to create the volume.

To create a concatenated, default volume, use the following command:

```
# vxassist make volume_name length
```

where *volume_name* is the name of the volume and *length* specifies the length of the volume in sectors (unless another unit of size is specified with a suffix character).

When you resize a volume, you can specify the length of a new volume in sectors, kilobytes, megabytes, or gigabytes. The unit of measure is added as a suffix to the length (s, m, k, or g). If no unit is specified, sectors are assumed.

For example, to create the volume `voldefault` with a length of 10 megabytes, use the following command:

```
# vxassist make voldefault 10m
```

Creating a Concatenated Volume on a Specific Disk

The Volume Manager automatically selects the disk(s) each volume resides on, unless you specify otherwise. If you want a volume to reside on a specific disk, you must designate that disk for the Volume Manager. More than one disk can be specified.

To create a volume on a specific disk, use the following command:

```
# vxassist make volume_name length diskname [...]
```

For example, to create the volume `volspecific` on `disk03`, use the following command:

```
# vxassist make volspecific 3m disk03
```

Creating a Striped Volume

A striped volume contains at least one plex that consists of two or more subdisks located on two or more physical disks. For more information on striping, see “Striping (RAID-0)” and “Striping Guidelines”.

To create a striped volume, use the following command:

```
# vxassist make volume_name length layout=stripe
```

For example, to create the striped volume `volzebra`, use the following command:

```
# vxassist make volzebra 10m layout=stripe
```

This creates a striped volume with the default stripe unit size on the default number of disks.

Indicate the disks on which the volumes are to be created by specifying the disk names at the end of the command line. For example, to create a 30 megabyte striped volume on three specific disks (`disk03`, `disk04`, and `disk05`), use the following command:

```
# vxassist make stripevol 30m layout=stripe disk03  
disk04\
```

disk05

Creating a RAID-5 Volume

NOTE

You may need an additional license to use this feature.

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. A RAID-5 volume can also contain one or more RAID-5 log plexes, which are used to log information about data and parity being written to the volume. For more information on RAID-5 volumes, see “RAID-5”.

To create a RAID-5 volume, use the following command:

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

For example, to create the RAID-5 volume volraid, use the following command:

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

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

Creating a Mirrored Volume

NOTE

You may need an additional license to use this feature.

To create a new mirrored volume, use the following command:

```
# vxassist make volume_name length layout=mirror
```

For example, to create the mirrored volume, volmir, use the following command:

```
# vxassist make volmir 5m layout=mirror
```


Starting a Volume

Starting a volume affects its availability to the user. Starting a volume changes its state, makes it available for use, and changes the volume state from DISABLED or DETACHED to ENABLED. The success of this task depends on the ability to enable a volume. If a volume cannot be enabled, it remains in its current state. To start a volume, use the following command:

```
# vxrecover -s volume_name ...
```

To start all DISABLED volumes, use the following command:

```
# vxrecover -s
```

If all mirrors of the volume become STALE, you can place the volume in maintenance mode. Then you can view the plexes while the volume is DETACHED and determine which plex to use for reviving the others. To place a volume in maintenance mode, use the following command:

```
# vxvol maint volume_name
```

To assist in choosing the revival source plex, list the unstarted volume and display its plexes. For example, to take plex vol01-02 offline, use the following command:

```
# vxmend off vol01-02
```

The vxmend command can change the state of an OFFLINE plex of a DISABLED volume to STALE. A vxvol start command on the volume then revives the plex. For example, to put a plex named vol01-02 in the STALE state, use the following command:

```
# vxmend on vol01-02
```

Listing Unstartable Volumes

An unstartable volume can be incorrectly configured or have other errors or conditions that prevent it from being started. To display unstartable volumes, use the vxinfo command. The vxinfo command displays information on the accessibility and usability of one or more volumes:

```
# vxinfo [volume_name]
```

Stopping a Volume

Stopping a volume renders it unavailable to the user. In a stopped volume, the volume state is changed from ENABLED or DETACHED to DISABLED. If the command cannot stop it, the volume remains in its current state. To stop a volume, use the following command:

```
# vxvol stop volume_name ...
```

For example, to stop a volume named vol01, use the following command:

```
# vxvol stop vol01
```

To stop all ENABLED volumes, use the following command:

```
# vxvol stopall
```

If all mirrors of the volume become STALE, place the volume in maintenance mode. You can then view the plexes while the volume is DETACHED and determine which plex to use for reviving the others. To place a volume in maintenance mode, use the following command:

```
# vxvol maint volume_name
```

To assist in choosing the revival source plex, list the unstarted volume and display its plexes. For example, to take plex vol01-02 offline, use the following command:

```
# vxmend off vol01-02
```

The vxmend command can change the state of an OFFLINE plex of a DISABLED volume to STALE. A vxvol start command on the volume then revives the plex. For example, to put a plex named vol01-02 in the STALE state, use the following command:

```
# vxmend on vol01-02
```

Resizing a Volume

Resizing a volume changes the volume size. To resize a volume, use either the `vxassist`, `vxvol`, or `vxresize` commands.

If the volume is not large enough for the amount of data that needs to be stored in it, extend the length of the volume. If a volume is increased in size, the `vxassist` command automatically locates available disk space.

When you resize a volume, you can specify the length of a new volume in sectors, kilobytes, megabytes, or gigabytes. The unit of measure is added as a suffix to the length (s, m, k, or g). If no unit is specified, sectors are assumed.

CAUTION

Do not shrink a volume below the size of the file system. If you have a VxFS file system, shrink the file system first, and then shrink the volume. If you do not shrink the file system first, you risk unrecoverable data loss.

Resizing Volumes With the `vxassist` Command

Four modifiers are used with the `vxassist` command to resize a volume, as follows:

- `growto`—increase volume to specified length
- `growby`—increase volume by specified amount
- `shrinkto`—reduce volume to specified length
- `shrinkby`—reduce volume by specified amount

Extending to a Given Length

To extend a volume *to* a specific length, use the following command:

```
# vxassist growto volume_name length
```

For example, to extend `volcat` to 2000 sectors, use the following command:

```
# vxassist growto volcat 2000
```

Extending by a Given Length

To extend a volume *by* a specific length, use the following command:

```
# vxassist growby volume_name length
```

For example, to extend volcat by 100 sectors, use the following command:

```
# vxassist growby volcat 100
```

Shrinking to a Given Length

To shrink a volume *to* a specific length, use the following command:

```
# vxassist shrinkto volume_name length
```

For example, to shrink volcat to 1300 sectors, use the following command:

```
# vxassist shrinkto volcat 1300
```

Do not shrink the volume below the current size of the file system or database using the volume. The vxassist shrinkto command can be safely used on empty volumes.

Shrinking by a Given Length

To shrink a volume *by* a specific length, use the following command:

```
# vxassist shrinkby volume_name length
```

For example, to shrink volcat by 300 sectors, use the following command:

```
# vxassist shrinkby volcat 300
```

Resizing Volumes with the vxvol Command

To change the length of a volume using the vxvol set command, use the following command:

```
# vxvol set len=value ... volume_name ...
```

For example, to change the length to 100000 sectors, use the following command:

```
# vxvol set len=100000 vol01
```

NOTE

The `vxvol set len` command cannot increase the size of a volume unless the needed space is available in the plexes of the volume. When the size of a volume is reduced using the `vxvol set len` command, the freed space is not released into the free space pool.

Changing the Volume Read Policy

Volume Manager offers the choice of the following read policies:

- **round**—reads each plex in turn in “round-robin” fashion for each nonsequential I/O detected. Sequential access causes only one plex to be accessed. This takes advantage of the drive or controller read-ahead caching policies.
- **prefer**—reads first from a plex that has been named as the preferred plex.
- **select**—chooses a default policy based on plex associations to the volume. If the volume has an enabled striped plex, the **select** option defaults to preferring that plex; otherwise, it defaults to round-robin.

The read policy can be changed from **round** to **prefer** (or the reverse), or to a different preferred plex. The `vxvol rdpol` command sets the read policy for a volume.

To set the read policy to **round**, use the following command:

```
# vxvol rdpol round volume_name
```

For example, to set the read policy for volume `vol01` to a round-robin read, use the following command:

```
# vxvol rdpol round vol01
```

To set the read policy to **prefer**, use the following command:

```
# vxvol rdpol prefer volume_name preferred_plex_name
```

For example, to set the policy for `vol01` to read preferentially from the plex `vol01-02`, use the following command:

```
# vxvol rdpol prefer vol01 vol01-02
```

To set the read policy to **select**, use the following command:

```
# vxvol rdpol select volume_name
```

Resizing Volumes with the vxresize Command

Use the `vxresize` command to resize a volume containing a file system. Although other commands can be used to resize volumes containing file systems, the `vxresize` command offers the advantage of automatically resizing the file system as well as the volume. For details on how to use the `vxresize` command, see the `vxresize(1M)` manual page. Note that only `vxfs` and `hfs` file systems can be resized with the `vxresize` command.

Removing a Volume

Once a volume is no longer necessary (it is inactive and archived, for example), remove the volume and free up the disk space for other uses.

Before removing a volume, refer to the following procedure:

- Step 1.** Remove all references to the volume.
- Step 2.** If the volume is mounted as a file system, unmount it with the command:
- Step 3.** If the volume is listed in `/etc/fstab`, remove its entry.
- Step 4.** Make sure that the volume is stopped with the command:

```
# umount /dev/vx/dsk/volume_name
```

```
# vxvol stop volume_name
```

The `vxvol stop` command stops all VM activity to the volume.

After following these steps, remove the volume with one of the following commands:

```
# vxedit rm volume_name  
# vxedit -rf rm volume_name
```

The `-r` option indicates recursive removal, which means the removal of all plexes associated with the volume and all subdisks associated with those plexes. The `-f` option forces removal, and is necessary if the volume is enabled.

NOTE

The `-r` option of the `vxedit` command removes multiple objects.

You can also remove an entire volume with the `vxassist` command. Use the keywords `remove` and `volume` and provide the volume name on the command line as shown in the following example:

```
# vxassist remove volume volume_name
```

Mirroring a Volume

NOTE

You may need an additional license to use this feature.

A mirror is a copy of a volume. The mirror copy is not stored on the same disk(s) as the original copy of the volume. Mirroring a volume ensures that the data in that volume is not lost if one of your disks fails.

NOTE

To mirror the root disk, use `vxrootmir` (1M). See the manual page for details.

Creating a Volume with Dirty Region Logging Enabled

To create a mirrored volume with Dirty Region Logging (DRL) enabled, create a mirrored volume with a log with this command:

```
# vxassist make volume_name length layout=mirror,log
```

The `vxassist` command creates one log plex for each log subdisk, by default.

Mirroring an Existing Volume

A mirror (plex) can be added to an existing volume with the `vxassist` command, as follows:

```
# vxassist mirror volume_name
```

For example, to create a mirror of the volume `voltest`, use the following command:

```
# vxassist mirror voltest
```

Another way to mirror an existing volume is by first creating a plex and then associating it with a volume, using the following commands:

```
# vxmake plex plex_name sd=subdisk_name ...
```



```
# vxplex att volume_name plex_name
```

Mirroring All Volumes

To mirror all existing volumes on the system to available disk space, use the following command:

```
# /etc/vx/bin/vxmirror -g diskgroup -a
```

To configure the Volume Manager to create mirrored volumes by default, use the following command:

```
# /etc/vx/bin/vxmirror -d yes
```

If you make this change, you can still make unmirrored volumes by specifying `nmirror=1` as an attribute to the `vxassist` command. For example, to create an unmirrored 20-megabyte volume named `nomirror`, use the following command:

```
# vxassist make nomirror 20m nmirror=1
```

Mirroring Volumes on a VM Disk

Mirroring the volumes on a VM disk gives you one or more copies of your volumes in another disk location. By creating mirror copies of your volumes, you protect your system against loss of data in case of a disk failure. This task only mirrors concatenated volumes. Volumes that are already mirrored or that contain subdisks that reside on multiple disks are ignored.

To mirror volumes on a disk, make sure that the target disk has an equal or greater amount of space as the originating disk and then do the following:

Step 1. Select menu item 5 (Mirror volumes on a disk) from the `vxdiskadm` main menu.

Step 2. At the following prompt, enter the disk name of the disk that you wish to mirror:

```
Mirror volumes on a disk
Menu: VolumeManager/Disk/Mirror
```

This operation can be used to mirror volumes on a disk. These

volumes can be mirrored onto another disk or onto any available disk space. Volumes will not be mirrored if they are already mirrored. Also, volumes that are comprised of more than one subdisk will not be mirrored.
Enter disk name [<disk>,list,q,?] **disk02**

- Step 3.** At the following prompt, enter the target disk name (this disk must be the same size or larger than the originating disk):

You can choose to mirror volumes from disk disk02 onto any available disk space, or you can choose to mirror onto a specific disk. To mirror to a specific disk, select the name of that disk. To mirror to any available disk space, select "any".
Enter destination disk [<disk>,list,q,?] (default: any)
disk01

- Step 4.** At the following prompt, press Return to make the mirror:

The requested operation is to mirror all volumes on disk disk02 in disk group rootdg onto available disk space on disk disk01.

NOTE: This operation can take a long time to complete.
Continue with operation? [y,n,q,?] (default: y)

The vxdiskadm program displays the status of the mirroring operation, as follows:

Mirror volume voltest-bk00 ...
Mirroring of disk disk01 is complete.

- Step 5.** At the following prompt, indicate whether you want to mirror volumes on another disk (y) or return to the vxdiskadm main menu (n):

Mirror volumes on another disk? [y,n,q,?] (default: n)

Backing Up Volumes Using Mirroring

If a volume is mirrored, backup can be done on that volume by taking one of the volume mirrors offline for a period of time. This removes the need for extra disk space for the purpose of backup only. However, it also removes redundancy of the volume for the duration of the time needed for the backup to take place.

NOTE

The information in this section does not apply to RAID-5.

You can perform backup of a mirrored volume on an active system with these steps:

Step 1. Optionally stop user activity for a short time to improve the consistency of the backup.

Step 2. Dissociate one of the volume mirrors (vol01-01, for this example) using the following command:

```
# vxplex dis vol01-01
```

Step 3. Create a new, temporary volume that uses the dissociated plex, using the following command:

```
# vxmake -U gen vol tempvol plex=vol01-01
```

Step 4. Start the temporary volume, using the following command:

```
# vxvol start tempvol
```

Step 5. Perform appropriate backup procedures, using the temporary volume.

Step 6. Stop the temporary volume, using the following command:

```
# vxvol stop tempvol
```

Step 7. Dissociate the backup plex from its temporary volume, using the following command:

```
# vxplex dis vol01-01
```

Step 8. Reassociate the backup plex with its original volume to regain redundancy of the volume, using the following command:

```
# vxplex att vol01 vol01-01
```

Step 9. Remove the temporary volume, using the following command:

```
# vxedit rm tempvol
```

For information on an alternative online backup method using the `vxassist` command, see “Performing Online Backup”.

Removing a Mirror

When a mirror is no longer needed, you can remove it. Removal of a mirror is required in the following instances:

- to provide free disk space
- to reduce the number of mirrors in a volume to increase the length of another mirror and its associated volume. The plexes and subdisks are removed, then the resulting space can be added to other volumes
- to remove a temporary mirror created to backup a volume and is no longer required

to change the layout of a mirror from concatenated to striped, or back

NOTE

The last valid plex associated with a volume cannot be removed.

CAUTION

To save the data on a mirror to be removed, the configuration of that mirror must be known. Parameters from that configuration (stripe unit size and subdisk ordering) are critical to the creation of a new mirror to contain the same data. Before this type of mirror is removed, its configuration must be recorded.

To dissociate and remove a mirror from the associated volume, use the following command:

```
# vxplex -o rm dis plex_name
```

For example, to dissociate and remove a mirror named `vol01-02`, use the following command:

```
# vxplex -o rm dis vol01-02
```

This command removes the mirror `vol01-02` and all associated subdisks.

You can first dissociate the plex and subdisks, then remove them with the commands: **# vxplex dis *plex_name*** **# vxedit -r rm *plex_name***

Together, these commands accomplish the same as **vxplex -o rm dis**.

Displaying Volume Configuration Information

You can use the `vxprint` command to display information about how a volume is configured.

To display the volume, plex, and subdisk record information for all volumes in the system, use the following command:

```
# vxprint -ht
```

You can display volume-related information for a specific volume, use the following command:

```
# vxprint -t volume_name
```

For example, to display information about the `voldef` volume, use the following command:

```
# vxprint -t voldef
```

Preparing a Volume to Restore From Backup

It is important to make backup copies of your volumes. This provides a copy of the data as it stands at the time of the backup. Backup copies are used to restore volumes lost due to disk failure, or data destroyed due to human error. The Volume Manager allows you to back up volumes with minimal interruption to users.

To back up a volume with the `vxassist` command, use the following procedure:

- Step 1.** Create a snapshot mirror of the volume to be backed up.

The `vxassist snapstart` task creates a write-only backup mirror, which is attached to and synchronized with the volume to be backed up. When synchronized with the volume, the backup mirror is ready to be used as a snapshot mirror. However, it continues being updated until it is detached during the actual snapshot portion of the procedure. This may take some time, depending on the volume size.

To create a snapshot mirror for a volume, use the following command:

```
# vxassist snapstart volume_name
```

For example, to create a snapshot mirror of a volume called `voldef`, use the following command:

```
# vxassist snapstart voldef
```

- Step 2.** Choose a suitable time to create a snapshot volume.

If possible, plan to take the snapshot at a time when users are accessing the volume as little as possible.

- Step 3.** Create a snapshot volume that reflects the original volume at the time of the snapshot.

The online backup procedure is completed by running the `vxassist snapshot` command on the volume with the snapshot mirror. This task detaches the finished snapshot mirror, creates a new normal volume, and attaches the snapshot mirror to it. The snapshot then becomes a read-only volume. This step should only take a few minutes.

To create a snapshot volume, use the following command:

```
# vxassist snapshot volume_name new_volume_name
```

For example, to create a snapshot volume of voldef, use the following command:

```
# vxassist snapshot voldef snapvol
```

The snapshot volume can now be used by backup utilities, while the original volume continues to be available for applications and users.

The snapshot volume occupies as much space as the original volume. To avoid wasting space, remove the snapshot volume when your backup is complete.

Recovering a Volume

A system crash or an I/O error can corrupt one or more plexes of a volume and leave no plex CLEAN or ACTIVE. You can mark one of the plexes CLEAN and instruct the system to use that plex as the source for reviving the others.

To place a plex in the CLEAN state, use the following command:

```
# vxmend fix clean plex_name
```

For example, to place the plex named vol01-02 in the CLEAN state, use the following command:

```
# vxmend fix clean vol01-02
```

For more information, see the vxmend(1M) manual page. For more information on recovery, see Chapter 8, Recovery,.

Moving Volumes from a VM Disk

Before you disable or remove a disk, you can move the data from that disk to other disks on the system. To do this, ensure that the target disks have sufficient space, and then use the following procedure:

- Step 1.** Select menu item 6 (Move volumes from a disk) from the vxdiskadm main menu.
- Step 2.** At the following prompt, enter the disk name of the disk whose volumes you wish to move, as follows:

```
Move volumes from a disk
Menu: VolumeManager/Disk/Evacuate
```

Use this menu operation to move any volumes that are using a disk onto other disks. Use this menu immediately prior to removing a disk, either permanently or for replacement. You can specify a list of disks to move volumes onto, or you can move

the volumes to any available disk space in the same disk group.

NOTE: Simply moving volumes off of a disk, without also removing

the disk, does not prevent volumes from being moved onto the disk by future operations. For example, using two consecutive move operations may move volumes from the second disk to the first.

Enter disk name [<disk>,list,q,?] **disk01**

After the following display, you can optionally specify a list of disks to which the volume(s) should be moved.

You can now specify a list of disks to move onto.

Specify a

list of disk media names (e.g., disk01) all on one line separated by blanks. If you do not enter any disk media names,

then the volumes will be moved to any available space in

the
disk group.

Step 3. At the following prompt, press Return to move the volumes:

```
Requested operation is to move all volumes from disk  
disk01 in  
group rootdg.
```

NOTE: This operation can take a long time to complete.
Continue with operation? [y,n,q,?] (default: y)

As the volumes are moved from the disk, the vxdiskadm program
displays the status of the operation:

```
Move volume voltest ...  
Move volume voltest-bk00 ...
```

When the volumes have all been moved, the vxdiskadm program
displays the following success message:

```
Evacuation of disk disk01 is complete.
```

Step 4. At the following prompt, indicate whether you want to move volumes
from another disk (**y**) or return to the vxdiskadm main menu (**n**):

```
Move volumes from another disk? [y,n,q,?] (default: n)
```

Adding a RAID-5 Log

NOTE

You may need an additional license to use this feature.

Only one RAID-5 plex can exist per RAID-5 volume. Any additional plexes become RAID-5 log plexes, which are used to log information about data and parity being written to the volume. When a RAID-5 volume is created using the `vxassist` command, a log plex is created for that volume by default.

To add a RAID-5 log to an existing volume, use the following command:

```
# vxassist addlog volume_name
```

To create a log for the RAID-5 volume `volraid`, use the following command:

```
# vxassist addlog volraid
```

Removing a RAID-5 Log

NOTE

You may need an additional license to use this feature.

To remove a RAID-5 log, first dissociate the log from its volume and then remove the log and any associated subdisks completely.

To remove a RAID-5 log from an existing volume, use the following procedure:

- Dissociate and remove the log from its volume in one operation with the following command:

```
# vxplex -o rm dis plex_name
```

- To identify the log plex, use the following command:

```
# vxprint -ht volume_name
```

where *volume_name* is the name of the RAID-5 volume. This produces output that lists a plex with the STATE field of LOG.

For example, to disassociate and remove the log plex volraid-02 from volraid, use the following command:

```
# vxplex -o rm dis volraid-02
```

You can also remove a RAID-5 log with the vxassist command, as follows:

```
# vxassist remove log volume_name
```

Use the attribute `nlog=` to specify the number of logs to be removed. By default, the vxassist command removes one log.

For more information on volume STATES, see Chapter 8, Recovery.

Adding a DRL Log

To put Dirty Region Logging into effect for a volume, a log subdisk must be added to that volume and the volume must be mirrored. Only one log subdisk can exist per plex.

To add a DRL log to an existing volume, use the following command:

```
# vxassist addlog volume_name
```

For example, to create a log for the volume vol03, use the following command:

```
# vxassist addlog vol03
```

When the vxassist command is used to add a log subdisk to a volume, a log plex is also created to contain the log subdisk by default.

Once created, the plex containing a log subdisk can be treated as a regular plex. Data subdisks can be added to the log plex. The log plex and log subdisk can be removed using the same procedures used to remove ordinary plexes and subdisks.

Removing a DRL Log

You can also remove a log with the `vxassist` command as follows.

```
# vxassist remove log volume_name
```

Use the attribute `nlog=` to specify the number of logs to be removed. By default, the `vxassist` command removes one log.

Creating Plexes

The `vxmake` command creates Volume Manager objects, such as plexes. When you create a plex, you identify subdisks and associate them to the plex that you want to create.

To create a plex from existing subdisks, use the following command:

```
# vxmake plex plex_name sd=subdisk_name,...
```

For example, to create a concatenated plex named `vol01-02` using two existing subdisks named `disk02-01` and `disk02-02`, use the following command:

```
# vxmake plex vol01-02 sd=disk02-01,disk02-02
```

Creating a Striped Plex

To create a striped plex, you must specify additional attributes. For example, to create a striped plex named `pl-01` with a stripe width of 32 sectors and 2 columns, use the following command:

```
# vxmake plex pl-01 layout=stripe stwidth=32 ncolumn=2\  
sd=disk01-01,disk02-01
```

To use a plex to build a volume, you must associate the plex with the volume. For more information, see the following section, “Associating Plexes”

Associating Plexes

A plex becomes a participating plex for a volume by associating the plex with the volume. To associate a plex with an existing volume, use the following command:

```
# vxplex att volume_name plex_name
```

For example, to associate a plex named vol01-02 with a volume named vol01, use the following command:

```
# vxplex att vol01 vol01-02
```

If the volume has not been created, a plex (or multiple plexes) can be associated with the volume to be created as part of the volume create command. To associate a plex with the volume to be created, use the following command:

```
# vxmake -U usetype vol volume_name plex=plex_name1,  
plex_name2...
```

For example, to create a mirrored, fsgen-type volume named home and associate two existing plexes named home-1 and home-2, use the following command:

```
# vxmake -hfsgen vol home plex=home-1,home-2
```

NOTE

You can also use the following command on an existing volume to add and associate a plex: **vxassist mirror volume_name**

Dissociating and Removing Plexes

When a plex is no longer needed, you can remove it. Remove a plex for the following reasons:

- to provide free disk space
 - to reduce the number of mirrors in a volume so you can increase the length of another mirror and its associated volume. When the plexes and subdisks are removed, the resulting space can be added to other volumes
 - to remove a temporary mirror that was created to back up a volume and is no longer needed
- to change the layout of a plex

CAUTION

To save the data on a plex to be removed, the configuration of that plex must be known. Parameters from that configuration (stripe unit size and subdisk ordering) are critical to the creation of a new plex to contain the same data. Before a plex is removed, you must record its configuration. See “Displaying Plex Information” for more information.

To dissociate and remove a plex from the associated volume, use the following command:

```
# vxplex -o rm dis plex_name
```

For example, to dissociate and remove a plex named vol01-02, use the following command:

```
# vxplex -o rm dis vol01-02
```

This command removes the plex vol01-02 and all associated subdisks.

You can first dissociate the plex and subdisks, and then remove them with the following commands:

```
# vxplex dis plex_name # vxedit  
-r rm plex_name
```

When used together, these commands produce the same result as the vxplex -o rm dis command.

Displaying Plex Information

Listing plexes helps identify free plexes for building volumes. Using the `vxprint` command with the `plex (-p)` option lists information about all plexes.

To display detailed information about all plexes in the system, use the following command:

```
# vxprint -lp
```

To display detailed information about a specific plex, use the following command:

```
# vxprint -l plex_name
```

The `-t` option prints a single line of information about the plex. To list free plexes, use the following command:

```
# vxprint -pt
```

Changing Plex Attributes

CAUTION

Change plex attributes with extreme care, and only if necessary.

The `vxedit` command changes the attributes of plexes and other volume Manager objects. To change plex attributes, use the following command:

```
# vxedit set field=value ... plex_name ...
```

The `comment` field and the `putil` and `tutil` fields are used by Volume Manager commands after plex creation. The `putil` field attributes are maintained on reboot; `tutil` fields are temporary and are not retained on reboot.

Both the `putil` and `tutil` fields have three functions and are numbered according to those functions. These fields can be modified as needed. Volume Manager uses the utility fields marked `putil0` and `tutil0`. Other VERITAS products use fields marked `putil1` and `tutil1`. Fields marked `putil2` and `tutil2` are user fields. Table 6-1, The `putil[n]` and `tutil[n]` Fields, lists the functions of the `putil` and `tutil` fields.

Table 6-1 The `putil[n]` and `tutil[n]` Fields

Field	Description of Utility Fields
<code>putil0</code>	Reserved for use by Volume Manager commands and is retained on reboot.
<code>putil1</code>	Reserved for use by high-level utilities such as the graphical user interface. This field is retained on reboot.
<code>putil2</code>	Reserved for use by the system administrator or site-specific applications. This field is retained on reboot.
<code>tutil0</code>	Reserved for use by Volume Manager commands and is cleared on reboot.
<code>tutil1</code>	Reserved for use by high-level utilities such as the graphical user interface. This field is cleared on reboot.
<code>tutil2</code>	Reserved for use by the system administrator or site-specific applications. This field is cleared on reboot.

Refer to the following command:

```
# vxedit set comment="my plex" tutil2="u" user="admin" \
```

vol01-02

The `vxedit` command is used to modify attributes as follows:

- set the comment field (identifying what the plex is used for) to my plex
- set `tutil2` to `u` to indicate that the subdisk is in use
- change the user ID to `admin`

To prevent a particular plex from being associated with a volume, set the `putil0` field to a non-null string, as shown in the following command:

```
# vxedit set putil0="DO-NOT-USE" vol01-02
```

Changing Plex Status: Detaching and Attaching Plexes

Once a volume has been created and placed online (ENABLED), Volume Manager can temporarily disconnect plexes from the volume. This is useful, for example, when the hardware on which the plex resides needs repair or when a volume has been left unstartable and a source plex for the volume revive must be chosen manually.

Resolving a disk or system failure includes taking a volume offline and attaching and detaching its plexes. The two commands used to accomplish disk failure resolution are the `vxmend` command and the `vxplex` command.

To take a plex OFFLINE so that repair or maintenance can be performed on the physical disk containing subdisks of that plex, use the following command:

```
# vxmend off plex_name ..
```

If a disk has a head crash, put all plexes that have associated subdisks on the affected disk OFFLINE. For example, if plexes `vol01-02` and `vol02-02` had subdisks on a drive to be repaired, use the following command:

```
# vxmend off vol01-02 vol02-02
```

This command places `vol01-02` and `vol02-02` in the OFFLINE state, and they remain in that state until changed.

Detaching Plexes

To temporarily detach one plex in a mirrored volume, use the following command:

```
# vxplex det plex_name
```

For example, to temporarily detach a plex named `vol01-02` and place it in maintenance mode, use the following command:

```
# vxplex det vol01-02
```

This command temporarily detaches the plex, but maintains the association between the plex and its volume. However, the plex is not used for I/O. A plex detached with the preceding command is recovered

at system reboot. The plex state is set to STALE, so that if a `vxvol start` command is run on the appropriate volume (for example, on system reboot), the contents of the plex is recovered and made ACTIVE.

When the plex is ready to return as an active part of its volume, follow this procedure:

- If the volume is not ENABLED, start it with the following command:

```
# vxvol start volume_name
```

If it is unstartable, set one of the plexes to CLEAN using the following command:

```
# vxmend fix clean plex_name
```

and then start the volume.

- If the plex does not yet have a *kernel state* of ENABLED, use the following command:

```
# vxplex att volume_name plex_name ...
```

As with returning an OFFLINE plex to ACTIVE, this command recovers the contents of the plex(es), then sets the plex state to ACTIVE.

Attaching Plexes

When the disk has been repaired or replaced and is again ready for use, the plexes must be put back online (plex state set to ACTIVE). To set the plexes to ACTIVE, use one of the following commands.

- If the volume is currently ENABLED, use the following command:

```
# vxplex att volume_name plex_name ...
```

For example, for a plex named `vol01-02` on a volume named `vol01`, use the following command:

```
# vxplex att vol01 vol01-02
```

This command starts to recover the contents of the plex and, after the revive is complete, sets the plex utility state to ACTIVE.

- If the volume is not in use (not ENABLED), use the following command:

```
# vxmend on plex_name
```

For example, for a plex named `vol01-02`, use the following command:

```
# vxmend on vol01-02
```

Changing Plex Status: Detaching and Attaching Plexes

In this case, the state of vol01-02 is set to STALE. When the volume is next started, the data on the plex is revived from the other plex, and incorporated into the volume with its state set to ACTIVE.

To manually change the state of a plex, see “Recovering a Volume”. See the `vxmake(1M)` and `vxmend(1M)` manual pages for more information about these commands.

Moving Plexes

Moving a plex copies the data content from the original plex onto a new plex. To move data from one plex to another, use the following command:

```
# vxplex mv original_plex new_plex
```

For a move task to be successful, the following criteria must be met:

- The old plex must be an active part of an active (ENABLED) volume.
- The new plex must be at least the same size or larger than the old plex.
- The new plex must not be associated with another volume.

The size of the plex has several implications:

- If the new plex is smaller or more sparse than the original plex, an incomplete copy is made of the data on the original plex. If an incomplete copy is desired, use the `-o force` option.
- If the new plex is longer or less sparse than the original plex, the data that exists on the original plex is copied onto the new plex. Any area that is not on the original plex, but is represented on the new plex, is filled from other complete plex(es) associated with the same volume.
- If the new plex is longer than the volume itself, then the remaining area of the new plex above the size of the volume is not initialized and remains unused.

Copying Plexes

This task copies the contents of a volume onto a specified plex. The volume to be copied cannot be enabled. The plex cannot be associated with any other volume. To copy a plex, use the following command:

```
# vxplex cp volume_name new_plex
```

After the copy task is complete, *new_plex* is not associated with the specified volume *volume_name*. The plex contains a complete copy of the volume data. The plex that is being copied should be the same size or larger than the volume. If the plex being copied is larger than the volume, an incomplete copy of the data results. For the same reason, *new_plex* should not be sparse.

Creating Subdisks

You can use the `vxmake` command to create Volume Manager objects, such as subdisks. To create a subdisk, specify the following criteria:

- name of the subdisk
- length of the subdisk
- starting point (offset) of the subdisk within the disk
- disk media name

To create a subdisk, use the following command:

```
# vxmake sd subdisk_name disk,offset,len
```

For example, to create a subdisk named `disk02-01` that starts at the beginning of disk `disk02` and has a length of 8000 sectors, use the following command:

```
# vxmake sd disk02-01 disk02,0,8000
```

By default, Volume Manager commands take sizes in sectors. Adding a suffix (such as `k`, `m`, or `g`) changes the unit of size.

If you intend to use the new subdisk to build a volume, you must associate the subdisk with a plex (see “Associating Subdisks”). Subdisks for all plex layouts (concatenated, striped, RAID-5) are created the same way.

Removing Subdisks

To remove a subdisk, use the following command:

```
# vxedit rm subdisk_name
```

For example, to remove a subdisk named disk02-01, use the following command:

```
# vxedit rm disk02-01
```

Moving Subdisks

Moving a subdisk copies the disk space contents of a subdisk onto another subdisk. If the subdisk being moved is associated with a plex, then the data stored on the original subdisk is copied to the new subdisk. The old subdisk is dissociated from the plex, and the new subdisk is associated with the plex. The association is at the same offset within the plex as the source subdisk. To move a subdisk, use the following command:

```
# vxsd mv old_subdisk_name new_subdisk_name
```

For the subdisk move task to work correctly, the following conditions must be met:

- The subdisks involved must be the same size.
- The subdisk being moved must be part of an active plex on an active (ENABLED) volume.
- The new subdisk must not be associated with any other plex.

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

The following message example 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, move a relocated subdisk back to the original disk. For example, to move the relocated subdisk disk05-01 back to disk02, use the following command:

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

NOTE

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

Moving Hot-Relocate Subdisks Back to a Disk

NOTE

You may need an additional license to use this feature.

You can move hot-relocated subdisks back to the disk where they originally resided after the disk is replaced due to a disk failure. To move hot-relocated subdisks, use the following procedure:

- Step 1.** Select menu item 14 (Unrelocate subdisks back to a disk) from the vxdiskadm main menu.
- Step 2.** Enter the name of the disk where the hot-relocated subdisks originally resided at the following prompt:

Enter the original disk name [<disk>,list,q,?]

If no hot-relocated subdisks reside in the system, the vxdiskadm program returns the following message:

```
Currently there are no hot-relocated disks  
hit RETURN to continue
```

- Step 3.** Move the subdisks to a different disk from the original disk by entering **y** at the following prompt; otherwise, enter **n** or press Return:

```
Unrelocate to a new disk [y,n,q,?] (default: n)
```

- Step 4.** If moving subdisks to the original offsets is not possible, use the “force option” to unrelocate the subdisks to the specified disk (but not necessarily to the exact original offsets). Enter **y** at the following prompt; otherwise, enter **n** or press Return:

```
Use -f option to unrelocate the subdisks if moving to the  
exact  
offset fails? [y,n,q,?] (default: n)
```

- Step 5.** The following output is displayed. To continue the operation, enter **y** or press Return at the following prompt; otherwise, enter **n** to end the operation:

```
Requested operation is to move all the subdisks which  
were  
hot-relocated from disk10 back to disk10 of disk group  
rootdg.  
Continue with operation? [y,n,q,?] (default: y)
```

- Step 6.** When complete, the operation displays the following message:

```
Unrelocate to disk disk10 is complete.
```

Splitting Subdisks

Splitting a subdisk divides an existing subdisk into two subdisks. To split a subdisk, use the following command:

```
# vxsd -s size split subdisk_name newsd1 newsd2
```

where:

- `subdisk_name` is the name of the original subdisk
- `newsd1` is the name of the first of the two subdisks to be created
- `newsd2` is the name of the second subdisk to be created

The `-s` option is required to specify the size of the *first* of the two subdisks to be created. The second subdisk occupies the remaining space used by the original subdisk.

If the original subdisk is associated with a plex before the task, upon completion of the split, both of the resulting subdisks are associated with the same plex.

To split the original subdisk into more than two subdisks, repeat the previous command as many times as necessary on the resulting subdisks.

Joining Subdisks

Joining subdisks combines two or more existing subdisks into one subdisk. To join subdisks, the subdisks must be contiguous on the same disk. If the selected subdisks are associated, they must be associated with the same plex, and be contiguous in that plex. To join a subdisk, use the following command:

```
# vxsd join subdisk1 subdisk2 new_subdisk
```

Associating Subdisks

Associating a subdisk with a plex places the amount of disk space defined by the subdisk at a specific offset within the plex. The entire area that the subdisk fills must not be occupied by any portion of another subdisk. There are several ways that subdisks can be associated with plexes, depending on the overall state of the configuration.

If you have already created all the subdisks needed for a particular plex, to associate subdisks at plex creation, use the following command:

```
# vxmake plex plex_name sd=subdisk_name,...
```

For example, to create the plex `home-1` and associates subdisks `disk02-01`, `disk02-00`, and `disk02-02` with plex `home-1`, use the following command:

```
# vxmake plex home-1 sd=disk02-01,disk02-00,disk02-02
```

Subdisks are associated in order starting at offset 0. If you use this type of command, you do not have to specify the multiple commands needed to create the plex and then associate each of the subdisks with that plex. In this example, the subdisks are associated to the plex in the order they are listed (after `sd=`). The disk space defined as `disk02-01` is first, `disk02-00` is second, and `disk02-02` is third.

This method of associating subdisks is convenient during initial configuration. Subdisks can also be associated with a plex that already exists. To associate one or more subdisks with an existing plex, use the following command:

```
# vxsd assoc plex_name sd_name [sd_name2 sd_name3 ...]
```

For example, to associate subdisks named `disk02-01`, `disk02-00`, and `disk02-02` with a plex named `home-1`, use the following command:

```
# vxsd assoc home-1 disk02-01 disk02-00 disk02-01
```

If the plex is not empty, the new subdisks are added after any subdisks that are already associated with the plex, unless the `-l` option is specified with the command. The `-l` option associates subdisks at a specific offset within the plex.

The `-l` option is needed when you create a sparse plex (that is, a plex with a gap between its subdisks) for a particular volume, and want to make this plex complete. To make the plex complete, create a subdisk of

a size that fits the hole in the sparse plex exactly. Then, to associate the subdisk with the plex by specifying the offset of the beginning of the hole in the plex, use the following command:

```
# vxsd -l offset assoc sparse_plex_name  
exact_size_subdisk
```

NOTE

The subdisk must be exactly the right size because Volume Manager does not allow for the space defined by two subdisks to overlap within a single plex.

For striped subdisks, to specify a column number and column offset for the subdisk, use the following command:

```
# vxsd -l column_#/offset assoc plex_name sd_name ...
```

If only one number is specified with the `-l` option for striped plexes, the number is interpreted as a column number and the subdisk is associated at the end of the column.

Associating Log Subdisks

Log subdisks are defined for and added to a plex that is to become part of a volume using Dirty Region Logging. Dirty Region Logging is enabled for a volume when the volume is mirrored and has at least one log subdisk.

For a description of Dirty Region Logging, see “Dirty Region Logging”, and “Dirty Region Logging and Cluster Environments”. Log subdisks are ignored as far as the usual plex policies are concerned, and are only used to hold the dirty region log.

NOTE

Only one log subdisk can be associated with a plex. Because this log subdisk is frequently written, care should be taken to position it on a disk that is not heavily used. Placing a log subdisk on a heavily-used disk can degrade system performance.

To add a log subdisk to an existing plex, use the following command:

```
# vxsd aslog plex subdisk
```

Volume Tasks

Associating Subdisks

where *subdisk* is the name to be used as a log subdisk. The plex must be associated with a mirrored volume before DRL takes effect.

For example, to associate a subdisk named `disk02-01` with a plex named `vol01-02` (which is already associated with volume `vol01`), use the following command:

```
# vxsd aslog vol01-02 disk02-01
```

NOTE

You can also add a log subdisk to an existing volume with the following command: `# vxassist addlog volume_name disk`

This command automatically creates a log subdisk within a log plex for the specified volume.

Dissociating Subdisks

To break an established connection between a subdisk and the plex to which it belongs, the subdisk is *dissociated* from the plex. A subdisk is dissociated when the subdisk is removed or used in another plex. To dissociate a subdisk, use the following command:

```
# vxsd dis subdisk_name
```

For example, to dissociate a subdisk named disk02-01 from the plex with which it is currently associated, use the following command:

```
# vxsd dis disk02-01
```

NOTE

You can also remove subdisks with the following command: **# vxsd -orm dis subdisk_name**

Changing Subdisk Attributes

CAUTION

Change subdisk attributes with extreme care, and only if necessary.

The `vxedit` command changes attributes of subdisks to other Volume Manager objects. To change information relating to a subdisk, use the following command:

```
# vxedit set field=value ... subdisk_name
```

For example, to change the comment field of a subdisk named `disk02-01`, use the following command:

```
# vxedit set comment="new_comment" disk02-01
```

Subdisk fields that can be changed using the `vxedit` command are:

- name
- `putil[n]` fields
- `tutil[n]` fields
- `len` (only if the subdisk is dissociated)
- comment

NOTE

Entering data in the `putil0` field prevents the subdisk from being used as part of a plex, if it is not already part of a plex.

Performing Online Backup

NOTE

You may need an additional license to use this feature.

Volume Manager provides snapshot backups of volume devices. This is done through `vxassist` and other commands. There are various procedures for doing backups, depending upon the requirements for integrity of the volume contents. These procedures have the same starting requirement: a plex that is large enough to store the complete contents of the volume. The plex can be larger than necessary, but if a plex that is too small is used, an incomplete copy results.

The recommended approach to volume backup is by using the `vxassist` command which is easy to use. The `vxassist snapstart`, `snapwait`, and `snapshot` tasks provide a way to do online backup of volumes with minimal disruption to users.

The `vxassist snapshot` procedure consists of two steps:

Step 1. Running `vxassist snapstart` to create a snapshot mirror

Step 2. Running `vxassist snapshot` to create a snapshot volume

You can use the `vxassist` command to create a snapshot of a RAID-5 volume by using the recommended approach to volume backup described in this section.

The `vxassist snapstart` step creates a write-only backup plex which gets attached to and synchronized with the volume. When synchronized with the volume, the backup plex is ready to be used as a snapshot mirror. The end of the update procedure is indicated by the new snapshot mirror changing its state to `SNAPDONE`. This change can be tracked by the `vxassist snapwait` task, which waits until at least one of the mirrors changes its state to `SNAPDONE`. If the attach process fails, the snapshot mirror is removed and its space is released.

Once the snapshot mirror is synchronized, it continues being updated until it is detached. You can then select a convenient time at which to create a snapshot volume as an image of the existing volume. You can

also ask users to refrain from using the system during the brief time required to perform the snapshot (typically less than a minute). The amount of time involved in creating the snapshot mirror is long in contrast to the brief amount of time that it takes to create the snapshot volume.

The online backup procedure is completed by running the `vxassist snapshot` command on a volume with a SNAPDONE mirror. This task detaches the finished snapshot (which becomes a normal mirror), creates a new normal volume and attaches the snapshot mirror to the snapshot volume. The snapshot then becomes a normal, functioning mirror and the state of the snapshot is set to ACTIVE.

If the snapshot procedure is interrupted, the snapshot mirror is automatically removed when the volume is started.

Use the following steps to perform a complete `vxassist` backup:

Step 1. Create a snapshot mirror for a volume with this command:

```
# vxassist snapstart volume_name
```

Step 2. When the `snapstart` step is complete and the mirror is in a SNAPDONE state, choose a convenient time to complete the snapshot task. Inform users of the upcoming snapshot and ask them to save files and refrain from using the system briefly during that time.

Create a snapshot volume that reflects the original volume with this command:

```
# vxassist snapshot volume_name temp_volume_name
```

Step 3. Use `fsck` (or some utility appropriate for the application running on the volume) to clean the temporary volume's contents. For example, you can use this command:

```
# fsck -y /dev/vx/rdisk/temp_volume_name
```

Step 4. Copy the temporary volume to tape, or to some other appropriate backup media.

Step 5. Remove the new volume with this command:

```
# vxedit -rf rm temp_volume_name
```


FastResync (Fast Mirror Resynchronization)

NOTE

You may need an additional license to use this feature.

The FastResync feature (also called Fast Mirror Resynchronization, which is abbreviated as FMR) performs quick and efficient resynchronization of stale mirrors by increasing the efficiency of the VxVM snapshot mechanism to better support operations such as backup and decision support.

Enabling FMR

When a new volume is created with `vxassist`, an attribute can be specified to turn FMR on or off. Both keywords `fmr` and `fastresync` can be used as attributes to specify that FMR will be used (or not) on a volume.

To create a volume with FMR enabled, use the `vxassist make` command as follows:

```
# vxassist make volume_name size fmr=on
```

The default is for FMR to be off, but you can change the default in the `vxassist` default file.

The FMR functionality can also be turned ON or OFF with the `vxvol` command. To use FMR, FMR must be enabled when the snapshot is taken, and FMR must remain enabled until after the snapback is completed. Turning FMR off will free all of the tracking maps for the specified volume. All subsequent reattaches will not use the FMR facility, but do a full resynchronization of the volume. This occurs even if FMR is later turned on.

To turn FMR on, use the following command:

```
# vxvol set fmr=on volume_name
```

To turn FMR off, use the following command:

```
# vxvol set fmr=off volume_name
```

Merging a Snapshot Volume

A snapshot copy of a volume can be merged back with the original volume. The snapshot plex is detached from the snapshot volume and

attached to the original volume. The snapshot volume is removed. This task resynchronizes the data in the volume so that the plexes are consistent.

To merge a snapshot with its original volume, use the following command:

```
# vxassist snapback replica-volume
```

where *replica-volume* is the snapshot copy of the volume.

By default, the data in the original plex is used for the merged volume. To use the data copy from the replica volume instead, use the following command:

```
# vxassist -o resyncfromreplica snapback replica-volume
```

Dissociating a Snapshot Volume

The link between a snapshot and its original volume can be permanently broken so that the snapshot volume becomes an independent volume.

To dissociate a snapshot from its original volume, use the following command:

```
# vxassist snapclear replica-volume
```

where *replica-volume* is the snapshot copy of the volume.

Displaying Snapshot Volume Information

The `vxassist snapprint` command displays the associations between the original volumes and their respective replicas (snapshot copies).

The syntax for the `snapprint` option is:

```
# vxassist snapprint [volume-name]
```

The output from this command displays the following:

V NAME	USETYPE	LENGTH	RP NAME	VOLUME	LENGTH
RRPLEXID					
v vol	fsgen	2048	rp vol-05	SNAP1-vol	3040
vol-04					rp

If a volume is specified it will display either output for that volume or an error message if no FMR maps are enabled for that volume. Otherwise, it will display information for all volumes in the disk group.

Mirroring Volumes on a VM Disk

Mirroring the volumes on a VM disk gives you one or more copies of your volumes in another disk location. By creating mirror copies of your volumes, you protect your system against loss of data in case of a disk failure. You can use this task on your root disk to make a second copy of the boot information available on an alternate disk. This allows you to boot your system even if your root disk is corrupted.

NOTE

This task only mirrors concatenated volumes. Volumes that are already mirrored or that contain subdisks that reside on multiple disks are ignored.

To mirror volumes on a disk, make sure that the target disk has an equal or greater amount of space as the originating disk and then do the following:

- Step 1.** Select menu item 6 (Mirror volumes on a disk) from the vxdiskadm main menu.
- Step 2.** At the following prompt, enter the disk name of the disk that you wish to mirror:

```
Mirror volumes on a disk
Menu: VolumeManager/Disk/Mirror
```

This operation can be used to mirror volumes on a disk. These

volumes can be mirrored onto another disk or onto any available disk space. Volumes will not be mirrored if they are

already mirrored. Also, volumes that are comprised of more than

one subdisk will not be mirrored.

Mirroring volumes from the boot disk will produce a disk that

can be used as an alternate boot disk.

Enter disk name [<disk>,list,q,?] disk02

- Step 3.** At the following prompt, enter the target disk name (this disk must be the same size or larger than the originating disk):

You can choose to mirror volumes from disk disk02 onto any available disk space, or you can choose to mirror onto a specific disk. To mirror to a specific disk, select the name of that disk. To mirror to any available disk space, select "any".
Enter destination disk [<disk>,list,q,?] (default: any)
disk01

NOTE

Be sure to always specify the destination disk when you are creating an alternate root disk. Otherwise, the Volume Manager selects a disk to be the alternate root disk. However, your system may not be able to boot from that disk.

Step 4. At the following prompt, press Return to make the mirror:

The requested operation is to mirror all volumes on disk disk02 in disk group rootdg onto available disk space on disk disk01.
NOTE: This operation can take a long time to complete.
Continue with operation? [y,n,q,?] (default: y)
vxdiskadm displays the status of the mirroring operation:
Mirror volume voltest-bk00 ...

Mirroring of disk disk01 is complete.

Step 5. At the following prompt, indicate whether you want to mirror volumes on another disk (**y**) or return to the vxdiskadm main menu (**n**):

Mirror volumes on another disk? [y,n,q,?] (default: n)

Moving Volumes from a VM Disk

Before you disable or remove a disk, you may want to move the data from that disk to other disks on the system. To do this, make sure that the target disks have sufficient space, then do the following:

Step 1. Select menu item 7 (Move volumes from a disk) from the vxdiskadm main menu.

Step 2. At the following prompt, enter the disk name of the disk whose volumes you wish to move:

```
Move volumes from a disk
Menu: VolumeManager/Disk/Evacuate
```

Use this menu operation to move any volumes that are using a disk onto other disks. Use this menu immediately prior to removing a disk, either permanently or for replacement. You can specify a list of disks to move volumes onto, or you can move the volumes to any available disk space in the same disk group.

NOTE: Simply moving volumes off of a disk, without also removing the disk, does not prevent volumes from being moved onto the disk by future operations. For example, using two consecutive move operations may move volumes from the second disk to the first.

```
Enter disk name [<disk>,list,q,?] disk01
```

After the following display, you can optionally specify a list of disks to which the volume(s) should be moved.

You can now specify a list of disks to move onto.

Specify a list

of disk media names (e.g., disk01) all on one line separated by

blanks. If you do not enter any disk media names, then the

volumes will be moved to any available space in the disk group.

At the following prompt, press **Return** to move the volumes:

```
Requested operation is to move all volumes from disk
disk01 in
group rootdg.
```

NOTE: This operation can take a long time to complete.

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

As the volumes are moved from the disk, vxdiskadm displays the status of the operation:

```
Move volume voltest ...
```

```
Move volume voltest-bk00 ...
```

When the volumes have all been moved, vxdiskadm displays the following success message:
Evacuation of disk disk01 is complete.

Step 3. At the following prompt, indicate whether you want to move volumes from another disk (y) or return to the vxdiskadm main menu (n):

Move volumes from another disk? [y,n,q,?] (default: n)

7 **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”
- “Disks in VxVM Clusters”
- “Dirty Region Logging and Cluster Environments”
- “Dynamic Multipathing (DMP)”
- “FastResync (Fast Mirror Resynchronization)”
- “Upgrading Volume Manager Cluster Functionality”
- “Cluster-related Volume Manager Utilities and Daemons”
- “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 can be 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—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—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.

Disks in a shared disk group are accessible from all nodes in a cluster, allowing applications on multiple cluster nodes to simultaneously access the same disk. A volume in a shared disk group can be simultaneously accessed by more than one node in the cluster, subject to licensing and disk group activation mode descriptions.

A shared disk group must be activated on a node in order for the volumes in the disk group to become accessible for application I/O from that node. The ability of applications to read or write to volumes is dictated by the activation mode of the disk group. Valid activation modes for a shared disk group are *exclusive-write*, *shared-write*, *read-only*, *shared-read* and *off* (or inactive, as shown in Table 7-1, “Activation Modes for Shared Disk Group.”

Table 7-1 Activation Modes for Shared Disk Group

Exclusive write	The node has exclusive write access to the disk group. No other node can activate the dg for write access.
Shared write	The node has write access to the disk group.
Read only	The node has read access to the disk group and denies write access for all other nodes in the cluster. The node has no write access to the disk group. Attempts to activate a disk group for either of the write modes on other nodes will fail.
Shared read	The node has read access to the disk group. The node has no write access to the disk group, however other nodes can obtain write access.
Off	The node has neither read nor write access to the disk group. Query operations on the disk group are permitted.

Special uses of clusters, such as high availability (HA) applications and off-host backup, can utilize disk group activation to explicitly control volume I/O capability from different nodes in the cluster. Use of activation modes is described in “Disk Group Activation”.

Notes:

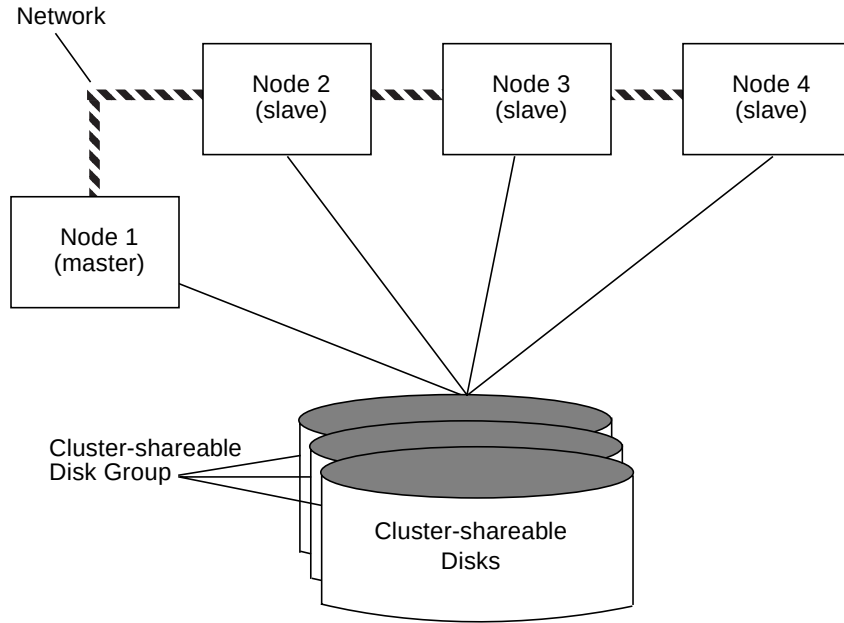
- The striped mirror volumes, task monitor, and online relayout features are not supported in clusters.
- The Volume Manager cluster feature does not currently support RAID-5 volumes in cluster-shareable disk groups. RAID-5 volumes can, however, be used in private disk groups attached to specific nodes of a cluster.
- If a disk group that contains unsupported objects is imported as shared, deport the disk group. Reorganize the contained volumes into supported layouts, and then reimport it as shared.

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.

The following figure 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 Figure 7-1, Example of a 4-Node Cluster,, Node 1 is the master node and Node 2, Node 3, and Node 4 are the slave nodes.

Figure 7-1 **Example of a 4-Node Cluster**



The system administrator designates a disk group as cluster-shareable using the `vxdg` utility (see “`vxdg` Utility” for more 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 place 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 can have simultaneous read and write access to any cluster-shareable disk group depending on the activation mode. 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. 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.

NOTE

If MC/ServiceGuard is chosen as your cluster manager, no additional

configuration of VxVM is required, apart from the cluster configuration requirements of MC/ServiceGuard.

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. The cluster manager for each node monitors other nodes in the cluster and calls the `vxclustd` cluster reconfiguration utility when there is a change in cluster membership. The `vxclustd` utility coordinates cluster reconfigurations and provides communication between VxVM and the cluster manager. The cluster manager and the `vxclustd` utility 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 more information on cluster reconfiguration, see “`vxclustd` Daemon”.

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.

NOTE

Volume reconfiguration is initiated and coordinated by the master node, so the system administrator must run the utilities that request changes to Volume Manager objects on the master node.

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

The utility on the master 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. The `vxconfigd` daemon on the master node then sends messages with the details of the changes to the `vxconfigd` daemons on all other nodes in the cluster. The `vxconfigd` daemons on each of the slave nodes then perform their own checking. For example, a slave 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` daemons on all nodes agree that the proposed change is reasonable, each `vxconfigd` daemon 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 a `vxconfigd` daemon on any node goes away during a reconfiguration process, all nodes are notified and the operation fails. If any node leaves the cluster, the operation fails 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 does not receive it, neither does any other slave.

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

When an error occurs (such as when a check on a slave fails or a node leaves the cluster), the error is returned to the utility and a message is issued to the console on the master 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 is 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 can take a long time if, for example, a long-running transaction is active.

When the VxVM shutdown procedure is invoked, the procedure 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 succeeds, the node has left the cluster. It is not possible to access the shared volumes until the node joins the cluster again.

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

Node Abort

If a node does not leave cleanly, this is because the host crashed. 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 fails, no node can access it and the nodes can agree to detach the disk. If the disk does not fail, but rather the access paths from some of the nodes fail, the nodes cannot agree on the status of the disk. A policy must exist to resolve this type of discrepancy.

Disk Detach Policies

In order to address the above discrepancy, the following policies (set for a diskgroup) are provided. These can be set by using the `vxedit(1M)` command.

Under the global connectivity policy for shared disk group(s), the detach occurs cluster-wide (globally), if any node in the cluster reports a disk(s) failure.

Under local connectivity policy, in the event of disk(s) failing, the failures are confined to the particular node(s) which saw the failure. Note that an attempt is made to communicate with all nodes in the cluster to ascertain the disk(s) usability. If all nodes report a problem with the disk(s), a cluster-wide detach occurs. This is the default policy.

Disk Group Activation

Disk group activation controls volume I/O capability from different nodes in the cluster. It is not possible to activate a diskgroup on a given node if it is activated in a conflicting mode on another node in the cluster. Table 7-2, Allowed and Conflicting Activation Modes, summarizes the allowed and conflicting activation modes for shared disk groups, as follows:

Table 7-2

Allowed and Conflicting Activation Modes

Disk group activated in the cluster as....	Attempt to activate disk group on another node as....
--	---

Table 7-2 **Allowed and Conflicting Activation Modes**

	Exclusive write	Shared write	Read only	Shared read
Exclusive write	Fail	Fail	Fail	Succeed
Shared write	Fail	Succeed	Fail	Succeed
Read only	Fail	Fail	Succeed	Succeed
Shared read	Succeed	Succeed	Succeed	Succeed

Shared disk groups can be automatically activated in any mode during disk group creation or during manual or auto-import. To control auto-activation of shared disk groups, the defaults file `/etc/default/vxdg` must be created.

The defaults file `/etc/default/vxdg` must contain the following lines:

```
default_activation_mode=activation-mode
enable_activation=true
```

where *activation-mode* is: *off*, *shared-write*, *shared-read*, *read-only*, or *exclusive-write*.

When a shared disk group is created or imported, it is activated in the specified mode. When a node joins the cluster, all shared disk groups accessible from the node are activated in the specified mode.

If the default file is edited when the `vxconfigd` utility is running, the `vxconfigd` utility must be restarted for the changes in the default file to take effect.

Notes:

- When enabling activation using the defaults file, it is recommended that the defaults file be identical on all nodes in the cluster. Otherwise, the results of activation are unpredictable.
- If the default activation mode is anything other than *off*, an activation following a cluster join, or a disk group creation or import may fail if another node in the cluster has activated the disk group in a conflicting mode.

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. For more information on DRL, see “Dirty Region Logging (DRL) Guidelines” and “Dirty Region Logging”.

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 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. A log region becomes clean again after the write to the mirror is complete. 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 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 in the non-clustered case, 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 the `vxplex -o rm dis` command) and then recreated (using the `vxassist addlog` command). The use of these two commands 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 considers the logs of the shared volumes to be invalid and conducts a full volume recovery. After this recovery completes, the Volume Manager uses 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 is probably too small to accommodate all the nodes in the cluster. The cluster feature therefore marks the log invalid and performs a full recovery anyway. Similarly, moving a DRL volume from a two-node cluster to a four-node cluster can result in too small a log size, which the cluster feature handles 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 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.

Dynamic Multipathing (DMP)

NOTE

You may need an additional license to use this feature.

In a clustered environment where Active/Passive type disk arrays are shared by multiple hosts, all hosts in the cluster should access the disk via the same physical path. If a disk from an Active/Passive type shared disk array is accessed via multiple paths simultaneously, it could lead to severe degradation of I/O performance. This requires path failover on a host to be a cluster coordinated activity for an Active/Passive type disk array.

For Active/Active type disk arrays, any disk can be simultaneously accessed through all available physical paths to it. Therefore, in a clustered environment all hosts do not need to access a disk, via the same physical path.

Disabling and Enabling Controllers

System maintenance tasks can be carried out using the `vxddmpadm` utility. The `disable` utility prevents DMP from issuing I/Os through the specified controller. To disable I/Os through the host disk controller `c2`, use the following command:

```
# vxddmpadm disable ctrl=c2
```

Previously disabled controllers can now be enabled to accept future I/Os. This operation succeeds only if the controller is accessible to the host and I/O can be performed on this controller. To enable the controller that was previously disabled, use the following command:

```
# vxddmpadm enable ctrl=c2
```

The `vxddmpadm` command provides information like the list of paths connected to the disk array and the list of DMP nodes on the system. It also lists the controllers and enclosures connected to the system. To list all paths controlled by the DMP node `c2t1d0`, use the following command:

```
# vxddmpadm getsubpaths dmpnodename=c2t1d0
```

You can also obtain all paths connected to a particular controller, for example c2, using the following command:

```
# vxdmpadm getsubpaths ctlr=c2
```

To get the node that controls a path to a disk array, use the following command:

```
# vxdmpadm getdmpnode nodename=c3t2d1
```

Assigning a User Friendly Name

You can assign a new name to a disk array using the following command:

```
# vxdmpadm setattr enclosure nike0 name=VMGRP_1
```

Assigning a logical name to the enclosure helps to easily identify the enclosure. The above command assigns the name VMGRP_1 to the enclosure nike0.

Display DMP Nodes

The `vxdmpadm getdmpnode` command displays the DMP node that controls a particular physical path. The physical path can be specified as the `nodename` attribute to this command. The node name must be a valid path listed in the `/dev/rdisk` directory.

The following command displays the DMP nodes to the enclosure `nike0`:

```
# vxdmpadm getdmpnode enclosure=nike0
```

Display Subpaths

The `vxdmpadm getsubpaths` command displays the paths controlled by the specific DMP node. The `dmpnodename` specified must be a valid node in the `/dev/vx/rdmp` directory. The `vxdmpadm getsubpaths` command also obtains all paths through a particular host disk controller.

For example, the following command lists all paths controlled by the DMP node `c2t1d0`:

```
# vxdmpadm getsubpaths dmpnodename=c2t1d0
```

The following command obtains all paths through a the host disk controller `c2`:

```
# vxdmpadm getsubpaths ctlr=c2
```


List Controllers

The `vxddmpadm listctlr` command lists specified disk controllers connected to the host. This command also lists the controllers on a specified enclosure or a particular type of enclosure, as follows:

```
# vxddmpadm listctlr all
# vxddmpadm listctlr enclosure=nike0 type=NIKE
```

List Enclosure

The `list enclosure` command displays all attributes of the enclosure(s) listed on the command line, as follows:

```
# vxddmpadm listenclosure nike0 fc1010d0
```

DMP Restore Daemon

The DMP restore daemon periodically analyzes the health of the paths. The restore daemon is started at the system startup time with default attributes. The interval of polling can be set by using the following command:

```
# vxddmpadm start restore interval=400
```

Depending on the policy specified restore checks the health of paths that were previously disabled or checks all paths. The paths that are back online are revived and the inaccessible paths are disabled. Set the policy using the following command:

```
# vxddmpadm start restore policy=check_all
```

To stop the restore daemon enter the command:

```
# vxddmpadm stop restore
```

The intervals of restore daemons running and the numbers of error daemons are obtained using the following commands:

```
# vxddmpadm stat restored
# vxddmpadm stat errord
```

FastResync (Fast Mirror Resynchronization)

NOTE

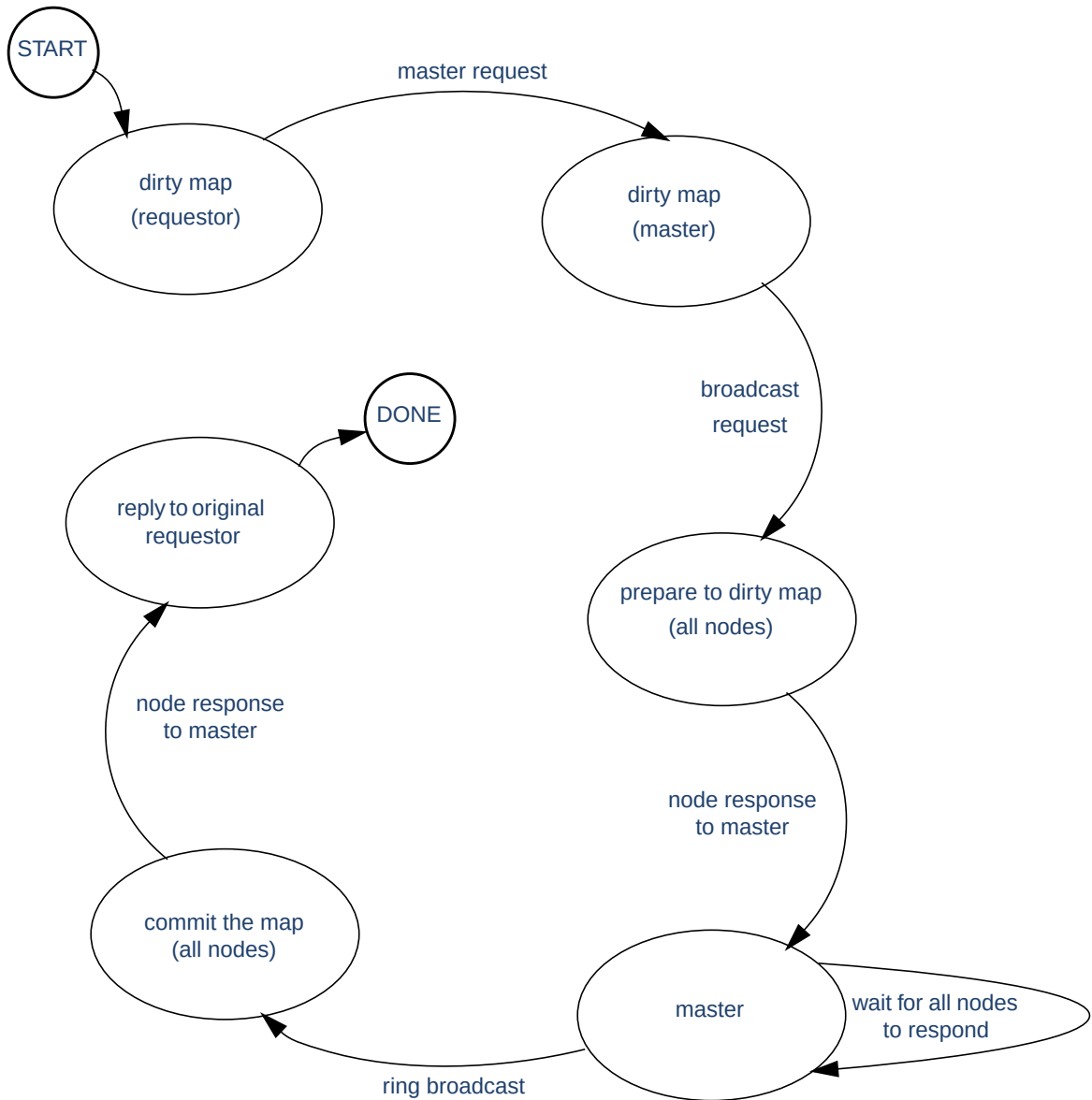
You may need an additional license to use this feature.

The FastResync feature (also called Fast Mirror Resynchronization, which is abbreviated as FMR) is supported for shared volumes. The update maps (FMR maps) are distributed across in the cluster. The existence of non-persistent maps is less moot in a cluster environment because only one node in the cluster must be up for the FMR maps to be available. A single node crash does not cause the loss of FMR maps.

Since the region size must be the same on all nodes in a cluster for a shared volume, the value of the `vol_fmr_logsz` tunable on the master overrides the tunable values on the slaves, if the slave values are different. Because the value of a shared volume can change, the `vol_fmr_logsz` tunable value is retained for the life of the volume or until FastResync is turned on for the volume.

The map updates are applied under the auspices of the master node. When the master node distributes updates to all nodes, all updates are applied either to all nodes or to none of the nodes. The master node orchestrates a two-phase commitment for applying any updates. See Figure 7-2, Bitmap Clusterization,.

Figure 7-2 **Bitmap Clusterization**



Upgrading Volume Manager Cluster Functionality

The rolling upgrade feature allows an administrator to upgrade the version of Volume Manager running in a cluster without shutting down the entire cluster. To install the new version of Volume Manager running on a cluster, the system administrator can pull out one node from the cluster, upgrade it and then join the node back into the cluster. This is done for each node in the cluster.

Every VxVM release, starting with Release 3.1, has a cluster protocol version number associated with it. This is different from the release number. The cluster protocol version is stored in the `/etc/vx/volboot` file. In a new installation of VxVM, the `volboot` file does not exist in the `/etc/vx` directory. The `vxctl init` command creates this file and sets the cluster protocol version to the highest supported version.

A new VxVM release supports two versions of cluster protocol. The lower version corresponds to the existing VxVM release. This has a fixed set of features and communication protocols. The higher version corresponds to a new release of VxVM which has a new set of these features. If the new release of VxVM does not have any functional or protocol changes, the cluster protocol version remains unchanged, for example, in case of bug fixes or minor changes. In this case, the `vxctl upgrade` command need not be executed.

During the rolling upgrade operation each node must be shut down and the VxVM release with the latest cluster protocol version must be installed. All nodes that have the new release of VxVM continue to use the lower level version. A slave node that has the new cluster protocol version installed tries to join the cluster. If the new cluster protocol version is not in use on the master node, it rejects the join and provides the current cluster protocol version to the slave node. The slave retries the join with the cluster protocol version provided by the master node. If the join fails at this point, the cluster protocol version on the master node is out of range of the protocol versions supported by the joining slave. In such a situation the system administrator must upgrade the cluster through each intermediate release of VxVM to reach the latest supported cluster protocol version.

All nodes are upgraded to the latest cluster protocol version and the new features are available.

Once all nodes have the new release installed, the `vxctl upgrade` command must be run on the Master node to switch to the higher cluster protocol version. See “vxctl Utility” for more information.

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:

- “vxclustd Daemon”
- “vxconfigd Daemon”
- “vxdctl Utility”
- “vxdg Utility”
- “vxdisk Utility”
- “vxdumpadm Utility”
- “vxrecover Utility”
- “vxstat Utility”

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, see their manual pages.

NOTE

Most Volume Manager commands require superuser privileges.

vxclustd Daemon

The vxclustd daemon is the Volume Manager cluster reconfiguration daemon. The vxclustd daemon provides communication between the cluster manager and VxVM and initiates cluster reconfigurations. Every node currently in the cluster runs an instance of the vxclustd daemon. Whenever cluster membership changes, the cluster manager notifies the vxclustd daemon, which then initiates a reconfiguration within VxVM.

The vxclustd daemon is started up by the cluster manager when the node initially attempts to join the cluster. The vxclustd daemon first registers with the cluster manager and obtains the following information from the cluster manager :

- cluster ID and cluster name
- node IDs and hostnames of all configured nodes
- IP addresses of the network interfaces through which the nodes communicate with each other

Registration also provides a callback mechanism for the cluster manager to notify the `vxclustd` daemon when cluster membership changes. After initializing kernel cluster variables, the `vxclustd` daemon waits for a callback from the cluster manager. When the `vxclustd` daemon obtains membership information from the cluster manager, it validates the membership change, and provides the new membership to the kernel. The reconfiguration process continues within the kernel and the `vxconfigd` daemon. This includes selection of a new master node if necessary, initiation of communication between `vxconfigd` daemons on the master and slave nodes, and a join protocol at the `vxconfigd` and kernel levels that validates VxVM objects and distributes VxVM configuration information across the cluster.

If reconfiguration completes successfully, the `vxclustd` daemon does not take any further action; it waits for the next membership change from the cluster manager. If reconfiguration within the kernel or within the `vxconfigd` daemon fails, the node must leave the cluster. The kernel fails I/Os in progress to shared disks, and stops access to shared disks and the `vxclustd` daemon. The `vxclustd` daemon invokes the cluster manager command to halt the cluster on this node.

When a clean node shutdown is performed, the `vxclustd` daemon waits until kernel cluster reconfiguration completes and then exits.

Notes:
• If MC/ServiceGuard is the cluster manager, it expects the <code>vxclustd</code> daemon registration to complete within a given timeout period. If registration times out, MC/ServiceGuard aborts cluster initialization and fails cluster startup. • When performing a clean node shutdown using MC/ServiceGuard's <code>cmhaltnode</code> command, VxVM kernel reconfiguration does not complete until all I/Os to shared disks have completed. This can take a long time, causing MC/ServiceGuard to time out.

vxconfigd Daemon

The `vxconfigd` daemon is the Volume Manager configuration daemon. The `vxconfigd` daemon maintains configurations of VxVM objects. The

`vxconfigd` daemon receives cluster-related instructions from the kernel. A separate copy of the `vxconfigd` daemon 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` daemon running on that particular node; utilities do not attempt to connect with `vxconfigd` daemons on other nodes. During startup of the cluster, kernel tells the `vxconfigd` daemon to begin cluster operation and tells it whether it is a master or slave node.

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

- cluster ID
- node IDs
- master node ID
- node's role
- network address of the `vxconfigd` on each node

On the master node, the `vxconfigd` daemon sets up the shared configuration (i.e., imports the shared disk groups) and informs the `vxclustd` daemon when it is ready for slaves to join.

On slave nodes, the kernel tells the `vxconfigd` daemon when the slave node can join the cluster. When the slave node joins the cluster, the `vxconfigd` daemon 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, the `vxconfigd` daemon notifies the kernel on all the other nodes. The master node then performs any necessary cleanup. If the master node leaves the cluster, the kernels choose a new master node and the `vxconfigd` daemons on all nodes are notified of the choice.

The `vxconfigd` daemon also participates in volume reconfiguration. See “Volume Reconfiguration” for information on the role of the `vxconfigd` daemon in volume reconfiguration.

`vxconfigd` Daemon Recovery

The Volume Manager `vxconfigd` daemon can be stopped and/or

restarted at any time. While the `vxconfigd` daemon is stopped, volume reconfigurations cannot take place and other nodes cannot join the cluster until the `vxconfigd` daemon 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 the `vxconfigd` daemon is stopped, different actions are taken depending on which node has a stopped daemon:

- If the `vxconfigd` daemon is stopped on the slave(s), the master takes no action. When the `vxconfigd` daemon is restarted on the slave, the slave's `vxconfigd` daemon 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` daemon has successfully rejoined to the master, it has very little information about the shared configuration and any attempts to display or modify the shared configuration can fail. In particular, if the shared disk groups are listed (using the `vxdg list` command), they are marked as disabled; when the rejoin completes successfully, they are marked as enabled.
- If the `vxconfigd` daemon is stopped on the master, the `vxconfigd` daemon on the slave(s) attempts to rejoin to the master periodically. This is not successful until the `vxconfigd` daemon is restarted on the master. In this case, the slave `vxconfigd` daemon information about the shared configuration has not been lost, so configuration displays are accurate.
- If the `vxconfigd` daemon is stopped on both the master and the slave(s), the slave does not display accurate configuration information until the `vxconfigd` daemon is restarted on both nodes and they reconnect.

When the `vxclustd` daemon notices that the `vxconfigd` daemon is stopped on a node, the `vxclustd` daemon restarts the `vxconfigd` daemon.

NOTE

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

vxconfigd daemon does not start.

vxctl Utility

When the vxctl utility is executed as “vxctlenable”, if DMP identifies a DISABLED primary path of a shared disk in an active/passive type disk array as physically accessible, this path is marked as ENABLED. However, I/Os continue to use the current path and are not routed through the path that has been marked ENABLED.

This is a deviation from single host behavior of this command where I/Os automatically failback to the primary path.

The vxctl utility manages some aspects of the vxconfigd volume configuration daemon. The -c option can be used to request cluster information. To use the vxctl utility to determine whether the vxconfigd daemon is enabled and/or running, use the following command:

```
# vxctl -c mode
```

Depending on the circumstances, this produces 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 the vxconfigd daemon is disabled, no cluster information is displayed.

The vxctl utility lists the cluster protocol version and cluster protocol range. After all the nodes in the cluster are updated with the new cluster protocol, update the entire cluster using the following command:

```
# vxctl upgrade
```

Check the existing cluster protocol version using the following command:

```
# vxctl protocolversion
Cluster running at protocol 10
```

Display the maximum and minimum cluster protocol version supported

by the current VxVM release using the following command:

```
# vxdctl protocolrange  
minprotoversion: 10, maxprotoversion: 20
```

The `vxdctl list` command displays the cluster protocol version running on a node. The `vxdctl list` command produces the following output:

```
Volboot file  
version: 3/1  
seqno: 0.19  
cluster protocol version: 20  
hostid: giga  
entries:
```

The `vxdctl support` command displays the maximum and minimum protocol version supported by the node and the current protocol version. The following is the output of the `vxdctl support` command:

```
Support information:  
  vold_vrsn: 11  
  dg_minimum: 60  
  dg_maximum: 70  
  kernel: 10  
  protocol_minimum: 10  
  protocol_maximum: 20  
  protocol_current: 20
```

vxdg Utility

The `vxdg` utility manages Volume Manager disk groups. The `vxdg` utility can be used to specify that a disk group is cluster-shareable. The `-s` option to the `vxdg` utility 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 using 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

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

Importing disk groups

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

```
# vx dg -s import diskgroup
```

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

Converting a disk group from shared to private

A disk group can be converted from shared to private by deporting it via `vx dg deport` and then importing it with the `vx dg import diskgroup` command.

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 are unable 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.

Force-importing a disk group or force-adding a disk

The `vx dg` command 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 using the following command:

```
# 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 does 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 using the following command:

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

Activating a shared disk group

A shared disk group can be activated using the following command:

```
# vxdg -g diskgroup set activation=mode
```

where *mode* is one of the following:

- exclusivewrite
- sharedwrite
- readonly
- sharedread
- off

Listing shared disk groups

vxdg can also be used to list shared disk groups. To display one line of information for each disk group, use the following command:

```
# 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.

To display one line of information for each shared disk group, use the following command:

```
# 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

To display information about one specific disk group, including whether it is shared or not, use the following command:

```
# vxdg list diskgroup
```

where *diskgroup* is the disk group name.

For example, 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
version:    70
activation: exclusive-write
detach-policy: local
copies:     nconfig=default nlog=default
config:     seqno=0.1976 permrlen=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 Utility

The `vxdisk` utility manages Volume Manager disks. To use the `vxdisk` utility to determine whether a disk is part of a cluster-shareable disk group, use the following command:

```
# vxdisk list accessname
```

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

The output from this command (for the device `c4t1d0`) is as follows:

```
Device:      c4t1d0
devicetag:   c4t1d0
type:        simple
hostid:      hpvm2
disk:        name=disk01 id=963616090.1034.hpvm2
timeout:     30
group:       name=rootdg id=963616065.1032.hpvm2
flags:       online ready private autoconfig autoimport
imported
pubpaths:    block=/dev/vx/dmp/c4t1d0
char=/dev/vx/rdmp/c4t1d0
version:     2.1
iosize:      min=1024 (bytes) max=64 (blocks)
public:      slice=0 offset=1152 len=8612361
private:     slice=0 offset=128 len=1024
update:      time=964035962 seqno=0.30
headers:     0 248
configs:     count=1 len=727
logs:        count=1 len=110
Defined regions:
  config  priv 000017-000247[000231]: copy=01
offset=000000 enabled
  config  priv 000249-000744[000496]: copy=01
offset=000231 enabled
  log     priv 000745-000854[000110]: copy=01
offset=000000 enabled
  lockrgn priv 000855-000919[000065]: part=00
offset=000000
Multipathing information:
numpaths:    2
c4t1d0  state=enabled   type=secondary
c5t3d0  state=enabled   type=primary
```

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`.

vxddmpadm Utility

The `vxddmpadm` utility is a command line administrative interface for the Dynamic Multipathing feature of VxVM. This utility can be used to list DMP database information and perform other system administration related tasks. For more information, see the manual page `vxddmpadm(1M)` and “Disabling and Enabling Controllers”.

Enabling and Disabling Controllers

Volume Manager does not allow enabling or disabling of controllers connected to a disk that is part of a shared VxVM disk group.

For example, consider a Galaxy disk array that is connected through controller `c0` to a host. This controller has a disk that is part of a shared disk group. In such a situation, following operations fail on that host:

```
# vxddmpadm disable ctrl=c0
# vxddmpadm enable ctrl=c0
```

The following error message is displayed:

```
vxvm: vxddmpadm: ERROR:
operation not supported for shared disk arrays.
```

DMP Restore Daemon

The DMP restore daemon does not automatically failback I/Os for a disk in an Active/Passive disk array if that disk is a part of a shared disk group.

When a restore daemon revives a DISABLED primary path to a shared disk in an Active/Passive disk array, DMP does not route the I/Os automatically through the primary path, but continues routing them through the secondary path. This behavior of the restore daemon is a deviation from that on a single host environment.

A shared disk here means a disk which is a part of a shared disk group.

In a clustered environment, failback of I/Os to the primary path happens only when the secondary path becomes physically inaccessible to the

host.

vxrecover Utility

The `vxrecover` utility recovers plexes and volumes after disk replacement.

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

NOTE

While the `vxrecover` utility is active, there may be some degradation in system performance.

vxstat Utility

The `vxstat` utility returns statistics for specified objects. In a cluster environment, the `vxstat` utility 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.

The caller can optionally specify a subset of nodes using the following command:

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

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

For example, to obtain statistics for node 2, volume `vol1`, use the following command:

```
# vxstat -g group1 -n 2 vol1
```

This command produces output similar to the following:

TIME(ms)		OPERATIONS		BLOCKS	AVG
TYP	NAME	READ	WRITE	READ	WRITE

READ	WRITE				
vol	vol1	2421	0	600000	0
99.0	0.0				

To obtain and display statistics for the entire cluster, use the following command:

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, the **vxstat -b** command returns 300 I/Os.

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.
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.
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.

8 **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.

The following topics are covered in this chapter:

- “Reattaching Disks”
- “VxVM Boot Disk Recovery”
- “Reinstallation Recovery”
- “Detecting and Replacing Failed Disks”
- “Plex and Volume States”
- “RAID-5 Volume Layout”
- “Creating RAID-5 Volumes”
- “Initializing RAID-5 Volumes”
- “Failures and RAID-5 Volumes”
- “RAID-5 Recovery”
- “Miscellaneous RAID-5 Operations”

NOTE

Rootability or bringing the root disk under the VxVM control is not supported on HP-UX 11i systems, but it is supported on HP-UX 11i Version 1.5 systems.

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, the `vxreattach` command 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.

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

VxVM Boot Disk Recovery

If there is a failure to boot from the VxVM boot disk on HP-UX 11i Version 1.5, use one of the following methods to recover. The method you choose depends on the type of failure:

- use the recovery media if:
 - the kernel is corrupted
 - files required for boot up are missing
- use the VxVM maintenance mode boot if:
 - LABEL file is missing or corrupt
 - /etc/vx/volboot file is missing or corrupt
 - ioconfig file is missing or corrupt
 - device files are missing or corrupt

Symptoms and procedures are described in the white paper, *VxVM Maintenance Mode Boot*.

If these methods fail, use the “Reinstallation Recovery” procedures described below.

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. On HP-UX 11i Version 1.5, first use the recovery methods described in “VxVM Boot Disk Recovery”. Follow the procedures below only if those methods fail.

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.

NOTE

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, during Volume Manager installation, 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. For information on replacing disks, see Chapter 4, Disk Tasks,.

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, use the following procedure. These steps are described in detail in the sections that follow:

Step 1. Prepare the system for installation.

Replace any failed disks or other hardware, and detach any disks not involved in the reinstallation.

Step 2. Install the operating system.

Reinstall the base system and any other unrelated Volume Manager packages.

Step 3. Install Volume Manager.

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

Step 4. Recover the Volume Manager configuration.

Step 5. Clean up the Volume Manager configuration.

Restore any information in volumes affected by the failure or reinstallation, and recreate 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, involve only 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. 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.

Ensure 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 on HP-UX 11i systems, follow the

instructions for loading the Volume Manager (from CD-ROM) in the *VERITAS Volume Manager 3.1 for HP-UX Release Notes*.

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

Recovering the Volume Manager Configuration

Once the Volume Manager package has been loaded, recover the Volume Manager configuration using the following procedure:

Step 1. Touch `/etc/vx/reconfig.d/state.d/install-db`.

Step 2. Shut down the system.

Step 3. Reattach the disks that were removed from the system.

Step 4. Reboot the system.

Step 5. When the system comes up, bring the system to single-user mode using the following command:

```
# exec init S
```

Step 6. When prompted, enter the password and press Return to continue.

Step 7. Remove files involved with installation that were created when you loaded Volume Manager but are no longer needed using the following command:

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

Step 8. Start some Volume Manager I/O daemons using the following command:

```
# vxiod set 10
```

Step 9. Start the Volume Manager configuration daemon, `vxconfigd`, in disabled mode using the following command:

```
# vxconfigd -m disable
```

Step 10. Initialize the `vxconfigd` daemon using the following command:

```
# vxdctl init
```

Step 11. Initialize the DMP subsystem using the following command:

```
# vxctl initdmp
```

Step 12. Enable vxconfigd using the following command:

```
# vxctl 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. For information on replacing disks, see Chapter 4, Disk Tasks,. 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 the vxctl enable command is successful, 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 [<Undefined Cross-Reference>](#). See “Hot-relocation Startup” for more information on starting hot-relocation.

Configuration Cleanup

To clean up the configuration of your system after reinstallation of the Volume Manager, you must address the following issues:

- volume cleanup
- disk cleanup
- final reconfiguration

- hot-relocation startup

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, perform these steps:

- Step 1.** Establish which VM disks have been removed or reinstalled using the following 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
c0t0d0	simple	-	-	error
c0t1d0	simple	disk02	rootdg	online
c0t2d0	simple	disk03	rootdg	online
-	-	disk01	rootdg	failed was:

c0t0d0

NOTE

Your system may use a *device name* that differs from the examples. For more information on device names, see “Disk Devices”.

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

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.

- Step 2.** Once you know which disks have been removed or replaced, locate all the mirrors on failed disks using the following 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.

- Step 3.** Check the status of each volume and print volume information using the following command:

```
# 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 produces the following output:

```
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 c1t5d 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.

- Step 4.** To remove the volume `v01`, use the following 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 produces the following output:

```
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
plv02-01v02DISABLEDNODEVICE30720STRIPE3/128RW
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.

Step 5. Use the `vxedit` command to remove the volume, as described in step 4.

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 the following:

```
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 307200 c1t3d0 ENA
pl v03-02 v03 DISABLED NODEVICE 30720 CONCAT - RW
sd disk01-04 v03-02 disk03 262144 307200 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.

- Step 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 disassociate and remove the plex v03-02, use the following command:

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

- Step 7.** Once all the volumes have been cleaned up, you must clean up the disk configuration as described in the following 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 `vxdbg` command. To remove the failed disk disk01, use the following command:

```
# vxdbg rmdisk disk01
```

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

Final Reconfiguration Once the root disk is under VxVM control, any other disks that were replaced should be added using the `vxdiskadm` command.

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 the `vxassist` command or the graphical user interface.

For example, to recreate the volumes v01 and v02, use the following 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 the `vxassist` command as described in “Mirroring Guidelines”.

To replace the plex removed from volume `v03`, use the following command:

```
# vxassist mirror v03
```

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

Hot-relocation Startup 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 the `vxnotify` process) to start.

To start hot-relocation, first start the watch daemon. This sends email to the administrator when any problems are found. To change the address used for sending problem reports, change the argument to `vxrelocd` using one of the following commands:

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

or

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

To determine whether or not hot-relocation has been started, use the following command:

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

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

NOTE

You may need an additional license to use this feature.

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. See “Hot-Relocation” for more information.

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 `vxrelocd` hot-relocation daemon 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, the `vxrelocd` daemon informs the system administrator by electronic mail of the failure and which Volume Manager objects are affected. The `vxrelocd` daemon then determines which subdisks (if any) can be relocated. If relocation is possible, the `vxrelocd` daemon 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, the `vxrelocd` daemon 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 “Placing Disks Under Volume Manager Control”. 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 the vxrelocd Process

Hot-relocation is turned on as long as the vxrelocd process is running. As a rule, leave hot-relocation turned on to take advantage of this feature if a failure occurs. However, if you disable this feature because you do not want the free space on some of your disks used for relocation, you must prevent the vxrelocd process from starting at system startup time.

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 the vxrelocd process behaves by either editing the vxrelocd line in the startup file that invokes the vxrelocd process (/sbin/rc2.d/S095vxvm-recover) or killing the existing vxrelocd process and restarting it with different options. After making changes to the way the vxrelocd process is invoked in the startup file, 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. Also restart the daemon immediately so that hot-relocation can take effect if a failure occurs.

You can alter the vxrelocd process in the following ways:

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

```
# vxrelocd root user_name1 user_name2 &
```

- To reduce the impact of recovery on system performance, you can instruct the `vxrelocd` process to increase the delay between the recovery of each region of the volume using the following command:

```
# 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

Use the `vxdbg` command `spare` to display information about all of the spare disks available for relocation. The output displays the following information:

```
GROUP DISK DEVICE TAG OFFSET LENGTH FLAGS
rootdg disk02 c0t2d0 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.

To display information about disks that are currently designated as spares, use the following commands:

- `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. To improve performance, move the relocated subdisks (after hot-relocation is complete).

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, an email messages is sent to `root`, as 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 c0t5d0.
The available plex home-01 will be used to recover the
data.
```

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

For example, the following message indicates 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 “Replacing Disks”). Once this is done, you can move a relocated subdisk back to the original disk. For example, move the relocated subdisk disk05-01 back to disk02 using the following command:

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

NOTE

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

Moving Hot-Relocated Subdisks

NOTE

You may need an additional license to use this feature.

After the disk that experienced the failure is fixed or replaced, vxunreloc can be used to move all the hot-relocated subdisks back to the disk. When a subdisk is hot-relocated, its original disk media name and the offset into the disk, are saved in the configuration database. When a subdisk is moved back to the original disk or to a new disk using

`vxunreloc`, the information is erased. The original `dm` name and the original offset are saved in the subdisk records. To print all of the subdisks that were hot-relocated from `disk01` in the `rootdg` disk group, use the following command:

```
# vxprint -g rootdg -se 'sd_orig_dmname="disk01"'
```

To move all the subdisks that were hot-relocated from `disk01` back to the original disk, type:

```
# vxunreloc -g rootdg disk01
```

The `vxunreloc` utility provides the `-n` option to move the subdisks to a different disk from where they were originally relocated. For example, when `disk01` failed, all the subdisks that resided on it were hot-relocated to other disks. After the disk is repaired, it is added back to the disk group using a different name, for example, `disk05`. To move all the hot-relocated subdisks to the new disk, use the following command:

```
# vxunreloc -g rootdg -n disk05 disk01
```

The destination disk should have at least as much storage capacity as was in use on the original disk. If there is not enough space, the `unrelocate` operation fails and none of the subdisks are moved.

When the `vxunreloc` utility moves the hot-relocated subdisks, it moves them to the original offsets. However, if some subdisks occupy part or all of the area on the destination disk, the `vxunreloc` utility fails. If the `vxunreloc` utility fails, perform one of the following procedures:

- move the existing subdisks somewhere else, and then re-run the `vxunreloc` utility
- use the `-f` option provided by the `vxunreloc` utility to move the subdisks to the destination disk, but allow the `vxunreloc` utility to find the space on the disk.

As long as the destination disk is large enough so that the region of the disk for storing subdisks can accommodate all subdisks, all the hot-relocated subdisks are “unrelocated” without using the original offsets.

A subdisk that has been hot-relocated more than once due to multiple disk failures can still be unrelocated back to its original location. For example, if `disk01` fails, a subdisk named `disk01-01` is moved to `disk02`. If `disk02` then experiences disk failure, all the subdisks residing on `disk02`, including the one that was hot-relocated to it, are moved again. When `disk02` is replaced, an `unrelocate` operation for

disk02 does not affect the hot-relocated subdisk disk01-01. However, a replacement of disk01, followed by the `unrelocate` operation, moves disk01-01 back to disk01 when the `vxunreloc` utility is run, immediately after the replacement.

Restart vxunreloc After Errors

Internally, the `vxunreloc` utility moves the subdisks in three phases. The first phase creates as many subdisks on the specified destination disk as there are subdisks to be unrelocated. When the subdisks are created, the `vxunreloc` utility fills in the comment field in the subdisk record with the string `UNRELOC` as an identification. The second phase moves the data. If all the subdisk moves are successful, the third phase cleans up the comment field of the subdisk records.

Creating the subdisk is an all-or-none operation. If the `vxunreloc` utility cannot create all the subdisks successfully, no subdisk is created and the `vxunreloc` utility exits. The subdisk move operation is not all-or-none. One subdisk move is independent of another, and as a result, if one subdisk move fails, the `vxunreloc` utility prints an error message and then exits. However, all of the subsequent subdisks remain on the disk where they were hot-relocated and are not moved back. For subdisks that were returned home, the comment field in their subdisk records is still marked as `UNRELOC` because the cleanup phase is never executed.

If the system goes down after the new subdisks are made on the destination, but before they are moved back, the `unrelocate` utility can be executed again after the system comes back. As described above, when a new subdisk is created, the `vxunreloc` utility sets the comment field of the subdisk as `UNRELOC`. When the `vxunreloc` utility is re-executed, it checks the `offset`, the `len`, and the `comment` fields of the subdisk on the destination disk to determine if it was left on the disk at a previous execution of the `vxunreloc` utility.

NOTE

Do not manually modify the string `UNRELOC` in the comment field.

If one out of a series of subdisk moves fails, the `vxunreloc` utility exits. Under this circumstance, you should check the error that caused the subdisk move to fail and determine if the unrelocation can proceed. When you re-execute the `vxunreloc` utility to resume the subdisk moves, it uses the subdisks created at a previous run.

The cleanup phase requires one transaction. The `vxunreloc` utility resets the comment field to a NULL string for all the subdisks marked UNRELOC that reside on the destination disk. This includes cleanup of subdisks that were unrelocated in a previous invocation of the `vxunreloc` utility that did not successfully complete.

Detecting Failed Disks

The Volume Manager hot-relocation feature automatically detects disk failures and notifies the system administrator of the failures by email. If hot-relocation is disabled or you miss the email, view disk failures using the `vxprint` command or by using the graphical user interface to view the status of the disks. Driver error messages are also displayed 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 listing 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 the following 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 the `vxrelocd` Process” for information on how to send the mail to users other than `root`.

To determine which disk is causing the failures in the above example, use the following command:

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

The following 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 the disk has become detached, this command does not perform any recovery. 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 email lists the disk that failed and all plexes that use the disk. The following is an example of an email:

```
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 the `vxdiskadm` utility and selecting item 4(Replace a failed or removed disk) from the main menu. If any initialized but unadded disks are available, select one of those disks as a replacement.

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, 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 the following command:

```
# vxinfo
```

Any volumes that are listed as `Unstartable` must be restored from backup.

To restart the volume `mkting` so that it can be restored from backup, use the following 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. To replace the disk, use the following procedure:

Step 1. Detach the disk from its disk group.

Step 2. Replace the disk with a new one.

To detach the disk, run the `vxdiskadm` utility and select item 3 (Remove a disk for replacement) from the main menu. If initialized disks are available as replacements, specify the disk as part of this operation. Otherwise, specify the replacement disk later by selecting item 4 (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. The following 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 the `vxdiskadm` utility and save the volume. Either back up the volume or move the volume off of the disk. For example, to move the volume `mkting` to a disk other than `disk02`, use the following command:

```
# vxassist move mkting !disk02
```

After the volume is backed up or moved, run the `vxdiskadm` utility 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 4 (Replace a failed or removed disk) from the `vxdiskadm` main menu.

Plex and Volume States

The following sections describe plex and volume states.

Plex States

Plex states reflect whether or not plexes are complete and are consistent copies (mirrors) of the volume contents. Volume Manager utilities automatically maintain the plex state. However, if a volume should not be written to because there are changes to that volume and if a plex is associated with that volume, you can modify the state of the plex. 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 and be attached to a volume later.

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 is stopped as a result of a system crash and the plex is 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, the **vxconfigd** utility 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 request 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

Some volume states are similar to plex states. The following are volume 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, as follows:

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 Layout

NOTE You may need an additional license to use this feature.

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, as follows:

- the **RAID-5 plex** is used to keep both data and parity for the volume
- the **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. See the following display:

```
PL NAME VOLUME KSTATE STATE LENGTH LAYOUT NCOL/WID MODE
SD NAME PLEX DISK DISKOFFS LENGTH [COL/]OFF DEVICE MODE
pl r5vol-01 rvol ENABLED ACTIVE 20480 RAID 3/16 RW
sd disk00-00 rvol-01 disk00 010240 0/0 c1t4d1 ENA
sd disk01-00 rvol-01 disk01 010240 1/0 c1t2d1 ENA
sd disk02-00 rvol-01 disk02 010240 2/0 c1t3d1 ENA
```

NOTE Your system may use a **device name** that differs from the examples. For more information on device names, see “Disk Devices”.

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. See the following vxprint output for a RAID-5 volume:

```
VNAMEVRGKSTATESTATELENGTHUSETYPEPEFPLEXRDPOL
PL NAMEVOLUMEKSTATESTATELENGTHLAYOUTNCOL/WIDMODE
SDNAMEPLEXDISKDISKOFFSLENGTH[COL/]OFFDEVICEMODE
vr5volRAID-5ENABLEDDEGRADED20480RAID-
plr5vol-01r5volENABLEDACTIVE20480RAID3/16RW
sddisk00-00r5vol-01disk000102400/0c1t4d1ENA
sddisk01-00r5vol-01disk010102401/0c1t2d1ENA
sddisk02-00r5vol-01disk020102402/0c1t3d1ENA
plr5vol-l1r5volENABLEDLOG1024CONCAT-RW
sddisk03-01r5vol-l1disk00010240c1t3d0ENA
plr5vol-l2r5volENABLEDLOG1024CONCAT-RW
sddisk04-01r5vol-l2disk02010240c1t1d1ENA
```

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

NOTE

You may need an additional license to use this feature.

You can create RAID-5 volumes by using either the `vxassist` command (recommended) or the `vxmake` command. 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 Command and RAID-5 Volumes

Create a RAID-5 volume by using the following command:

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

For example, to create a 10M RAID-5 volume named `volraid`, use the following command:

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

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

vxmake Command and RAID-5 Volumes

You can create a RAID-5 volume by using the `vxmake` command, which is similar to the command used to create other volumes (see “Creating Volumes”). Also see the `vxmake(1M)` manual page for details on creating other volumes. Subdisks for use in a RAID-5 volume are created using the same method as other subdisks.

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. For example, to create a four-column RAID-5 plex with a stripe unit size of 32 sectors, use the following 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, the `vxmake` command assumes a four-column RAID-5 plex and places one subdisk in each column. Striped plexes are created using this same method. If the subdisks are to be created later, use the following 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 the `vxsd assoc` command (see “Manipulating RAID-5 Subdisks”).

For example, to create a three-column RAID-5 plex using six subdisks, use the following 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, specify the usage type to be RAID-5 using the following command:

```
# vxmake -Uraid5 vol raidvol
```

RAID-5 plexes and RAID-5 log plexes can be associated implicitly using the following command:

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

Initializing RAID-5 Volumes

NOTE

You may need an additional license to use this feature.

A RAID-5 volume must be initialized if it was created by the `vxmake` command and has not yet been initialized or if it has been set to an uninitialized state. If RAID-5 is created using the `vxassist` command with default options, then the volume is initialized by the `vxassist` command.

To initialize a RAID-5 volume, use one of the following commands:

```
# vxvol init zero volume_name
```

or

```
# vxvol start volume_name
```

The `vxvol init zero` command 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.

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

Failures and RAID-5 Volumes

NOTE

You may need an additional license to use this feature.

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 the following display:

```
VNAMEVVGKSTATESTATELENGTHHSETYPEPEFPLEXRDPOL
PLNAMEVOLUMEKSTATESTATELENGTHLAYOUTNCOL/WIDMODE
SDNAMEPLEXDISKDISKOFFSLENGTH[COL/]OFFDEVICEMODE
SVNAMEPLEXVOLNAMENVOLLAYRLLENGTH[COL/]OFFAM/NMMODE
vr5vol-ENABLEDDEGRADED204800raid5-RAID
plr5vol-01r5volENABLEDACTIVE204800RAID3/16RW
sddisk01-01r5vol-01disk0101024000/0c2t9d0ENA
sddisk02-01r5vol-01disk0201024001/0c2t10d0ENA
sddisk03-01r5vol-01disk0301024002/0c2t11d0ENA
plr5vol-02r5volENABLEDLOG1440CONCAT-RW
sddisk04-01r5vol-02disk04014400c2t12d0ENA
```

```
plr5vol-03r5volENABLEDLOG1440CONCAT-RW
sddisk05-01r5vol-03disk05014400c2t14d0ENA
```

The volume `r5vol` is in degraded mode, as shown by the volume 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 the `vxprint -ht` command, failure within a RAID-5 log plex is indicated by the plex state being BADLOG. This is shown in the following display, where the RAID-5 log plex `r5vol-l1` has failed:

```
VNAMERVGKSTATESTATELENGTHUSETYPEPEPREFPLEXRDPOL
PLNAMEVOLUMEKSTATESTATELENGTHLAYOUTNCOL/WIDMODE
SDNAMEPLEXDISKDISKOFFSLENGTH[COL/]OFFDEVICEMODE
vr5volRAID-5ENABLEDACTIVE20480RAID-
plr5vol-01r5volENABLEDACTIVE20480RAID3/16RW
sddisk00-00r5vol-01disk000102400/0c1t4d1ENA
sddisk01-00r5vol-01disk010102401/0c1t2d1dS
sddisk02-00r5vol-01disk020102402/0c1t3d1ENA
plr5vol-11r5volDISABLEDBADLOG1024CONCAT-RW
sddisk03-01r5vol-11disk001024010240c1t3d0ENA
plr5vol-12r5volENABLEDLOG10240-RW
sddisk04-01r5vol-12disk021024010240c1t1d1ENA
```

RAID-5 Recovery

NOTE

You may need an additional license to use this feature.

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 initiates a parity resynchronization.

In the case of a failing RAID-5 log plex, relocation only occurs if the log plex is mirrored; the `vxrelocd` daemon 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. For an example of the output of the `vxprint -ht` command, see the following example for a stale RAID-5 volume:

```
VNAMEUSETYPEKSTATESTATELENGTHTHREADPOLPREFFLEX
PLNAMEVOLUMEKSTATESTATELENGTHLAYOUTNCOL/WIDMODE
SDNAMEPLEXDISKDISKOFFSLENGTH[COL/]OFFDEVICEMODE
vr5volRAID-5ENABLEDNEEDSYNC20480RAID-
plr5vol-01r5volENABLEDACTIVE20480RAID3/16RW
sddisk00-00r5vol-01disk000102400/0c1t4d1ENA
sddisk01-00r5vol-01disk010102401/0c1t2d1ENA
sddisk02-00r5vol-01disk020102402/0c1t3d1ENA
```

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 8-1, Invalid RAID-5 Volume,, use the following command:

```
# vxvol resync r5vol
```

Parity is regenerated by issuing `VOL_R5_RESYNC` ioctl's 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 the `-o iosize` option 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 checkpoint=size` option controls how often the checkpoint is saved. If the option is not specified, the default checkpoint size is used.

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, the process doing the recovery can crash, or the can volume start with an option to prevent subdisk recovery. In addition, the disk on which the subdisk resides can be 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 8-1, Invalid RAID-5 Volume,, use the following command:

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

A RAID-5 volume that has multiple stale subdisks can be caught up all at once. To catch multiple stale subdisks, use the `vxvol recover` command with only the volume name, as follows:

```
# vxvol recover r5vol
```

Recovering Logs After Failures

RAID-5 log plexes can become detached due to disk failures, as shown in Figure 8-2, Read-Modify-Write,. These RAID-5 logs can be reattached by using the `att` keyword for the `vxplex` command. To reattach the failed RAID-5 log plex, use this command:

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

Miscellaneous RAID-5 Operations

NOTE

You may need an additional license to use this feature.

Many operations exist for manipulating RAID-5 volumes and associated objects. These operations are usually performed by other commands, such as the `vxassist` command and the `vxrecover` command, as part of larger operations, such as evacuating disks. These command line operations are not 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. To a RAID-5 log can be added, use the following command:

```
# vxplex att r5vol r5log
```

The attach (`att`) 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 contains 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 are 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 ioctl`s.

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 using the following command:

```
# vxplex dis r5log3
```

When 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 in the `vxprint` output for a RAID-5 volume on “Disk Failures” page 247, use the following 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 ioctl`s. Otherwise, it is marked as stale and is recovered when the volume is started.

To remove subdisks from the RAID-5 plex, use the following command:

```
# 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 the `vxsd mv` utility. For example, if `disk03` is to be evacuated and `disk22` has enough room by using two portions of its space, use the following 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 the vxsd utility 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 the vxsd split command and the vxsd join command. These operations work the same way as those 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

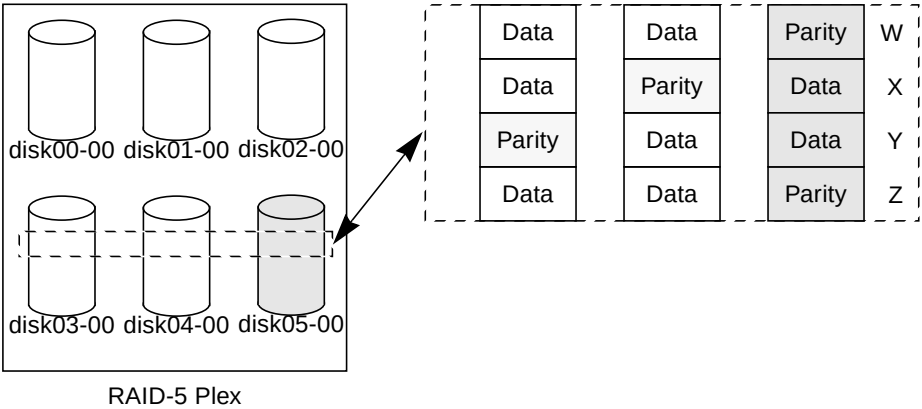
When this occurs, the `vxvol start` command returns the following error message:

```
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 8-1, Invalid RAID-5 Volume, which shows a RAID-5 volume that has become invalid due to stale parity and a failed subdisk.

Figure 8-1 Invalid RAID-5 Volume



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 the `vxvol start` command is as follows:

```
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 is not always 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:

- Step 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 the `vxvol` command. 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.

- Step 2.** Any existing logging plexes are zeroed and enabled. If all logs fail during this process, the start process is aborted.
- Step 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.
- Step 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.

- Step 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.

- Step 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 the following 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 using the following 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 is 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

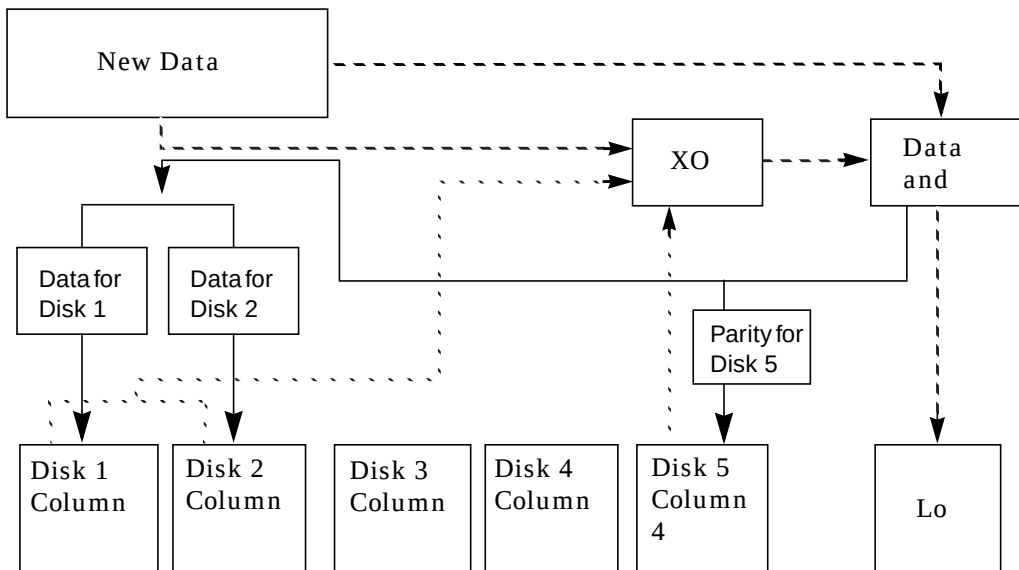
When you write to a RAID-5 array, the following procedure is used for each stripe involved in the I/O:

- Step 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.

- Step 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.
- Step 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. See Figure 8-2, Read-Modify-Write,

Figure 8-2 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.

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.

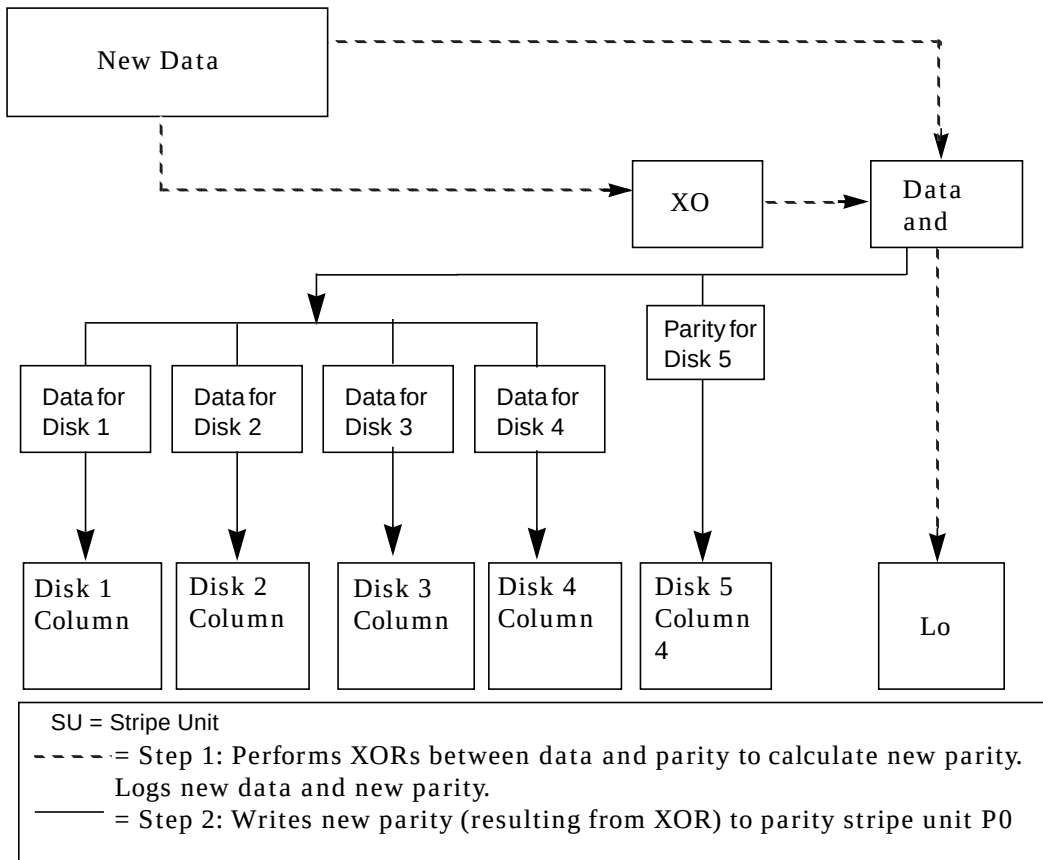
A full-stripe write procedure consists of the following steps:

Step 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.

Step 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. See Figure 8-3, Full-Stripe Write,

Figure 8-3 Full-Stripe Write



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:

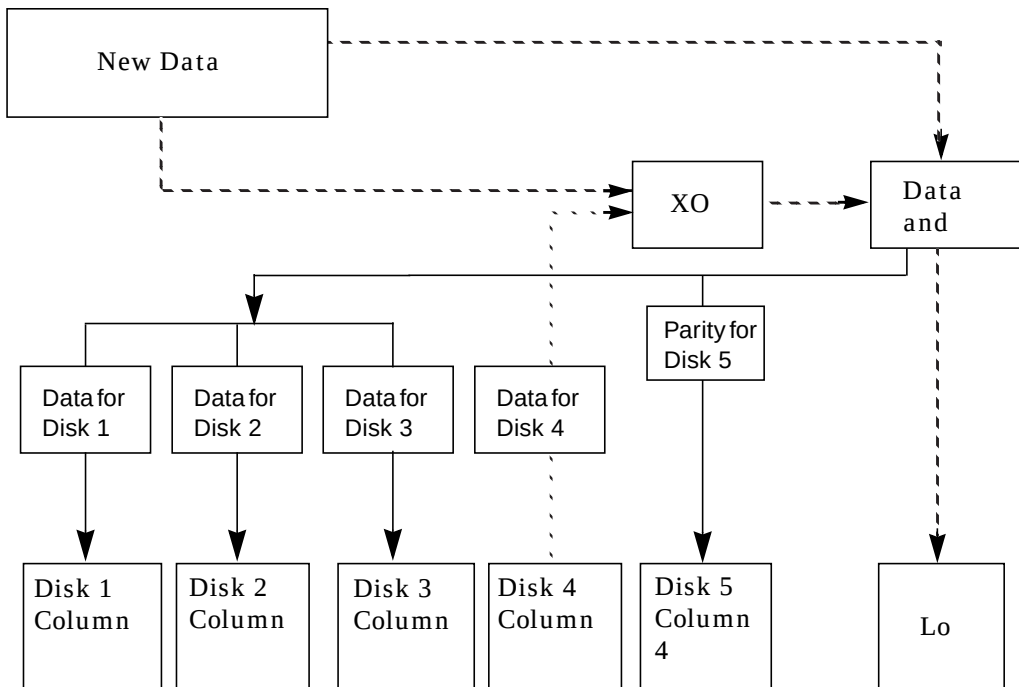
Step 1. Unaffected data is read from the unchanged data stripe unit(s).

Step 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.

Step 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.

See Figure 8-4, 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 8-4 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

9 **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 contains information on Volume Manager features. Volume Manager provides flexibility in configuring storage to improve system performance. Two basic strategies are 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

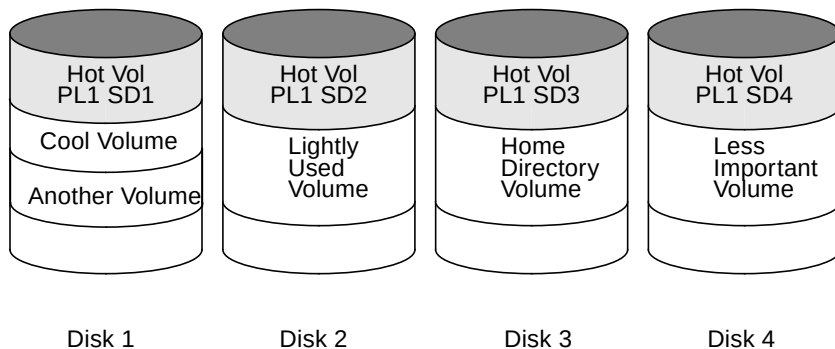
Striping is a way of “slicing” data and storing it across multiple devices in to improve access performance. Striping provides increased access bandwidth for a plex. Striped plexes improve access performance for both read and write operations.

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 9-1, Use of Striping for Optimal Data Access,, 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 9-1 **Use of Striping for Optimal Data Access**



Mirroring

NOTE

You may need an additional license to use this feature.

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

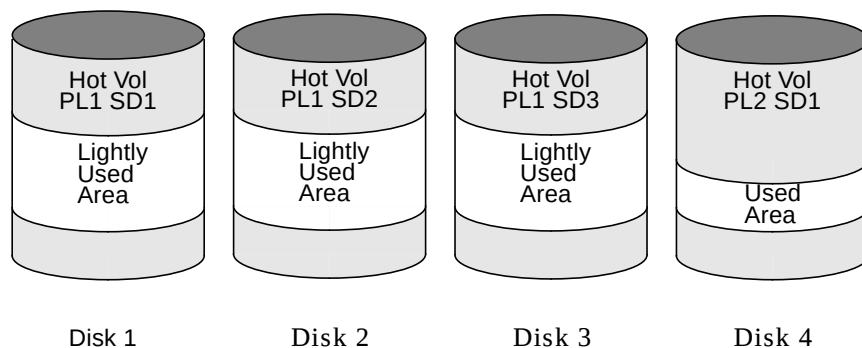
30 percent writes), then mirroring can result in somewhat reduced performance.

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 9-2, Use of Mirroring and Striping for Improved Performance, 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 9-2 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

NOTE

You may need an additional license to use this feature.

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

NOTE

You may need an additional license to use this feature.

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”.

Using RAID-5

NOTE

You may need an additional license to use this feature.

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.

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 a 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` command, 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 Command)

The `vxstat` command accesses activity information on volumes, plexes, subdisks, and disks under Volume Manager control. The `vxstat` utility

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 the vxstat utility. For detailed information on available options, see 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.

The vxstat utility can also reset the statistics information to zero. Use the vxstat -r command 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.

The following is an example of output produced using the vxstat utility:

		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 Command)

The `vxtrace` command traces operations on volumes. The `vxtrace` command 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, see 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 `vxstat -r` utility 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 `vxstat` command with no arguments. The following is an example of output produced using the `vxstat` command with no arguments:

TYP	NAME	OPERATIONS		BLOCKS		AVG TIME(ms)	
		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 `vxstat -d` command. The following is an example of disk statistics produced using the `vxstat -d` command:

		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

To move the volume archive onto another disk, first identify which disk(s) it is on using the following command:

vxprint -tvh archive

The following is an example display:

```
VNAMESETYPESTATESTATELENGTHTHREADPOLREFPLEX
PLNAMEVOLUMEKSTATESTATELENGTHLAYOUTNCOL/WDTHMODE
SDNAMEPLEXPLOFFSDISKOFFSLENGTH[COL/]OFFFLGAS
varchivefsgenENABLEDACTIVE204800SELECT-
plarchive-01archiveENABLEDACTIVE204800CONCATRW
sddisk03-03archive-010409600204800/0c1t2d0
```

NOTE

Your system may use a **device name** that differs from the examples. For more information on device names, see “Disk Devices”.

The associated subdisks list indicates that the archive volume is on disk disk03. To move the volume off disk03, use the following 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 specify a *dest_disk*, the volume is moved to an available disk with enough space to contain the volume.

For example, to move the volume from disk03 to disk04, use the

following command:

```
# 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 the following 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 `/stand/system`. Other values can only be tuned using the command line interface to the Volume Manager.

Tunables

In many cases, tunables are contained in the `volinfo` structure, as described in the `vxio(7)` manual page. Tunables are modified by adding a line to `/stand/system` file. Changed tunables take effect only after relinking the kernel and booting the system from the new kernel.

Modify tunables with the following format:

```
#tunable tunable_values
```

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. I/O requests to the volume manager larger than this value are 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).

Raising this limit can cause difficulties if the size of an I/O causes the process to take more memory or kernel virtual mapping space than exists and thus deadlock. The maximum limit for `vol_maxio` is 20% of the smaller of physical memory or kernel virtual memory. It is inadvisable to go over this limit since deadlock is likely to occur.

If you have stripes larger than `vol_maxio`, full stripe I/Os are broken up

which prevents full-stripe read/writes. This throttles the volume I/O throughput for sequential I/O or larger I/O.

This tunable limits the size of an I/O at the top of the volume manager, not at the individual disk. For example, If you have an 8x64K stripe, then the 256K value only allows I/Os that use half the disks in the stripe and thus it cuts potential throughput in half. If you have more columns or you've used a larger interleave factor, then your relative performance is worse.

This tunable should be set, as a minimum, to the size of your largest stripe. This tunable guideline applies to both Raid0 striping and Raid5 striping.

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.

voldr1_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 `voldr1_max_drtregs` 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_maxpool_sz

This tunable defines the maximum memory used by a Volume Manager from the system for its internal purposes. The default value of this tunable is 4 Megabytes. This tunable has a direct impact on the performance of VxVM.

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 the `vxconfigd(1M)` command 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 `vx dg init` command for a new group setup (see the `vx dg(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 the following command:

```
# vxedit set nconfig=5 foodg
```

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.

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 c0t0d0. The c#t#d# syntax identifies the controller, target address, and disk.

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 vxddmpadm 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 c0t0d0. 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.

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. Encapsulation is not applicable on some systems.

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 `util[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.

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. Sector size is set per device (hard drive, CD-ROM, and so on). Although all devices within a system are usually configured to the same sector size for interoperability, this is not always the case. 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/rdisk`, 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.

s2

s2

b#

b0

b#

s0

b0

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*).

private disk group A disk group in which the disks are accessed by one specific host.

node One of the hosts in a cluster.

master node A node that is distinguished from the other nodes.

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-shareable disk group A disk group in which the disks are shared between more than one host.

In a cluster, an atomic operation takes place either on all nodes or not at all.

A

- adding
 - a disk, 153, 160
 - a disk to a disk group, 187, 209
 - a DRL log, 262
 - a mirror to a volume, 248
 - a RAID-5 log, 260
- adding disks, 157
 - format, 157
- associating log subdisks, 283
 - vxsd, 283
- associating mirrors
 - vxmake, 265
- associating plexes
 - vxmake, 265
- associating subdisks
 - vxmake, 282
 - vxsd, 282

B

- backing up a volume, 255
- backups, 82, 94, 287
 - mirrors, 251
 - vxassist, 94, 287
- Boot Disk recovery, 336

C

- changing volume attributes, 382
- checkpoint, 374
- cli
 - disk device names, 85
- cluster
 - disks, 306
 - shared objects, 297
 - terminology, 331
- cluster environment, 296
- cluster functionality, 296, 297
- cluster protocol range, 322
- cluster protocol version, 322
- cluster reconfiguration, 301
- cluster-shareable disk group, 297
- columns, in striping, 40
- command-line utilities, 150
- concatenated volume
 - creating, 238
- concatenation, 38
- concepts, 58
- configuration guidelines, 390
- conversion
 - storage layout, 97

- copying mirrors
 - vxplex, 274
- creating
 - a concatenated volume, 238
 - a RAID-5 volume, 240
 - a spanned volume, 238
 - a striped volume, 239
 - a volume, 131, 132, 133, 134, 237
 - a volume on a VM disk, 239
- creating mirrors
 - vxmake, 264
- creating RAID-5 volumes, 367
- creating subdisks, 275
 - vxmake, 275
- creating volumes, 366, 367
 - manually, 64
 - vxassist, 64

D

- daemons, 71
 - configuration, 135
 - hot-relocation, 103
 - Volume Manager, 71
 - vxrelocd, 105, 139, 349
- data
 - preserving, 81
 - redundancy, 45
- data assignment, 391
- defaults file
 - vxassist, 134
- degraded mode, 370
- deporting
 - a disk group, 154, 220
 - disk groups, 224
- description file, 136
- Dirty Region Logging, 112, 262
 - guidelines, 78
 - log subdisks, 283
- disabling
 - a disk, 155, 178
 - a disk group, 154, 220
- disk failures, 370
 - and recovery, 334
- disk group utilities, 150, 211
- disk groups, 32, 209
 - adding a disk, 209
 - creating, 212
 - default, 212
 - deporting, 220, 224, 225
 - disabling, 220

- displaying information, 231
- enabling, 154
- importing, 218, 223, 224, 225
- initializing, 212
- moving, 223, 224, 225
- moving between systems, 224
- removing, 228
- renaming, 216
- using, 227
- disk information, displaying, 205
- disk media name, 31, 149
- disk names, 209
- disk utilities, 150, 211
- disks, 149
 - adding, 157
 - detached, 175, 354
 - disabling, 178
 - displaying information, 205
 - enabling, 173
 - failure, 347, 370
 - and hot-relocation, 103
 - and recovery, 334
 - hot-relocation spares, 350
 - in a cluster, 306
 - initialization, 157
 - mirroring volumes on, 249, 291
 - moving, 172
 - moving volumes from, 258, 292
 - physical
 - adding, 153, 160
 - adding to disk group, 187
 - bringing under VxVM control, 160
 - disabling, 155
 - displaying information, 202
 - enabling, 155
 - moving volumes from, 154
 - removing, 153, 182
 - replacing, 154, 179
 - reserving, 201
 - reattaching, 335
 - reinitializing, 197
 - removing, 182
 - replacing, 356, 357
 - VM
 - creating a volume on, 239
 - VM disks, 31
- displaying
 - disk group information, 231
 - disk information, 202, 205
 - free disk space, 87
 - multipath information, 62, 202

- volume configuration, 254
- displaying subdisks
 - vxprint, 89
- dissociating mirrors
 - vxplex, 252, 266
- DMP
 - dynamic multipathing, 122
 - load balancing, 123
 - path failover mechanism, 122
- DMP configuration, 135
- DRL, 262
- dynamic multipathing
 - DMP, 122

E

- enabling
 - a disk, 155, 173
 - a disk group, 154
 - access to a disk group, 218
- exiting
 - vxdiskadm, 253

F

- failed disks, 347
 - detecting, 175, 354
- failures, 374
 - disk, 370
 - system, 369
- Fast Mirror Resynchronization
 - see FastResync, 114, 289, 313, 314
- FastResync, 114, 288, 313
- FMR
 - see FastResync, 114, 289, 314
- forcibly starting volumes, 379
- format utility, 157
- free space
 - displaying, 87

G

- getting performance data, 396
- graphical user interface, 56, 150
- guidelines
 - Dirty Region Logging, 78
 - mirroring, 77
 - mirroring and striping, 79
 - RAID-5, 80

H

- hosts

- multiple, 297
- hot-relocation, 103, 155
 - designating spares, 104
 - modifying vxrelocd, 138, 349
 - removing spares, 185, 192

I

I/O

- statistics, 398
 - obtaining, 396
- tracing, 398, 401

- I/O daemon, 72

importing

- disk groups, 224
- importing a disk group, 154, 218
- increasing volume size, 243
- information, 286
- initializing disks, 157

J

- joining subdisks
 - vxsd, 281

L

layout

- left-symmetric, 50

listing mirrors

- vxprint, 267

load balancing

- DMP, 123

log

- adding, 262
- RAID-5, 260

log plexes, 364

- log subdisks, 78, 112, 283, 308

- associating, 283

logging, 52

- logs, 364, 375

M

- managing volumes, 131, 132, 133, 134

- manipulating subdisks, 376

- master node, 299

minor numbers

- reserving, 230

- mirroring, 45, 154, 392, 394

- all volumes, 249

- guidelines, 77

- volumes on a disk, 249, 291

- mirrors, 33, 35
 - adding to a volume, 248
 - backup using, 251
 - creating, 264
 - displaying, 267
 - dissociating, 252, 266
 - offline, 241, 242, 270
 - online, 241, 242
 - recover, 176, 355
 - removing, 252, 266
- moving
 - volumes from a disk, 154, 258, 292
- moving disk groups
 - vx dg, 223, 224
 - vxrecover, 223, 224
- moving disks, 172
- moving mirrors, 273
 - vxplex, 273
- moving subdisks
 - vx sd, 277

N

- name
 - disk access, 30
 - disk media, 31, 149
- nodes, 297

O

- OFFLINE, 270
- offlining
 - a disk, 155, 178
- online backup, 94, 287
- online relayout, 97
 - failure recovery, 101
 - how it works, 98
 - transformation characteristics, 101
 - transformations and volume length, 101
 - transformations not supported, 102
 - types of transformation, 98

P

- parity, 44, 48, 374
- parity recovery, 373, 374
- partitions, 30
- path failover
 - DMP, 122
- pathname of a device, 149
- performance, 395
 - guidelines, 391

- management, 390
- monitoring, 396
- optimizing, 391
- priorities, 396
- performance data, 396
 - getting, 396
 - using, 398
- plex kernel states, 361
 - DETACHED, 361
 - DISABLED, 361
 - ENABLED, 361
- plex states, 358
 - ACTIVE, 359
 - CLEAN, 359
 - EMPTY, 358
 - IOFAIL, 361
 - OFFLINE, 360
 - STALE, 360
 - TEMP, 360
 - TEMPRM, 360
- plex states cycle, 361
- plexes, 33, 364
 - and volume, 35
 - as mirrors, 35
 - attach, 137
 - attaching, 270, 271
 - changing information, 268
 - copying, 274
 - creating, 264
 - definition, 33
 - detach, 137
 - detaching, 270
 - displaying, 267
 - listing, 267
 - moving, 273
 - offline, 241, 242
 - online, 241, 242
 - striped, 40
- private disk group, 297
- private region, 148
- public region, 148
- putil, 268

R

- RAID-0, 40
- RAID-1, 45
- RAID-5, 364, 366, 367, 369, 370, 373, 374, 375, 376, 377, 378, 379, 382, 395
 - guidelines, 80
 - recovery, 372, 380
 - snapshot, 94, 287

- subdisk moves, 376
- RAID-5 log, 260
- RAID-5 plexes, 364
- RAID-5 volume
 - creating, 240
- RAID-5 volumes, 380
- read
 - policies, 393
- reattaching disks, 335
- reconfiguration procedures, 338
- reconstructing-read, 370
- Recovery
 - VxVM Boot Disk, 336
- recovery, 334, 380
 - logs, 374
 - RAID-5 volumes, 372, 380
 - volumes, 257
- reinitializing disks, 197
- reinstallation, 338, 340
- removing
 - a disk, 153, 182
 - a DRL, 262
 - a physical disk, 182
 - a RAID-5 log, 261
 - a volume, 247
- removing disk groups, 228
 - vxdg, 228
- removing disks
 - vxdg, 183
- removing mirrors, 252
- removing subdisks
 - vxedit, 276
- renaming disk groups, 216
- replacing a disk, 154, 179
- replacing disks, 356
 - vxdiskadm, 356
- reserving disks for special purposes, 201
- resilvering, 126
- resynchronization
 - and Oracle databases, 126
 - volume, 110
- rootdg, 32, 209
 - renaming, 216

S

- shared objects, 297
- size
 - volumes
 - increasing, 243
- slave node, 299

- snapshot, 94, 95, 287, 288
 - RAID-5, 94, 287
- spanned volume
 - creating, 238
- spanning, 38
- splitting subdisks
 - vxsd, 280
- standard disk devices, 148
- starting
 - vxdiskadm, 205
- starting volumes, 377
- states
 - plex, 358
 - volume, 362
- Storage Administrator, 56
- storage layout
 - conversion, 97
- stripe column, 40
- stripe units, 41
- striped plex, 40
- striped volume
 - creating, 239
- striping, 40, 391, 394
- subdisk moves
 - RAID-5, 376
- subdisks, 64
 - associating, 89, 282
 - changing information, 286
 - creating, 275
 - displaying, 89
 - dissociating, 285
 - joining, 281
 - log, 112, 283, 308
 - moving, 277
 - removing, 276
 - splitting, 280
- system failures, 369

T

- tracing I/O, 398
 - vxtrace, 398
- tunables, 402
- tuning
 - Volume Manager, 402
- tutil, 268

U

- unusable volumes, 378
- using disk groups
 - vxassist, 227

- using I/O statistics, 398
- using performance data, 398
- utility descriptions
 - vxassist, 131
 - vxctl, 135
 - vxedit, 128
 - vxmake, 136
 - vxmend, 138
 - vxplex, 137
 - vxprint, 138
 - vxsd, 138
 - vxstat, 140
 - vxvol, 143

V

- VM disks, 31
 - definition, 31
- volume kernel states, 363
 - DETACHED, 363
 - DISABLED, 363
 - ENABLED, 363
- Volume Manager, 56, 58, 84
 - and operating system, 60
 - daemons, 71
 - layouts, 62
 - objects, 58
- Volume Manager graphical user interface, 150
- volume reconfiguration, 302
- volume resynchronization, 110
- volume states, 362
 - ACTIVE, 362, 363
 - CLEAN, 362, 363
 - EMPTY, 362, 363
 - SYNC, 362, 363
- volumes, 34
 - adding a mirror, 248
 - and plexes, 35
 - backing up, 255
 - cleanup, 342
 - concatenated, creating, 238
 - creating, 131, 132, 133, 134, 237
 - concatenated, 238
 - definition, 29, 34
 - displaying configuration, 254
 - increasing size, 243
 - kernel state, 143
 - layout, 364
 - managing, 131, 132, 133, 134
 - mirroring, 248
 - mirroring all existing volumes, 249

- mirroring on disk, 249, 291
- moving from disk, 258, 292
- operations, 140
- RAID-5
 - creating, 240
- read policy, 245
- recovery, 257
- removing, 247
- removing a RAID-5 log from, 261
- spanned
 - creating, 238
- starting, 241, 242
- stopping, 241, 242
- striped
 - creating, 239
- vxassist, 81, 94, 95, 131, 132, 133, 134, 227, 243, 248, 287, 288, 357
 - backup, 94, 287
 - creating volumes, 64
 - defaults, 133
 - description of, 131
 - growby, 243
 - growto, 243
 - shrinkby, 243
 - shrinkto, 243
- vxassist addlog, 260
- vxassist growby, 244
- vxassist growto, 243
- vxassist make, 201, 239
- vxassist snapshot, 256
- vxassist snapstart, 255
- vxclust, 318
- vxconfigd, 71, 135, 319
- vxctl, 322
 - description of, 135
- vxdg, 183, 211, 223, 224, 226, 323
 - moving disk groups, 223, 224
 - removing disk groups, 228
- vxdg free, 87
- vxdg list, 231
- vxdisk, 150, 184, 327
 - rm, 184
- vxdisk list, 202
- vxdiskadd, 150, 157, 160, 202
- vxdiskadm, 150, 157, 184, 211, 356
 - replacing disks, 356
 - starting, 205
- vxedit, 247, 253, 266, 268, 276, 286
 - description of, 128
 - removing subdisks, 276
- vxedit rename, 200
- vxedit set, 201

- vxinfo, 356
- vxiod, 71, 72
- vxmake, 136, 248, 264, 265, 275, 282, 366, 367
 - associating mirrors, 265
 - associating subdisks, 282
 - creating mirrors, 264
 - creating subdisks, 275
 - description of, 136
- vxmend, 138, 241, 242, 257, 270, 271
- vxplex, 137, 248, 251, 252, 265, 270, 271, 273
 - copying mirrors, 274
 - description of, 137
 - dissociating mirrors, 252, 266
 - moving mirrors, 273
- vxprint, 86, 89, 138, 175, 254, 267, 354
 - description of, 138
 - displaying subdisks, 89
 - listing mirrors, 267
- vxreattach, 335
- vxrecover, 176, 329, 355
 - moving disk groups, 223, 224
- vxrelocd, 103, 105, 139, 349
 - modifying, 138, 349
- vxsd, 137, 277, 280, 282, 285
 - associating log subdisks, 283
 - associating subdisks, 282
 - description of, 138
 - joining subdisks, 281
 - moving subdisks, 277
 - splitting subdisks, 280
- VxSmartSync, 126
- vxstat, 140, 175, 329, 355, 396, 398
 - description of, 140
- vxtrace, 140, 396, 398
- VxVM, 56, 84
- vxvol, 143, 243, 244, 271, 356
 - description of, 143

W

- writes
 - full-stripe, 384
 - read-modify, 382
 - reconstruct, 385