Digital UNIX

Logical Storage Manager

Order Number: AA-Q3NCE-TE

March 1996

Product Version: Digital UNIX Version 4.0 or higher

This manual provides an overview of the Digital UNIX Logical Storage Manager (LSM) concepts and describes how to configure and maintain systems under the control of LSM. It also describes advanced volume management concepts including striping, mirroring, and disk group management. Reference information is included to help you perform common disk management operations using the LSM interfaces.

Digital Equipment Corporation
Maynard, Massachusetts

Contents

About This Guide

Part 1: Introduction and Setup

1 Introduction to the Logical Storage Manager

2 LSM Configurations

3 Setting Up LSM

4 Encapsulating Existing User Data to LSM Volumes

5 LSM Root and Swap Disk Mirroring

Part 2: Disks and Volumes, Command Line Interface

6 Disk and Disk Group Operations

7 Volume and File System Operations

8 Advanced Volume Operations

Part 3: Menu Interface

9 Menu Interface (voldiskadm)

Part 4: Graphical Interface

10 Introduction to the Visual Administrator (dxlsm)

11 Visual Administrator Menus

12 Using the Visual Administrator (dxlsm)

14 Recovering from Errors

15 LSM Performance Management

Appendixes: Visual Administrator Reference, Error Messages, and Examples

A Visual Administrator Reference

B Logical Storage Manager Error Messages

C LSM Command Examples

Examples

Figures

Tables

About This Guide

The Digital UNIX® Logical Storage Manager (LSM) provides high data availability for disk storage devices on Alpha systems. This guide explains LSM concepts and the system administrator tools and techniques you need to implement and manage LSM on Alpha systems.

LSM replaces the Logical Volume Manager (LVM). For LVM users who have not migrated to LSM yet, see Chapter 4 for information about the LSM encapsulation tools to help you migrate LVM volumes, UNIX style partitions, and AdvFS storage domains to LSM.

Audience

This guide is intended for system administrators who need to configure and maintain systems under the control of LSM. This guide assumes that the reader has a:

- Working knowledge of the Digital UNIX operating system
- Basic understanding of system administration
- Basic understanding of volume management
- Basic understanding of disk structures

New and Changed Features

This manual has been revised to reflect all the LSM changes that are part of the current release.

This version of the manual also provides the following new features:

- New and revised diagrams and tables to define and introduce LSM concepts
- Additional instructions for LSM administrative tasks
- Additional troubleshooting and problem-solving information
- Many new and expanded examples in Appendix C
- A revised and expanded index
- Reorganization into 15 chapters and 5 main parts

Organization

This manual consists of fifteen chapters (organized into five main parts) and three appendixes:

Part 1	Provides introductory and setup information (Chapters 1 through 5).
Chapter 1	Contains an overview of the Logical Storage Manager, including a discussion of LSM fundamentals, components of an LSM configuration, LSM interfaces, and the LSM system architecture.
Chapter 2	Describes LSM configurations, including concatenated disks, mirroring, and striping.
Chapter 3	Provides information to help you start LSM and perform initial LSM set up.
Chapter 4	Describes how to encapsulate existing user data.
Chapter 5	Describes how to encapsulate the partitions used for the root file system and swap areas to LSM volumes and how to mirror the volumes.
Part 2	Provides disk and volume management information, using the command line interface (Chapters 6 through 8).
Chapter 6	Describes how to manage disks under LSM.
Chapter 7	Describes how to manage volumes and file systems.
Chapter 8	Describes how to manage subdisks and plexes.
Part 3	Provides information on the LSM Support Operations menu interface, `voldiskadm` (Chapter 9).
Chapter 9	Describes how to perform disk, disk group, and volume operations using the LSM Support Operations menu interface (`voldiskadm`).
Part 4	Provides task-oriented information on the graphical interface, `dxlsm` (Chapters 10 through 12).
Chapter 10	Gives an overview of the Visual Administrator graphical user interface (`dxlsm`).
Chapter 11	Describes the `dxlsm` menus.
Chapter 12	Describes how to manage disks and volumes using `dxlsm`.
Part 5	Provides performance tuning, troubleshooting, and error recovery information (Chapters 13 through 15).
Chapter 13	Discusses advanced LSM volume management concepts.
Chapter 14	Gives troubleshooting information and describes recovery procedures such as how to replace a bad disk.

Chapter 15	Suggests performance priorities and guidelines for use with LSM.
Appendix A	Provides comprehensive reference information describing the menus, windows, icons, and other features of the Visual Administrator (`dxlsm`) interface.
Appendix B	Lists the error messages generated by LSM, explains their meanings, and suggests actions for the system administrator to take if these messages occur.
Appendix C	Provides examples of LSM commands and system output that show how to set up and manage an LSM configuration.

Related Documents

The following documents provide information related to the Logical Storage Manager:

- The Digital UNIX *Installation Guide* describes how to install the Logical Storage Manager.
- The Digital UNIX *Release Notes* describe supported LSM features and products, and known problems and fixes.

The printed version of the Digital UNIX documentation set is color coded to help specific audiences quickly find the books that meet their needs. (You can order the printed documentation from Digital.) This color coding is reinforced with the use of an icon on the spines of books. The following list describes this convention:

Audience	Icon	Color Code
General users	G	Blue
System and network administrators	S	Red
Programmers	P	Purple
Device driver writers	D	Orange
Reference page users	R	Green

Some books in the documentation set help meet the needs of several audiences. For example, the information in some system books is also used by programmers. Keep this in mind when searching for information on specific topics.

The *Documentation Overview, Glossary, and Master Index* provides information on all of the books in the Digital UNIX documentation set.

Reader’s Comments

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

You can send your comments in the following ways:

- Fax: 603-881-0120 Attn: UEG Publications, ZK03-3/Y32
- Internet electronic mail: `readers_comment@zk3.dec.com`

  A Reader’s Comment form is located on line in the following location:

  `/usr/doc/readers_comment.txt`

- Mail:

  Digital Equipment Corporation
  UEG Publications Manager
  ZK03-3/Y32
  110 Spit Brook Road
  Nashua, NH 03062-9987

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

Please include the following information along with your comments:

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

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

Conventions

The following conventions are used in this manual:

`%` `$`	A percent sign represents the C shell system prompt. A dollar sign represents the system prompt for the Bourne and Korn shells.

#	A number sign represents the superuser prompt.
% **cat**	Boldface type in interactive examples indicates typed user input.
file	Italic (slanted) type indicates variable values, placeholders, and function argument names.
[|] **{ | }**	In syntax definitions, brackets indicate items that are optional and braces indicate items that are required. Vertical bars separating items inside brackets or braces indicate that you choose one item from among those listed.
. . .	In syntax definitions, a horizontal ellipsis indicates that the preceding item can be repeated one or more times.
cat(1)	A cross-reference to a reference page includes the appropriate section number in parentheses. For example, cat(1) indicates that you can find information on the cat command in Section 1 of the reference pages.
Return	In an example, a key name enclosed in a box indicates that you press that key.
Ctrl/*x*	This symbol indicates that you hold down the first named key while pressing the key or mouse button that follows the slash. In examples, this key combination is enclosed in a box (for example, Ctrl/C).
mouse	The term mouse refers to any pointing device, such as a mouse, a puck, or a stylus.
MB1, MB2, MB3	On a three-button mouse, MB1 indicates the left mouse button. MB2 indicates the middle mouse button, and MB3 indicates the right mouse button. (You can redefine the buttons by using platform-specific window management facilities.)
Menu → Option → Submenu Option	The right arrow indicates an abbreviated instruction for choosing a menu option or submenu option. The following example means pull down the Modify menu, move the pointer to pull down the Image submenu, and choose the Clear option: Choose Modify → Image → Clear

Part 1: Introduction and Setup

Introduction to the Logical Storage Manager 1

This chapter introduces the Digital UNIX Logical Storage Manager (LSM), its features and capabilities, concepts, and terminology. The `volintro`(8) reference page also provides a quick reference of LSM terminology and command usage.

1.1 LSM Overview

The Logical Storage Manager (LSM) is an integrated, host-based disk storage management tool that protects against data loss, improves disk input/output (I/O) performance, and customizes the disk configuration. System administrators use LSM to perform disk management functions without disrupting users or applications accessing data on those disks.

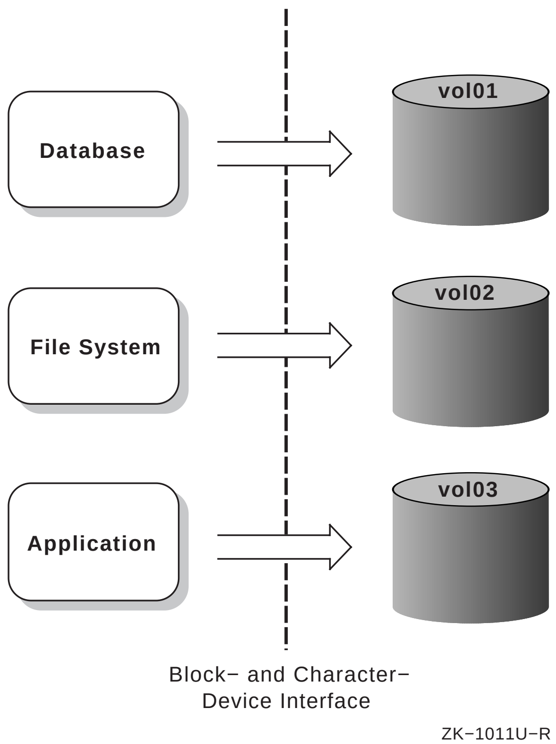
The LSM software is included as optional subsets in the base Digital UNIX system. When building the kernel, there are specific kernel options that need to be selected to configure LSM. All Digital UNIX systems can use the basic LSM functions, but additional functions such as mirroring, striping, and the graphical administration tool require a separate LSM license.

1.2 LSM Fundamentals

LSM builds virtual disks, called volumes, on top of UNIX system disks. A *volume* is a Digital UNIX special device that contains data used by a UNIX file system, a database, or other application. LSM transparently places a volume between a physical disk and an application which then operates on the volume rather than on the physical disk. A file system, for instance, is created on the LSM volume rather than on a physical disk.

Figure 1-1 shows how disk storage is handled in systems that use LSM.

Figure 1-1: Disk Storage Management with LSM



In general, disk storage management often requires that for each file system or database created, you must be able to do the following:

- Allocate and reallocate disk space as space requirements change
- Address the space allocated for a particular file system or database
- Access data through an application programming interface

All of these requirements can be done more easily when you use LSM. Table 1-1 compares disk storage management requirements for systems running with and without LSM.

Table 1-1: Disk Storage Management With and Without LSM

Requirement	Without LSM	With LSM
Space Allocation	UNIX disks are divided into *partitions*. A partition is defined by its start address on the physical disk and its length. The administrator must partition the disks according to the needs of the users on the system. Partitions cannot be moved or extended in size once the partition is in use.	LSM obtains space for a file system or raw database by creating an LSM volume of the appropriate size. A volume is built from one or more areas of disk space (also called subdisks) located on one or more physical disks. This makes it possible to extend volumes by adding disk space that is not contiguous with the space already allocated, and to create volumes that exceed the size of a physical disk.
Addressing	A UNIX partition is addressed through a physical address, generally referred to as the *device name* or *devname*. Reconfiguring disks (for example, moving a disk to a new controller) requires a change of the addresses through which the partitions are accessed because the disk's unit number has changed. The administrator must also manually change all references to the partitions on the reconfigured disk devices.	LSM volumes are addressed using a *volume name* that is independent of the manner in which the volume is mapped onto physical disks. You establish a symbolic *disk name* or *disk media name* to refer to a disk that is managed by LSM (for example: `disk01`). This makes it possible to easily readjust LSM volume and space allocation in case disks are moved in the configuration without affecting the application.
Data Access	Data storage and retrieval on a UNIX partition is achieved through the standard block- and character-device interfaces using the physical-device address. In addition, because the partitioning of disks cannot be changed easily, it is difficult for the administrator to ensure that data is placed on the available disk drives for optimal access and performance.	LSM volumes can be accessed through the standard block- and character-device interfaces, using names that are independent of the physical storage addresses used by the volume. In addition, because you can change LSM volume configurations on line without interrupting user access to the data, you can dynamically change data placement for optimal access and performance.

1.3 LSM Features

Table 1-2 summarizes the LSM features.

Table 1-2: LSM Features and Benefits

Feature	Benefit
Manages disk administration	Frees you from the task of partitioning disks and maintaining disk-space administration. However, LSM allows you to keep control over disk partitioning and space allocation, if desired.
Allows transparent disk configuration changes	Allows you to change the disk configuration without rebooting or otherwise interrupting users. Also allows routine administrative tasks, such as file system backup, while the system is in active use.
Stores large file systems	Enables multiple physical disks to be combined to form a single, larger logical volume. This capability, called *concatenation*, removes limitations imposed by the actual physical properties of individual disk sizes, by combining the storage potential of several devices. Note that disk concatenation is available on all systems, including those that do not have the LSM software license.
Eases system management	Simplifies the management of disk configurations by providing convenient interfaces and utilities to add, move, replace, and remove disks.
Protects against data loss	Protects against data loss due to hardware malfunction by creating a *mirror* (duplicate) image of important file systems and databases.
Increases disk performance	Improves disk I/O performance through the use of *striping*, which is the interleaving of data within the volume across several physical disks.
Provides recovery from boot disk failure	Allows you to mirror the root file system and swap partition. By duplicating the disks that are critical to booting, LSM ensures that no single disk failure will leave your system unusable.

1.4 Hardware and Software Requirements

The following sections describe the hardware and software requirements, licensing, and configuration limitations for LSM.

1.4.1 Hardware Requirements

LSM does not depend on specific hardware in order to operate. All functions can be performed on any supported Alpha computer running Digital UNIX, Version 3.2 or higher. There are no restrictions on the devices supported beyond the valid configurations defined in the Digital UNIX Software Product Descriptions.

All Small Computer Systems Interface (SCSI) and DIGITAL Storage Architecture (DSA) disks supported by this version of Digital UNIX are supported by LSM. SCSI redundant arrays of independent disks (RAID) hardware devices are supported as standard disks, with each RAID device-logical unit viewed as a physical disk.

1.4.2 Software Requirements

LSM has the following software requirements:

- The LSM software is supplied as part of the base operating system. Use the version of LSM that is supplied with the Digital UNIX release that you are using.
- The LSM Visual Administrator is a Motif-based application that requires you have the Basic X Environment subset installed on the system.

1.4.3 Licensing Requirements

The LSM software is furnished under the licensing provisions of the Digital Equipment Corporation Standard Terms and Conditions. However, note that the base Digital UNIX license allows you to use the LSM concatenation and spanning feature. You do not need an LSM software license to include multiple physical disks within a single LSM volume.

To use LSM advanced features, such as mirroring, striping, and the Visual Administrator (`dxlsm`), you must have an LSM license. License units for LSM are allocated on an unlimited system use basis.

Refer to the manual *Software License Management* in the Digital UNIX documentation set for more information about the Digital UNIX License Management Facility (LMF).

1.4.4 Configuration Limitations

The maximum configuration supported by the Digital UNIX Logical Storage Manager is defined as follows:

- Maximum of 4093 volumes
- Maximum of 8 plexes per volume

  A plex is an identical copy of data. Volumes that are not mirrored have one plex, while mirrored volumes can have anywhere from two to eight plexes.
- Maximum of 4096 subdisks per plex.

  A subdisk is a basic unit of disk space allocation for a plex.

Refer to the LSM Software Product Description (SPD) for the maximum number of disks and the maximum volume size.

See Section 3.4 for information on changing the default configuration limits.

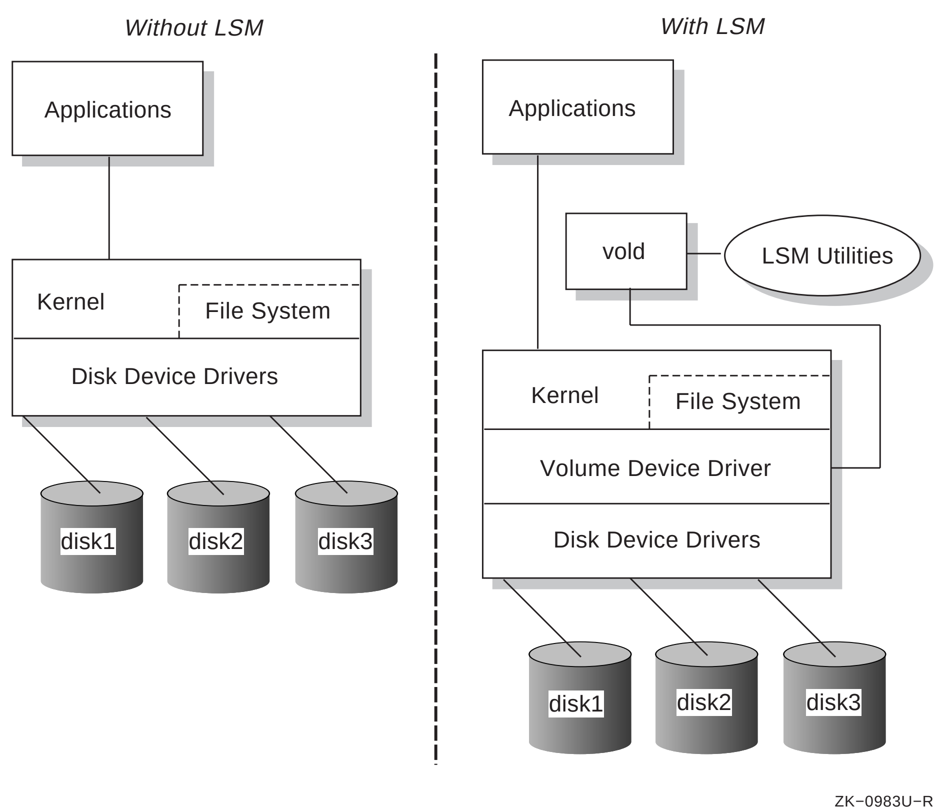
1.5 LSM and System Architecture

Architecturally, the LSM device driver fits between the file systems and the disk device drivers. An LSM-built kernel includes volume device drivers that provide a level of abstraction between the physical disks and the file systems or third-party databases. The file systems and databases are placed on LSM volumes and perform I/O requests to an LSM volume in the same way that they perform I/O requests to any other disk driver.

Once an LSM volume is defined and configured, the file systems and databases issue I/O requests directly to the LSM volume, not to the device drivers.

The system architecture in Figure 1-2 shows the relationships between the kernel, file systems and application databases, and the device drivers for systems with and without LSM installed.

Figure 1-2: LSM Software Architecture



1.5.1 Volume Device Driver and Volume Daemons

The central components of the LSM architecture, the volume device driver and the volume configuration daemon (`vold`), are shown in Figure 1-2 and described in the following list. The list also describes the volume extended I/O daemon (`voliod`), because this process is started immediately after the initial installation of the `vold` daemon.

- Volume Device Driver

  The LSM *volume device driver* is an installable driver that maps the logical configuration of the system to the physical disk configuration. Although the volume device driver resides between the disk device and the applications, it maps the configuration transparently to the file systems, databases, and applications above it. Thus, applications do not need to be changed in order to access data on LSM volumes.

The volume device driver implements striping or concatenation, read and write error handling, and mirroring in LSM configurations.

The volume device driver supports the devices described in Table 1-3 in LSM configurations.

Table 1-3: Devices Supported by the Volume Device Driver

Device	Description
Volume	LSM creates a volume device-special file for every virtual disk (volume) defined by the system administrator (for example, `/dev/vol/...`).
Plex	Plex devices are used internally by LSM for resynchronizing mirrors and other special operations. LSM creates a character device-special file for every plex in `/dev/plex`.
Additional devices	Additional devices are used to communicate between LSM utilities (`volconfig`, `volevent`, `volinfo`, `voliod`) and the kernel.

- Volume Configuration Daemon (`vold`)

  The volume configuration daemon, `vold`, is responsible for the interface of the LSM utilities to the kernel. All LSM configuration database changes are centralized through the `vold` daemon, including:

  – Creation and deletion of LSM objects

  – LSM configuration changes due to system administrator interaction

  – LSM configuration changes due to asynchronous error events

  The `vold` daemon packages the LSM configuration change into a transaction and passes it to the LSM `volspec` driver to record the actual change. Because these changes are performed by the daemon and not the kernel, the robustness of LSM is increased.

- Volume Extended I/O Daemon (`voliod`)

  The volume extended I/O daemon, `voliod`, does the following:

  – Allows for some extended I/O operations without blocking calling processes

  – Allows the volume device driver to schedule writes to volumes that have block-change logging enabled (described in Section 13.1.4)

  – Starts the LSM kernel error-log daemons to handle errors and recovery for LSM volumes

If there are volumes with block-change logging enabled, then there will be multiple `voliod` processes running on the system. The `voliod` processes are started by the `vold` daemon and are killed by the kernel when these processes are no longer needed. Rebooting after your initial installation should start the `voliod` daemon.

For more detailed information about these daemons, refer to the `vold`(8) and `voliod`(8) reference pages.

1.5.2 LSM Objects

LSM consists of physical disk devices, logical entities (also called *objects*) and the mappings that connect the physical and logical objects. LSM logically binds together the physical disk devices into a logical LSM volume that represents the disks as a single virtual device to applications and users.

LSM organizes and optimizes disk usage and guards against media failures using the following objects:

- Volumes
- Plexes
- Subdisks

Each object has a dependent relationship on the next-higher object, with subdisks being the lowest level objects in the structure, and volumes the highest level. LSM maintains a configuration database that describes the objects in the LSM configuration, and implements utilities to manage the configuration database. Multiple mirrors, striping, and concatenation are additional techniques you can perform with the LSM objects to further enhance the capabilities of LSM.

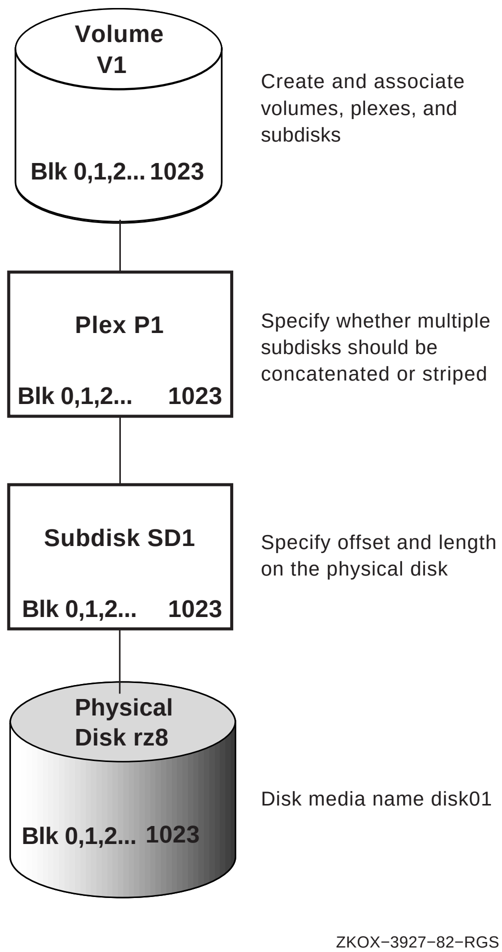
Table 1-4 describes the LSM objects.

Table 1-4: LSM Objects

Object	Description
Volume	Represents an addressable range of disk blocks used by applications, file systems, or databases. A volume is a virtual disk device that looks to applications and file systems like a regular disk-partition device. In fact, volumes are logical devices that appear as devices in the `/dev` directory. The volumes are labeled `fsgen` or `gen` according to their usage and content type. Each volume can be composed of from one to eight plexes (two or more plexes mirror the data within the volume). Due to its virtual nature, a volume is not restricted to a particular disk or a specific area thereof. The configuration of a volume can be changed (using LSM utilities) without causing disruption to applications or file systems using that volume.
Plex	A collection of one or more subdisks that represent specific portions of physical disks. When more than one plex is present, each plex is a replica of the volume; the data contained at any given point on each is identical (although the subdisk arrangement may differ). Plexes can have a striped or concatenated organization.
Subdisk	A logical representation of a set of contiguous disk blocks on a physical disk. Subdisks are associated with plexes to form volumes. Subdisks are the basic components of LSM volumes; subdisks form a bridge between physical disks and virtual volumes.
Disk	A collection of nonvolatile, read/write data blocks that are indexed and can be quickly and randomly accessed. LSM supports standard disk devices, including SCSI and DSA disks. Each disk used by LSM is given two identifiers: a disk access name and an administrative name.
Disk Group	A collection of disks that share the same LSM configuration database. The root disk group, `rootdg`, is a special private disk group that always exists.

Figure 1-3 shows the relationship of volumes, plexes, subdisks, and physical disks for a simple volume where 1024 blocks on a volume map to a physical disk. In this illustration, the mapping is a straight pass-through to the physical disk.

Figure 1-3: LSM Object Relationships



1.6 LSM Disks

You can use any standard disk device, for example SCSI or DSA disks, with LSM. Standard disk devices are those that can be used with Digital UNIX utilities, such as `disklabel` and `newfs`.

Section 1.6.1 and Section 1.6.2 describe the characteristics of standard devices, and how these devices are named for use with LSM.

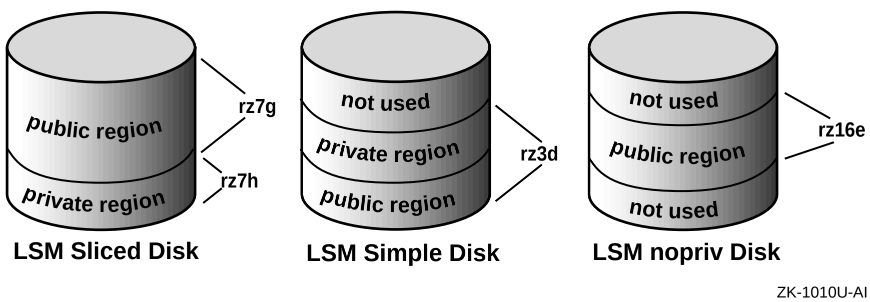
1.6.1 Types of LSM Disks

An LSM disk typically uses two regions on each physical disk. These regions have the following characteristics:

- A small region, called the private region, in which LSM keeps its disk media label and a configuration database
- A large region, called the public region, that forms the storage space for building subdisks

Figure 1-4 shows the private and public regions in LSM *simple* and *sliced* disks. The third disk, an LSM *nopriv* disk, does not contain a private region. All of these types of disks can be added into an LSM disk group.

Figure 1-4: Types of LSM Disks



The disks shown in Figure 1-4 have the following characteristics:

- Simple disks have both public and private regions in the same partition (`rz3g`).
- Sliced disks typically are used when the entire disk (rz7) will be under LSM control. The disk label for a sliced disk contains information that identifies the private (`rz7h`) and the public (`rz7g`) regions.
- Nopriv disks are used for encapsulating existing data. Because a nopriv disk has no private region containing LSM configuration information, it must be added to an existing disk group that includes a simple or a sliced disk.

LSM configuration databases are stored in the private regions of simple and sliced disks. For purposes of availability and performance, each simple or sliced disk can contain 0, 1 or 2 copies of its configuration database. See Section 3.3.2.2 for details.

The public regions of the LSM disks collectively form the storage space for application use.

Note

To add a new disk with no configuration database into a disk group, use a simple or sliced disk, with the `nconfig` attribute set to 0. Do not initialize a new disk as a nopriv disk– this disk type is appropriate only for encapsulation of existing data.

1.6.2 Naming LSM Disks

When you perform disk operations, you should understand the disk-naming conventions for a *disk access name* and *disk media name.* This is because disk access names and disk media names are treated internally as two types of LSM disk objects. Some operations require that you specify the disk access name, while others require the disk media name. The following list describes these disk naming conventions:

- Disk access name (also referred to as *devname* or *device name*)

  The device name or address used to access a physical disk. A disk access name has the following form:

  dd[*l*] *n*[*nnn*] [*p*]

  The elements in the disk access name are described in the following table:

Element	Description
`dd`	A two-character device mnemonic that shows the disk type. Use `ra` for DSA disks, `rz` for SCSI disks, and `re` for Redundant Arrays of Independent Disks (RAID) devices.
[l]	The SCSI logical unit number (LUN), in the range from a to h, to correspond to LUNs 0 through 7. This argument is used for HSZ hardware RAID devices.
n[nnn]	The disk unit number ranging from 1 to 4 digits.
[p]	The partition letter, in the range from a to h, to correspond to partitions 0 through 7. This argument is optional.

  For example, `rz` in the device name `rz3` represents a pseudonym for a SCSI disk, and `rzb10h` represents a disk access name for a SCSI disk having a LUN of one, which applies to Digital SCSI RAID devices.

  For a simple disk or a nopriv disk, you must specify a partition letter. For example, `rz3d`. For a sliced disk, you must specify a physical drive that does not have a partition letter (for example, `rz3`). The proper full pathname of the `d` partition on this sliced device is `/dev/rz3d`. However, for easier reading, this document often lists only the disk-access name and `/dev` is assumed. Also, note that you do not specify `/dev` in front of the device name when using LSM commands.

- Disk media name (also referred to as the *disk name*)

  An administrative name for the disk, such as `disk01`. If you do not assign a disk media name, it defaults to `disk`*`nn`*, where *nn* is a sequence number within the disk group.

1.7 LSM Disk Groups

You can organize a collection of physical disks that share a common configuration or function into disk groups. LSM volumes are created within a disk group and are restricted to using disks within that disk group.

Disk groups can be used to simplify management and provide data availability. For example:

- On a system with many disks, you might want to divide disk usage into a few disk groups based on function. This would reduce the size of the LSM configuration database for each disk group, as well as reduce the amount of overhead incurred in configuration changes.
- If a system will be unavailable for a prolonged amount of time due to a hardware failure, the physical disks in a disk group can be moved for use on another system. This is possible because each disk group has a self-describing LSM configuration database.

All systems with LSM installed have the `rootdg` disk group. By default, operations are directed to this disk group. Most systems do not need to use more than one disk group.

Note

> You do not have to add disks to disk groups when a disk is initialized; disks can be initialized and kept on standby as replacements for failed disks. A disk that is initialized but not added to a disk group can be used to immediately replace a failing disk in any disk group.

Each disk group maintains an LSM configuration database that contains detailed records and attributes about the existing disks, volumes, plexes, and subdisks in the disk group.

1.7.1 LSM Configuration Databases

The LSM configuration database contains records describing all the objects (volumes, plexes, subdisks, disk media names, and disk access names) being used in a disk group.

Typically, one or two copies of the LSM configuration database are located in the private region (illustrated in Figure 1-4) of each disk within a disk group. LSM maintains multiple identical copies of the configuration database in case of full or partial disk failure.

The contents of the `rootdg` configuration database are slightly different. The difference between a `rootdg` configuration database and an ordinary LSM configuration database is that the `rootdg` configuration database contains records for disks outside of the `rootdg` disk group in addition to the ordinary disk-group configuration information. Specifically, a `rootdg` configuration includes disk-access records that define all disks on the system.

The `volboot` file is used by the LSM volume daemon, `vold`, during startup to locate copies of the `rootdg` configuration database. This file contains a list of the disks that have configuration copies in standard locations. The `volboot` file is located in `/etc/vol`.

1.7.2 Moving and Replacing LSM Disks in a Disk Group

When a disk is added to a disk group it is given a disk media name, such as `disk02`. This name relates directly to the physical disk. LSM uses this naming convention (described in Section 1.6.2) because it makes the disk independent of the manner in which the volume is mapped onto physical disks. 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).

1.8 LSM Interfaces

LSM provides three different methods to manage LSM disks: a graphical user interface, a menu interface, and a command-line interface. You can use any of these interfaces (or a combination of the interfaces) to change volume size, add plexes, and perform backups or other administrative tasks. Table 1-5 describes these LSM interfaces.

Table 1-5: LSM Administration Interfaces

Interface	Type	Description
Visual Administrator (dxlsm)	Graphical	Uses windows, icons, and menus to manage LSM volumes. The dxlsm interface requires a workstation (bit-mapped display) and the Basic X Environment subset installed to provide its icon and menu-driven approach to volume management. This simple-to-use interface translates mouse-based icon operations into LSM commands. **Note** The Visual Administrator (dxlsm) interface requires the LSM software license.
Support Operations (voldiskadm)	Menu	Provides a menu interface to manage LSM volumes. Each entry in the main menu leads you through a particular operation by providing you with information and asking you questions. Default answers are provided for many questions so that common answers can be selected easily. This is a character-cell interface that does not require a workstation for operation.
Command Line	Command	Provides two approaches to LSM administration. With the top-down approach, you use the LSM volassist command to automatically build the underlying LSM objects. With the bottom-up approach, you use several commands (including volmake, volplex, volume, and volsd) to build individual objects in order to customize the construction of an LSM volume.

Once a disk is under the control of LSM, all system administrative tasks relating to that disk must be performed using LSM utilities and commands.

The LSM interfaces can be used interchangeably. LSM objects created by one interface are fully interoperable and compatible with objects created by the other interfaces.

1.8.1 Top-Down vs. Bottom-Up Storage Management

As described in Table 1-5, the command-line interface provides you with both a top-down and a bottom-up approach to LSM storage management.

With the top-down approach, you use the `volassist` utility to automatically build the underlying LSM objects. With the bottom-up approach, you use a combination of low-level commands to build individual objects to customize the construction of LSM volumes.

These two approaches are interchangeable. You can create one volume with one approach, then create another volume using the other approach, and modify either volume with either approach.

Most administrators prefer the top-down approach and find it adequate for most LSM activities and operations. The bottom-up approach provides the most control for defining and manipulating LSM objects, as well as for recovering from unusual errors or problems.

1.8.1.1 Top-Down Approach

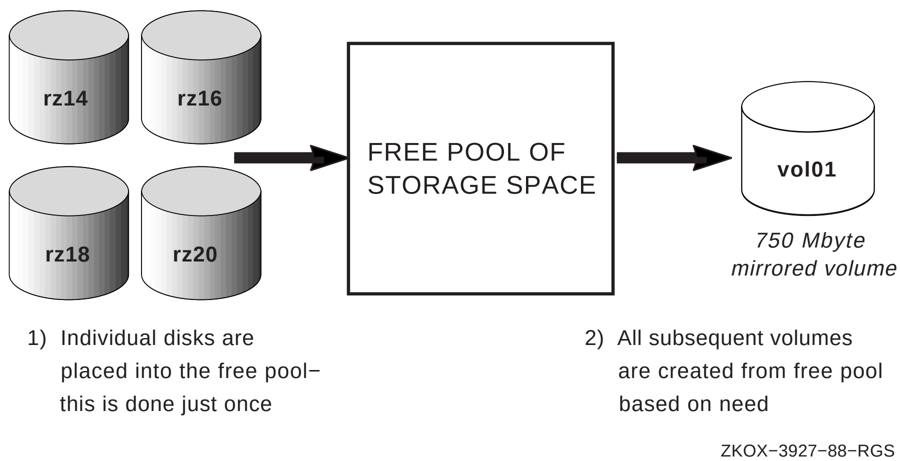
The top-down approach for managing storage space involves placing disks into one large pool of free storage space. When you need storage space, you use the `volassist` command to specify to LSM what you need, and LSM allocates the space from this free pool. Based on your needs (for example, striped and mirrored volumes), LSM automatically allocates the storage from different physical disks to properly satisfy the volume configuration requirements.

The following example of the `volassist` command creates a 750MB mirrored volume:

```
# volassist make vol01 750mb mirror=true
```

Figure 1-5 illustrates the two-step process of creating a pool of storage space and using it to create volumes as they are needed.

Figure 1-5: Top-Down Administration with LSM



When you specify to LSM what is needed (for example, a 750MB, mirrored volume), LSM does the following:

- Manages the storage space and determines which disks to use
- Determines how much space is free within a disk's partition
- Creates the necessary subdisks, plexes, and volume objects
- Starts the volume

The top-down approach enables you to provide loose requirements on your volume needs. However, if necessary, you can be very specific on the constraints and attributes of the volume by providing additional parameters and options to `volassist`. Refer to the `volassist`(8) reference page for a full list of options and constraints that you can use when creating or managing LSM volumes.

1.8.1.2 Bottom-Up Approach

The bottom-up approach is used when you want to manage the free disk space yourself or you require additional control of the placement and definition of the subdisk, plex, and volume objects. You must ensure that you define and properly configure the volume's subdisks on different physical disks when using the bottom-up approach for a mirrored or striped volume.

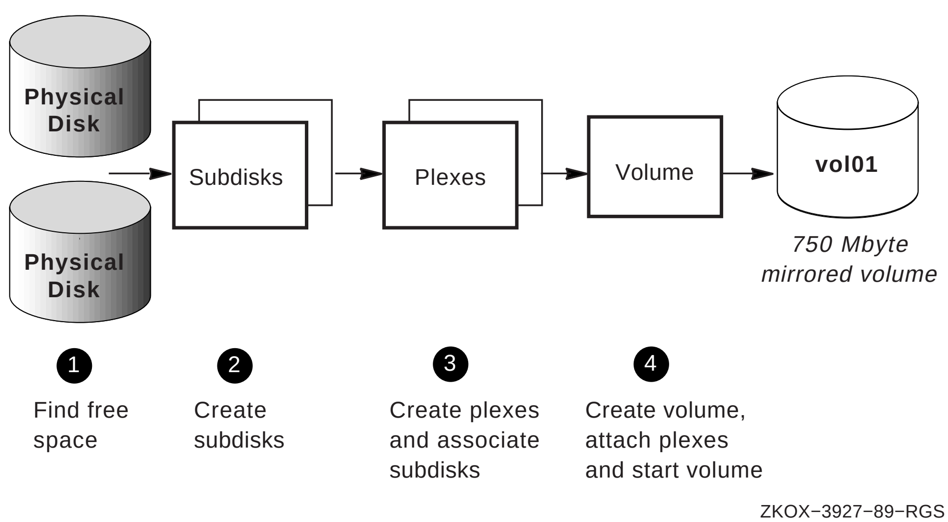
Use the `volmake` command to create subdisks, plexes, and volumes with the bottom-up administration approach. For example, to create a 750MB mirrored volume with `volmake`, enter the following commands:

```
# volmake sd rz11h-01 rz11h,0,1536000
# volmake plex vol01-02 sd=rz11h-01
# volmake sd rz9g-05 rz9g,8192,1536000
# volmake plex vol01-01 sd=rz9g-05
# volmake vol vol01 usetype=fsgen plex=vol01-01,vol01-02
# volume start vol01
```

As shown in Figure 1-6, the bottom-up approach for managing storage space involves the following steps:

1. Finding free space on the LSM disks
2. Using the free space to create the subdisks
3. Creating the plexes and associating subdisks
4. Creating the volume, attaching plexes, and starting the volume

Figure 1-6: Bottom-Up Administration with LSM



1.8.2 LSM Command Hierarchy

Table 1-6 lists the top-down commands that you can use to manage storage volumes on LSM. Note that although they have different interfaces, the `dxlsm` and `voldiskadm` utilities are also considered top-down commands. Most LSM commands can be used by privileged users only.

Table 1-6: Top-Down LSM Commands

Command	Description
volsetup	Initialize LSM by creating the `rootdg` diskgroup
voldiskadd	Add a disk for use with the Logical Storage Manager
dxlsm	Invoke a graphical utility for common LSM operations
voldiskadm	Invoke a menu-based utility for common LSM operations
volassist	Create, mirror, backup, grow, shrink, and move volumes
volevac	Evacuate all volumes from a disk
volencap	Encapsulate partitions (place existing data under LSM control)
volrecover	Recover plexes and volumes after disk replacement
volrootmir	Mirror the root and swap volumes
volmirror	Mirror all volumes on a specified disk
volwatch	Monitor the Logical Storage Manager for failure events

Table 1-7 lists the bottom-up commands you can use to manage storage volumes on LSM.

Table 1-7: Bottom-Up LSM Commands

Command	Description
volinstall	Set up LSM environment after LSM installation (pre-LSM Version 1.2)
voldisksetup	Set up a disk for use with the Logical Storage Manager
voldisk	Define and manage LSM disks
voldg	Manage LSM disk groups
volmake	Create LSM configuration records
volsd	Perform LSM operations on subdisks
volplex	Perform LSM operations on plexes
volume	Perform LSM operations on volumes
volprint	Display records from the LSM configuration
voledit	Create, remove, and modify LSM records
volmend	Mend simple problems in configuration records
voldctl	Control the volume configuration daemon and volboot information
volinfo	Print accessibility and usability of volumes
volstat	Invoke the LSM statistics management utility
volnotify	Display LSM configuration events
voltrace	Trace operations on volumes

1.9 Accessing LSM Volumes for I/O

Once you create LSM volumes using one of the LSM interfaces, users and applications can access LSM volumes in the same way that they access any disk device:

- Block-special files for LSM volumes are located in:

  `/dev/vol/`*`diskgroupname`*

- Character-special files are located in:

  `/dev/rvol/`*`diskgroupname`*

The variable *`diskgroupname`* refers to the disk group name that contains the volume. Note that volumes in the `rootdg` disk group are located in the the `/dev/vol/` and the `/dev/rvol/` directories too.

To create a new UNIX file system (UFS) on an LSM volume, use the `newfs` command with a disk type argument that specifies any known disk type. The disk type is used to provide the sector and track size information for the `newfs` command. For example, to create a new UFS on the LSM volume vol01, enter the following commands:

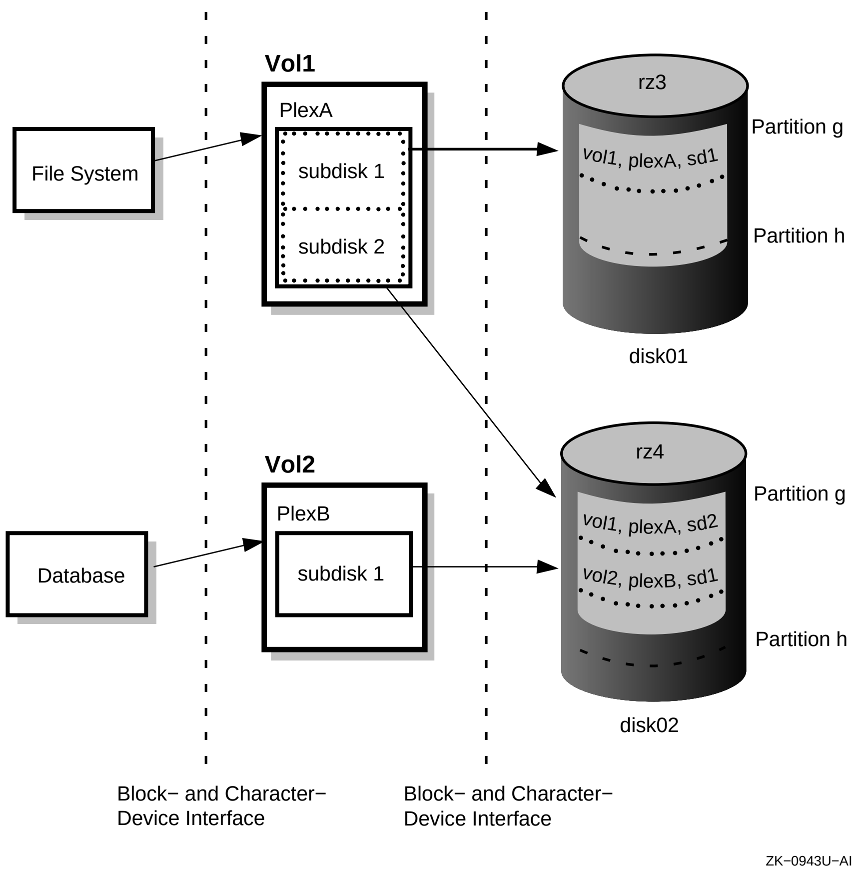
```
# newfs /dev/rvol/rootdg/vol01 rz29
# mount /dev/vol/rootdg/vol01 /mnt
```

On a system that does not have LSM installed, I/O activity from the UNIX system kernel is passed through disk device drivers that control the flow of data to and from disks.

The LSM software maps the logical configuration of the system to the physical disk configuration. This is done transparently to the file systems, databases, and applications above it because LSM supports the standard Digital UNIX block-device and character-device interfaces to store and retrieve data on LSM volumes. Thus, applications do not need to be changed to access data on LSM volumes.

Figure 1-7 shows how file systems, databases, and applications store and retrieve data on LSM volumes.

Figure 1-7: I/O Activity to LSM Volumes



Section 1.5 describes the LSM volume device driver that handles I/O to LSM volumes and other central software components.

1.10 LSM Encapsulation Tools

LSM provides a set of tools that you can use to reconfigure existing user data into LSM volumes, without physically moving the data. This process is referred to as *encapsulation*.

The LSM encapsulation process examines the UNIX device, LVM volume group, or AdvFS domain the user specifies as input, and generates files containing instructions that actually implement the encapsulation changes.

Refer to Chapter 3 for information about encapsulating user data on UNIX style partitions, LVM volume groups, or AdvFS storage domains.

1.11 Introduction to Root and Swap Mirroring

LSM allows you to mirror the root and swap partitions to help maximize system availability. Using LSM to mirror the root and swap volumes provides complete redundancy and recovery capability in the event of boot disk failure. By mirroring disk drives that are critical to booting, you ensure that no single disk failure will leave your system unusable.

To provide root and swap mirroring, you encapsulate the partitions used for the root file system and swap partition to LSM volumes. The encapsulated root and swap devices appear to applications as volumes and provide the same characteristics as other LSM volumes.

If you do not mirror your system’s root and swap devices, you may lose the ability to use or reboot the system in the event of the failure of the boot disk.

See Chapter 5 for complete information about root and swap mirroring; see Section 7.9 for information about setting up an LSM mirrored volume for secondary swap.

LSM Configurations 2

LSM supports a variety of LSM configurations including concatenated disks, mirroring, striping, and multiple disk configurations. This chapter describes these configurations, and presents some options you should consider when planning your LSM configuration.

2.1 Planning an LSM Configuration

Before setting up LSM volumes, plexes, and subdisks, you should consider the needs of your site, the hardware available to you, and the rationale for creating volumes and disk groups.

Table 2-1 presents some configuration options and describes the planning considerations that apply to LSM configurations.

Table 2-1: LSM Configuration Considerations

Configuration	Description
Concatenated Volumes	You concatenate multiple LSM disks to form a big volume. You can use a concatenated volume to store a large file or file systems that spans more than one disk. Disk concatenation frees you from being limited by the actual physical sizes of individual disks so that you can combine the storage potential of several devices. Use the default disk group, `rootdg`, to create a concatenated volume out of the public regions available. You can also add more LSM disks and create volumes out of the new disks you added.
Mirrored Volumes	You associate multiple plexes with the same volume to create a mirrored volume. If you are concerned about the availability of your data, then plan to mirror data on your system. You should map plexes that are associated with the same volume, to different physical disks. For systems with multiple disk controllers, it is best to map a volume's plexes to different controllers. The `volassist` command will fail if you specify a device that is already in the volume as the mirrored plex, whereas the bottom-up commands will not fail.

Table 2-1: (continued)

Configuration	Description
Striped Volumes	For faster read/write throughput, use a volume with a striped plex. On a physical disk drive, the drive performs only one I/O operation at a time. On an LSM volume with its data striped across multiple physical disks, multiple I/Os (one for each physical disk) can be performed simultaneously.
	The basic components of a striped plex are the stripe width, the number of stripes, and the size of the plex in multiples of the stripe width that was used. Stripe blocks of the stripe width size are interleaved among the subdisks resulting in an even distribution of accesses between the subdisks. The stripe width defaults to 128 sectors, but you can tune the size to specific application needs. The `volassist` command automatically rounds up the volume length to multiples of the stripe width.
Mirrored and Striped Volumes	Use mirrored and striped volumes when speed and availability are important. LSM supports mirroring of striped plexes. This configuration offers the improved I/O performance of striping while also providing data availability.
	Note that the different striped plexes in a mirrored volume do not have to be symmetrical. For instance, a three-way striped plex can be mirrored with a two-way striped plex, as long as the plex size is the same.

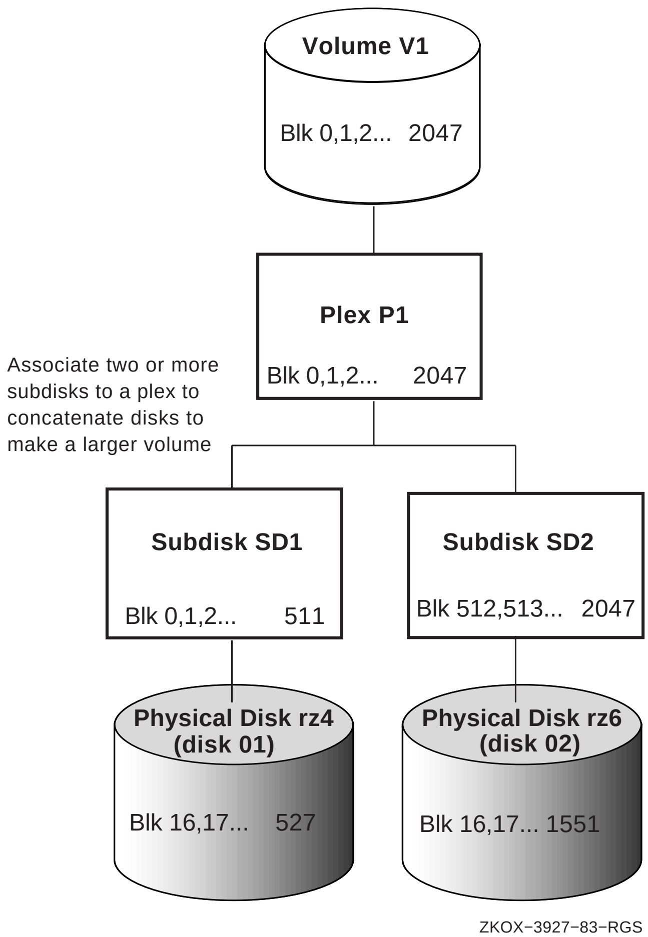
In addition, note that reads can be serviced by any plex in a mirrored volume. Thus, a mirrored volume provides increased read performance. However, LSM issues mirrored writes to all plexes in a mirrored volume. Because the writes are issued in parallel, there is a small amount of additional overhead as a result of write I/O to a mirrored volume.

2.2 Disk Concatenation

Disk concatenation involves arranging subdisks both sequentially and contiguously in the address space of a plex. With concatenation, subdisks are linked together into the logical address space. Data is then accessed from each of the subdisks in sequence.

Figure 2-1 gives an example of a concatenated disk.

Figure 2-1: Disk Concatenation Using LSM



Concatenated volumes consist of subdisks from one or more disks. Concatenated volumes with subdisks from more than one disk are also referred to as *spanned* volumes because the volume spans multiple physical disks. The advantages of using each type of volume are as follows:

- 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.
- Spanning is a special instance of concatenation that allows a particularly large file system or database to span multiple physical disks. Spanning takes advantage of the LSM virtual-disk environment to avoid the size limitations associated with physical disk size.

2.3 Disk Striping

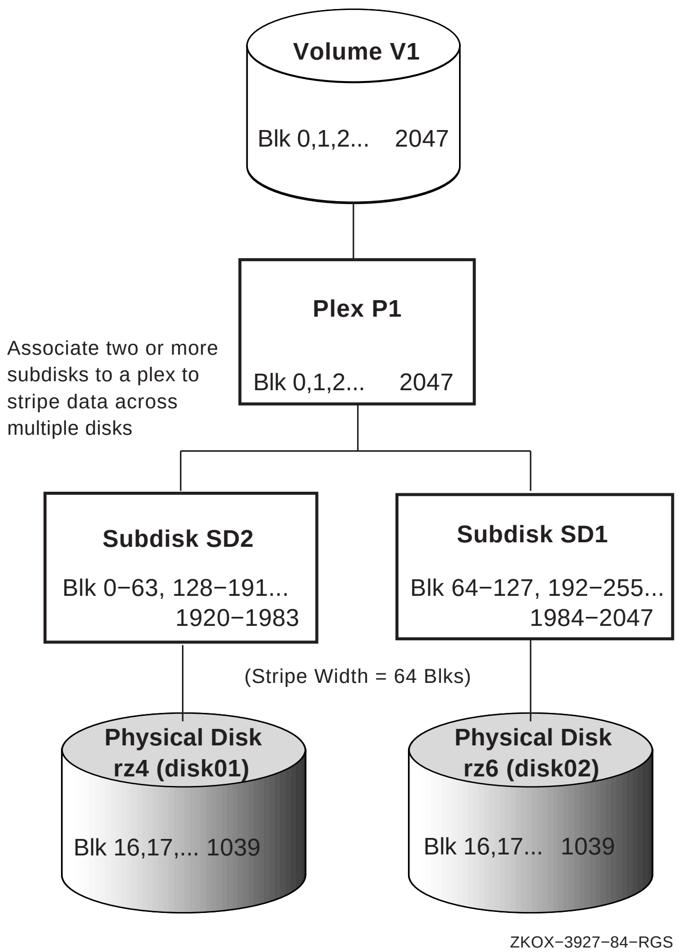
The manner in which storage space is allocated to file systems and databases has a direct impact on disk head movements and on the distribution of the I/O load between disk drives. An optimal allocation minimizes head movements and distributes the I/O load evenly between disk drives.

Striping involves spreading data across several physical disks. By supporting striping in addition to concatenation as a storage-allocation scheme for plexes, LSM makes it possible to evenly distribute the I/O load for a plex across a number of disk drives.

Stripes are relatively small, equally-sized fragments that are allocated alternately and evenly to the subdisks of a single plex. A striped plex consists of a number of equally-sized subdisks, each located on a separate disk drive. There should be at least two subdisks in a striped plex, each of which should exist on a different disk.

Data is stored on the subdisks in *stripe blocks* of a fixed size (referred to as the *stripe width*). Stripe blocks are interleaved between the subdisks as shown in Figure 2-2, resulting in an even distribution of accesses between the subdisks.

Figure 2-2: Striping Disks with LSM



By allocating storage evenly across multiple disks, striping helps to balance I/O load in cases where high traffic areas exist on certain subdisks. Throughput increases with the number of disks across which a plex is striped. The increase in throughput depends on the applications and file systems being used, and on the number of users using them at the same time.

The effect of striping on performance depends on the choice of the stripe width and on application characteristics. LSM uses a default stripe width of 128 sectors, which works well in most environments.

2.4 Mirroring Disks

In a system without LSM, failures of a physical disk result in the loss of the data on that disk. To recover from such an event, the data needs to be restored from a backup and all changes made to the data since that backup have to be reapplied. This is a time-consuming process, during which applications have no access to the data.

LSM makes it possible to protect critical data against disk failures by maintaining multiple copies (called mirrors) of the data in a volume. The LSM object that corresponds to a mirror is a *plex*. In the event of a physical disk failure, the plex on the failed disk becomes temporarily unavailable, but the system continues to operate using the unaffected plexes. Note the following rules when using LSM plexes to mirror disks:

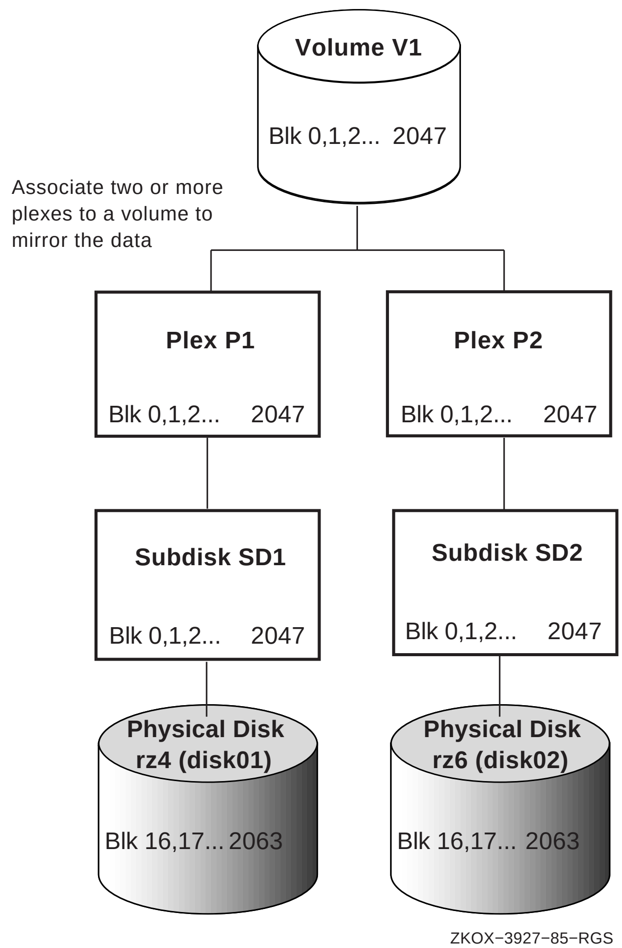
- Although a volume can have a single plex, at least two plexes are required to provide duplication of data. Each of these plexes should contain disk space from different disks for the redundancy to be effective. If possible, locate the plexes' disk space on different controllers for a higher level of availability and safety.
- When striping or spanning across a large number of disks, failure of any one of those disks will usually make the entire plex unusable. The chance of one out of several disks failing is enough to make it worthwhile to consider mirroring to improve the reliability of a striped, spanned, or concatenated volume.

All plexes are kept up to date as updates are made to the contents of the volume. If a read to a plex fails, other plexes are used to correct or mask the error. Users of a volume are shielded against any failures unless all plexes fail.

If your applications perform an equal proportion of read and write operations, or if your applications perform more writes, you are not likely to gain performance (in fact, you might lose performance) if you mirror the data. However, if your files or applications perform significantly more read operations, you can improve performance with mirroring.

Figure 2-3 shows a mirrored LSM configuration.

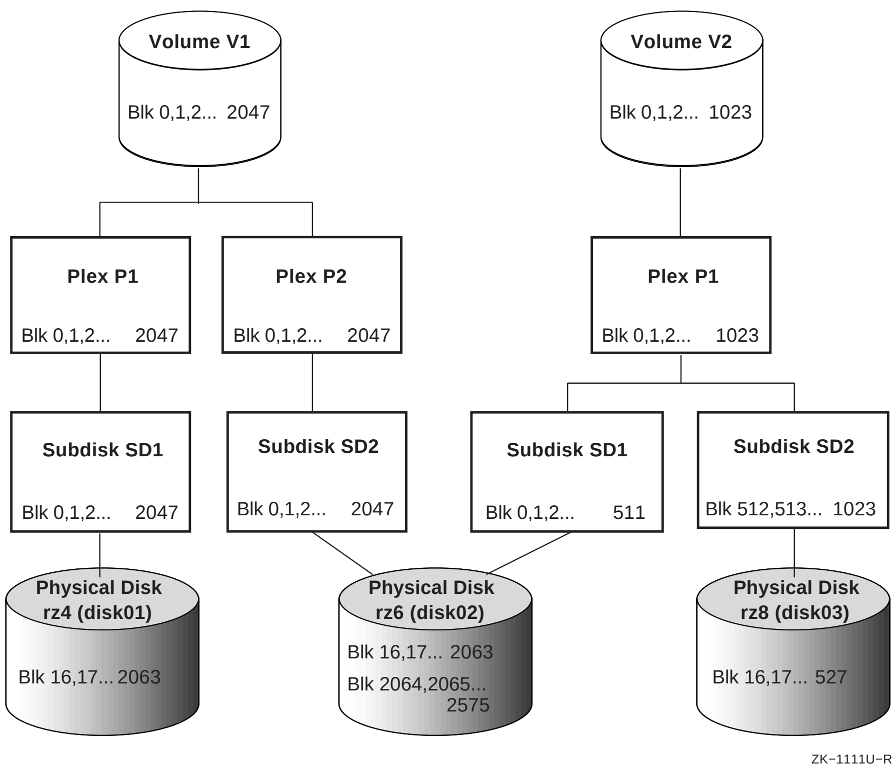
Figure 2-3: Mirroring Using LSM



2.5 Multiple Logical Volumes per Physical Disk

In LSM, each subdisk maps to a physical disk offset and length. This means that different LSM volumes can have subdisks that map to different areas of the same physical disk. For example, as shown in Figure 2-4, the mirrored volume `V1` can use disks `rz4` and `rz6`, and the concatenated volume `V2` can also use `rz6` plus `rz8`.

Figure 2-4: Same Physical Disk for Multiple Logical Volumes

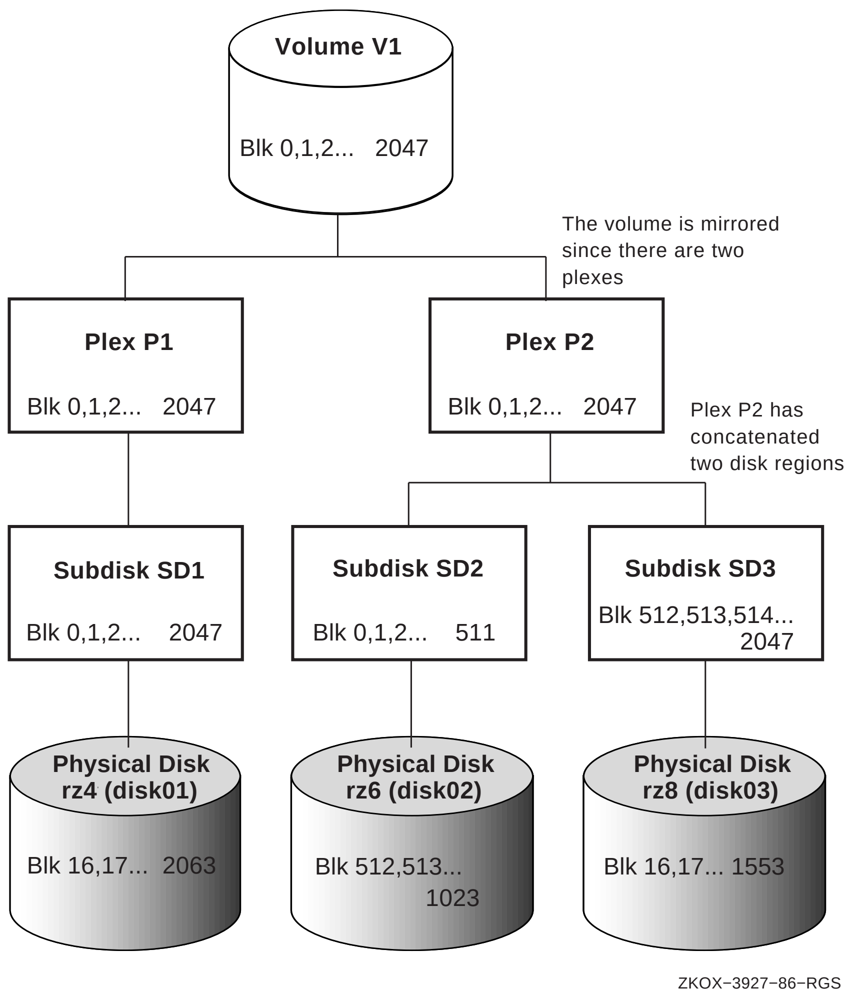


2.6 Making LSM Configuration Changes

LSM provides a high degree of flexibility in the way volumes can be mapped to disk and partition devices. For example, you can use LSM to build combinations of plexes with subdisks, as shown in Figure 2-5. This flexibility allows you to optimize performance, change volume size, add plexes, and perform backups or other administrative tasks without interrupting system applications and users.

LSM permits dynamic reconfiguration of the volumes, making it easy to adapt to changes in I/O load and application needs, and to maximize system availability. See Section 13.3 for more information about implementing configuration changes.

Figure 2-5: LSM Objects in Combination

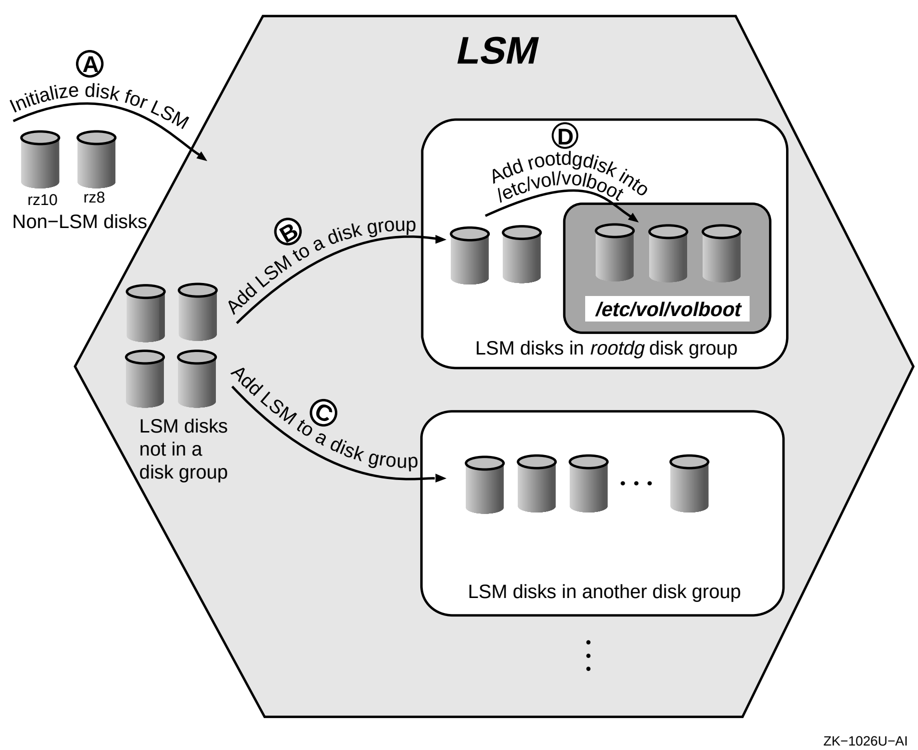


Setting Up LSM 3

This chapter describes how to set up the Logical Storage Manager (LSM). It describes how to reenable LSM after an installation, as well as how to set up LSM for the first time.

To begin using LSM, you must initialize disks or partitions for LSM use and configure the disks into an LSM disk group. Figure 3-1 is a conceptual representation showing how disks are placed under LSM control.

Figure 3-1: Configuring Disks into an LSM Disk Group



3.1 Preparing for Digital UNIX Installation

If you are already running LSM, perform the following steps before performing a full installation of Digital UNIX:

1. Make a backup copy of the `/etc/vol/volboot` file. You will need to restore this file after the full installation.
2. If LSM volumes are in use for the `root` (/), `swap`, `/usr`, or `/var` file systems, unencapsulate the LSM volumes. Refer to Section C.20 and Section C.19 for unencapsulation instructions and examples.

   You will need to reencapsulate the file systems and swap devices after the installation of Digital UNIX.
3. Use the `/usr/sbin/volsave` command to save the current copy of the LSM configuration. Copy the saved LSM configuration to tape. Refer to Section 7.4 and Section C.21 for details.

See the *Installation Guide* for complete information on preinstallation tasks for LSM.

3.2 Reenabling LSM after a Reinstallation

If you are already running LSM and the `rootdg` disk group is already initialized, you do not need to reenable LSM. For example, if you have already performed an upgrade installation, skip this section.

If you had LSM initialized on a system before doing a full installation, you can reenable the LSM configuration by performing the following steps:

1. Copy the `/etc/volboot` file from a backup:

   ```
   # cp /backup/volboot /etc/volboot
   ```

2. Create the LSM special device file:

   ```
   # /sbin/volinstall
   ```

3. Start the LSM daemons and volumes:

   ```
   # /sbin/vol-startup
   ```

3.3 Initializing LSM

If you are setting up LSM for the first time, you can use either of the following approaches to initialize and configure disks for use with LSM.

- Use the `volsetup` utility to create the LSM configuration database for the first time. Then, use the `voldiskadd` utility to add more disks into LSM. This is the simplest method to set up an LSM configuration.

- Use a series of LSM low-level commands (for example, `voldisksetup`, `voldg`, `voldctl`) to create the database for the first time and later add new disks into an existing LSM configuration. Although this method is more complicated, it allows more control over your LSM configuration and environment.

These two approaches are described in the following sections. See also Appendix C for detailed examples of setup procedures.

3.3.1 Using the volsetup Utility

The `volsetup` utility provides an easy way to initialize LSM. This utility automatically modifies disk labels, initializes disks for LSM, creates the default disk group, `rootdg`, and configures disks into the `rootdg` disk group. Note that you invoke the `volsetup` utility only once. To later add more disks, use the `voldiskadd` utility (as described in Section 6.2.1).

The `volsetup` utility prompts you to estimate how many disks will be managed by LSM. The utility uses the estimate to define optimal values for the private region size (in sectors), and the number of configuration and log copies per disk. The default values for LSM configurations are shown in Table 3-1.

Table 3-1: Default Values for LSM Configurations

Number of Disks	Private Region Size	nconfig and nlog
1 to 4	1024	2
5 to 8	1024	1
More than 8	1024	1 for first 8 disks, 0 for others
More than 128	1536	1 for first 8 disks, 0 for others
More than 256	2048	1 for first 8 disks, 0 for others

Follow these steps to use `volsetup`:

1. If you are in single-user mode, set the host name for your system before initializing LSM.
2. Execute the `/sbin/volsetup` interactive utility by entering the

following command:

```
# /sbin/volsetup rz1
```

In this example, the disk `rz1` is used to initialize the `rootdg` disk group. If you do not give the name of a disk, LSM prompts you. Refer to the `volsetup`(8) reference page for information on how to handle partition overlap error messages.

Note

> When you are first setting up LSM, do not include the boot disk in the disks you specify to `volsetup`. After you have initialized LSM, you can encapsulate the root and swap partitions and add them to the `rootdg` disk group. See Section 5.2 for details.

3. The `volsetup` utility modifies the `/etc/inittab` file. On a system reboot, LSM is started automatically by the LSM entries in the `inittab` file. (See `inittab`(4) for more information.)
4. The LSM script `/sbin/lsmbstartup` starts the LSM daemon `vold` and the error daemon `voliod`. After running the `volsetup` procedure, check that the `vold` daemon is running. See Section 14.4 for more information.

The `volsetup` utility creates the `/etc/vol/volboot` file. This file is used to locate copies of the `rootdg` disk group configuration when the system starts up. Do not delete the `/etc/vol/volboot` file; it is critical for starting LSM. To update the `volboot` file, use `voldctl`; do not manually edit `/etc/vol/volboot`.

3.3.2 Initializing LSM with Individual Commands

As an alternative to using the `volsetup` command (described in Section 3.3), you can use individual LSM commands to initialize and configure disks for use with LSM. Use individual LSM commands when you need additional flexibility and control to match the LSM configuration to a site's particular needs and to optimize performance. You use individual LSM commands to perform the following tasks:

- Modify the disk's label and set up a private region on the disk.
- Add the disk into an LSM disk group.
- Add the disk to the `/etc/vol/volboot` file.

The following sections describe how to use LSM commands to accomplish these tasks.

3.3.2.1 Initializing /etc/vol/volboot, Starting vold, and Initializing the rootdg Disk Group

Follow these steps to initialize `/etc/vol/volboot`, start the `vold` daemon and initialize the `rootdg` disk group:

1. If you are in single-user mode, set the host name for your system, as follows:

   ```
   # /sbin/hostname hostname
   ```

2. Create a special device file using the following command:

   ```
   # /sbin/volinstall
   ```

3. Start `vold` in the disabled mode using the following command:

   ```
   # /sbin/vold -m disable
   ```

4. Initialize `/etc/vol/volboot` using the following command:

   ```
   # voldctl init
   ```

5. Initialize the `rootdg` disk group using the following command:

   ```
   # voldg init rootdg
   ```

Note

You can use the `voldg init` command only once to create a disk group.

3.3.2.2 Selecting Private Region Parameters

Use the LSM individual commands described here to create and initialize the private region on the disk:

1. Set the number of configuration databases and the number of log regions with the `nconfig` and `nlog` options, respectively, using the guidelines described in Table 3-2. Starting with LSM Version 4.0, the default value for these options is 1.

   Whenever a disk group contains a large number of disks, distribute the disks that contain a configuration database and kernel change log across different controllers. By distributing the data across multiple controllers, you obtain higher availability.

Table 3-2: Settings for nconfig and nlog

If you expect to have...	Then...
Up to four disks in the disk group	Set the `nconfig` and `nlog` options to 2.
Up to eight disks in the disk group	Use the default value of 1 for the `nconfig` and `nlog` options.
More than eight disks in the disk group	Initialize the first eight disks by setting the `nconfig` and `nlog` options to 1, and initialize the remaining disks by setting the `nconfig` and `nlog` options to 0.

2. Set the size of the private region with the `privlen` option using the guidelines described in Table 3-3.

Table 3-3: Private Region Sizes

If you plan to add the disk to...	Then...
An existing disk group	Set the private region size for the disk large enough to accommodate the disk group's current database size.
A new disk group that does not currently exist (but you plan to create the new diskgroup)	Try to estimate the future growth of the disk group and include additional space as you determine the disk's private region size.

Note

Adding a disk with a smaller private region size than what is in use by other disks in a disk group can shrink the diskgroup's configuration database size.

Typically, the private region size should be a minimum of 512 sectors and a maximum of 2048 sectors. Starting with LSM Version 4.0, the default private region size is 1024 sectors. For systems configured with 128 or fewer physical disks, you can use the default private region size. With this value, the private region's configuration database can usually contain up to 1400 records, which is sufficient for systems configured with as many as 128 disks (assuming a typical configuration that has approximately 8 records per disk).

3. Determine the number of records needed in an LSM configuration database. To do this consider the following:
   - Each object (volume, plex, subdisk, and disk) created in the disk group requires one record in the configuration database.
   - The size of each LSM record varies. Approximately two records can fit in one sector (512 bytes).
   - Disks that will be added to the `rootdg` disk group need one record for each LSM disk group.
   - Disks that will be added to the `rootdg` disk group need one record for each disk in all other disk groups.
   - Certain LSM configuration changes (for example, moving a subdisk) involve creating and later deleting two to four temporary records to perform the operation.
   - Forecast expected disk growth and allow additional records to handle that growth.
   - Private regions that contain two configuration database copies will cut in half the number of records available.

The following sections show how to initialize and add disks to an LSM environment.

3.3.2.3 Initialize Disk Label and Add to LSM

Use the `voldisksetup` utility to initialize a disk for LSM use. The disk must already have a disk label.

The utility modifies the disk's label and initializes the private region. The private region contains:

- The disk header
- The table of contents
- Zero or more copies of the configuration database
- Zero or more copies of the kernel change log

The `voldisksetup` utility performs actions that are equivalent to using the `disklabel` and `voldisk` commands.

The following examples of the voldisksetup command demonstrate how to initialize a complete disk and how to initialize a disk partition (rz8 and rz10d) for use with LSM:

```
# voldisksetup -i rz8 nconfig=1 privlen=1024

# voldisksetup -i rz10d nconfig=1 privlen=1024
```

See the voldisksetup(8) reference page for more information about the voldisksetup command, its options, and partition overlap error messages.

Note

> To add a disk with no configuration database, set the nconfig attribute to 0 when initializing the disk with voldisksetup. Do not initialize a new disk as a nopriv disk – this disk type is appropriate only for encapsulation of existing data.

3.3.2.4 Adding a Disk to a Disk Group

To create a new disk group, use the voldg init command, as show in Section 3.3.2.1. After the disk group is created, you can add other disks into the disk group using the voldg adddisk command. For example, the following command line shows how to add another disk to the rootdg disk group:

```
# voldg -g rootdg adddisk disk02=rz10d
```

See the voldg(8) reference page for more information about this command and its options.

3.3.2.5 Disks Added to /etc/vol/volboot

When LSM is initialized, the disks in rootdg are added to the /etc/vol/volboot file.

When the system is booted, LSM uses information contained in the /etc/vol/volboot file to find the location of at least one disk that was added to the rootdg disk group. Using the configuration database on the disk listed in /etc/vol/volboot, LSM obtains information about the LSM disks, disk groups, and other configuration data necessary to start up LSM.

3.3.2.6 Starting LSM Manually

To manually start LSM, take the following actions:

1. Start two error daemons by entering the following:

```
# voliod set 2
```

2. Start up `vold` in enabled mode by entering the following:

```
# vold -k
```

   If `vold` is running in disabled mode, enter the following:

```
# voldctl enable
```

3. Enable the LSM volumes by entering the following:

```
# volrecover -sb
```

3.4 Increasing the Configuration Limits

When you configure LSM, the default limit set for the number of volumes and plexes allowed per system is smaller than the maximum number allowed. The default and maximum LSM limits are shown in Table 3-4.

Table 3-4: Configuration Limits

LSM Object	Default Limit per System	Maximum Limit per System
Volumes[a]	1021	4093
Plexes	1024	4096
Subdisks per plex	4096	4096

Table note:

a. These limits include the reserved volumes `rootvol` and `swapvol`. Three additional volume minor numbers are reserved for future use.

You can increase the default number of volumes allowed on a system by editing the `/etc/sysconfigtab` file. For example, to increase the maximum number of volumes allowed on the system from 1024 to 2048 volumes, edit the /etc/sysconfigtab file and add the following lines:

```
lsm:
       max-vol=2048
```

The change to /etc/sysconfigtab will take affect on the next system reboot. Note that in this example lsm is the name of the subsystem and max-vol is the attribute that is being changed. The maximum number of plexes allowed per system is the same as the maximum number of volumes configured on the system. If the attribute max-vol is not specified in /etc/sysconfigtab, LSM uses its default.

To boot the system with a different value for max-vol than either the default number of volumes or the value of max-vol specified in /etc/sysconfigtab, boot the system in interactive mode. For example, the value of max-vol can be set to 3072 by specifying maxvol=3072 when the system is booted interactively, as shown here:

```
>>> boot -fl i
...............
...............
...............
[Enter kernel name ... ] vmunix maxvol=3072
```

In this example, the value 3072 overrides the value of max-vol that is set in the /etc/sysconfigtab file and the default value of 1024.

See the sysconfigtab(4) reference page for information on the format used for the /etc/sysconfigtab file.

3.5 Post-Setup Tasks

After you have initialized LSM, there are several other steps you should complete to begin using LSM:

1. Encapsulate existing LVM volumes, UFS file systems, and AdvFS file systems that you want to put under LSM control. See Chapter 4.
2. Encapsulate and mirror the root and swap partitions. See Chapter 5.
3. Create new LSM volumes. See Chapter 7.
4. Use the volsave command to save copies of your configuration files. See Section 7.4 for information on using this command.

Encapsulating Existing User Data to LSM Volumes 4

This chapter describes how to place existing user data under LSM control by using a process called encapsulation.

4.1 Data Encapsulation

LSM supports data encapsulation from the following formats:

- Logical Volume Manager (LVM) volumes
- UNIX File System (UFS) data partitions
- Advanced File System (AdvFS) storage domains

Note

> LSM does not support encapsulation of user data on ULTRIX Disk Shadowing (UDS) volumes or ULTRIX Striping Driver stripe volumes.

During the encapsulation process, LSM transforms an LVM volume group, a UNIX style disk or disk partition, or an AdvFS storage domain into an LSM logical volume. Using a physical device name that you supply in an encapsulation command, LSM identifies how the device can be used for file systems and generates LSM volumes to cover those areas on the disk.

The following commands allow you to perform a one-time conversion of existing user data into LSM volumes:

- The `vollvmencap` command migrates LVM volume groups to LSM. This command is described in Section 4.3.
- The `volencap` command migrates UNIX physical devices or disk partitions (for example, a UFS file system residing on a partition or database data in a partition) and AdvFS domains to LSM. This command is described in Section 4.4 and Section 4.5.

Note

In previous releases of LSM, a separate command, `voladvdomencap`, was used for migrating AdvFS domains to LSM. Starting with Version 4.0, the functionality of this command is provided by the `volencap` command. The `voladvdomencap` command is supported only for backward compatibility and is scheduled to be withdrawn from a future release of LSM.

See Chapter 5 for information about encapsulating the partitions used for the root file system and swap partition to LSM volumes.

See Section 7.11, Section C.20, and Section C.19 for information on unencapsulation procedures.

4.2 Encapsulation Requirements

The following list describes requirements for performing encapsulation functions:

- The `rootdg` disk group must exist and be enabled, except during the first encapsulation of the system disk.

  Some configurations require that the encapsulation process create an LSM nopriv disk. This type of encapsulation uses `rootdg` to store the configuration data during encapsulation.

- Back up all user data before beginning the encapsulation process.

  In the event that a failure occurs during encapsulation, you can restore the saved data to return data to its original state.

- Perform encapsulation when data is off line.

  To minimize the risk of configuration changes during encapsulation, ensure the data remains off line during the encapsulation process.

4.3 LVM Volume Encapsulation

The LVM encapsulation process uses the name of a volume group that you specify with the `vollvmencap` command, and transforms the LVM volumes into LSM volumes.

Note

The Logical Volume Manager (LVM) is no longer supported on Digital UNIX systems. Support for the LVM encapsulation tools will also be retired in a future release of Digital UNIX. At that time, any data still under LVM control will be lost.

4.3.1 Overview of LVM Support in Digital UNIX

Encapsulation of LVM volumes is based on volume groups, which are collections of physical volumes, each of which contains the following:

- LVM metadata area
- LVM data area
- LVM bad block replacement directory

In addition, the LVM data area is divided into physical extents, which are the basic building blocks of LVM volumes. The physical extents for all physical volumes in a volume group are all the same size.

Finally, a volume consists of a series of logical extents, each of which maps to one or more physical extents. Because Digital UNIX does not support mirroring for LVM, each logical extent can map to only one physical extent, except in the occasional event when LVM requires temporary mirrors to be added to a volume for the duration of the command execution. For encapsulation purposes, these transient conditions are not considered.

Note

The physical extent bad block directory is not used.

The `/etc/lvmtab` file defines all the volume groups and their associated physical volumes on a system. When a system reboots, LVM restarts based on the information defined in this file.

There is an LVM record at the beginning of each physical volume in a volume group. The LVM record contains a number, and the location and length of the metadata region. In the metadata region, there are entries for each logical volume defined in the volume group, and mappings of each physical extent to logical extents of the physical volumes.

A typical LVM configuration has a few physical device partitions in a volume group. An arbitrary number of volumes is defined by mapping logical extents to physical extents in a volume group. These volumes are used for UFS file systems. User data can also be accessed directly through the device interface.

4.3.2 Encapsulating LVM Volumes

To begin the encapsulation process, you supply the name of an LVM volume group as input to the `vollvmencap` command. For example:

```
# /usr/sbin/vollvmencap /dev/vg1
```

The `vollvmencap` command generates scripts containing the LSM commands needed to create LSM volumes. You execute the command scripts created by `vollvmencap` by running the `/sbin/vol-lvm-reconfig` command, as shown here:

```
# /sbin/vol-lvm-reconfig
```

Note that the LVM volumes in the volume group that was encapsulated must not be in use when `/sbin/vol-lvm-reconfig` is executed. For example, all file systems using LVM volumes must be unmounted.

When the encapsulation is successful, a message is printed that indicates a name of a script that you must execute to remove LVM volumes. Run this script only after ensuring that the encapsulation was successful.

Note the following requirements for the LVM encapsulation process:

- The entire volume group must be encapsulated at the same time.
- The volume group configuration must be static for the duration of the encapsulation process.
- The LVM volume group can be accessed using the `vgchange command`.
- A volume group can be encapsulated into the default LSM disk group `rootdg` or another disk group. In either case, the disk group must exist before the encapsulation process begins.

The encapsulation process creates an LSM subdisk for contiguous physical extents in physical volume mappings to a logical volume. Because LVM volumes in Digital UNIX are not mirrored, the LSM volume has only one plex. The plex consists of a set of subdisks obtained by mapping physical extents associated with each logical extent. The plex is used to create an LSM volume. The LSM volume name replaces the LVM volume name in the `/etc/fstab` file.

4.3.3 Preserving Block 0

Block 0 on a Digital UNIX disk device is read-only by default. UFS does not use block 0 when putting data on device partitions. To preserve the LBN mapping, the LSM nopriv disk must start at LBN block 0. As long as this disk is used for UFS volumes, this does not present a problem. However, if the disk is reused for other applications which write to block 0, then a write failure will occur. To help avoid such failures, earlier releases of LSM labeled the LSM nopriv disk with the unique administration name *device-name* _blk0 . If the volume is no longer needed, remove this nopriv disk from the LSM disk group and redefine the disk without block 0

Starting with Version 4.0, the voldiskadd and voldisksetup utilities automatically map out block 0. Digital recommends that you use these utilities to add disks to LSM. Note that if volencap is used to add a disk to LSM, it will not preserve block 0. This can cause problems if an application writes to that part of the disk.

4.4 UNIX Style Partition Encapsulation

The encapsulation process for UNIX style disks and disk partitions uses the volencap command to change a disk or partition into an LSM disk.

The volencap command automatically encapsulates user data for common configuration layouts such as the following:

- An empty physical device.
- An empty partition.
- An entire physical device where one or more partitions are in use. Each partition that is in use will be encapsulated into a volume.
- A partition that is in use on a physical device.

4.4.1 Overview of Digital UNIX Partitions

The partitions on a physical device are mapped by a partition table called the *disklabel*. The disk's partitions and disk label have the following characteristics:

- The disk label maps a physical disk to a maximum of eight partitions that are labeled from a to h.
- Partition c covers the whole disk.
- The partitions can overlap each other.
- Each partition has an fstype attribute. If a partition is not in use, the fstype is unused. If a partition is in use, the fstype can have values such as 4.2BSD, AdvFS, swap, LSMsimp, LSMpubl, LSMpriv, or LSMnopriv.
- The disk label resides at block 0 of a physical device. For most devices, the default disk label is defined in /etc/disktab.
- You can modify an online disk label using the utility program /usr/sbin/disklabel.

Each available partition has a special device file in the /dev directory. Users and applications access storage through these special device files. The voldisk, voldisksetup, and voldiskadd utilities perform partition overlap checks to ensure that partitions being initialized to LSM do not have valid UFS, AdvFS, swap, or LSM data. If the fstype field of a partition indicates that there is valid data, the utilities issue a warning.

4.4.2 Encapsulating UNIX Partitions

The volencap and vol-reconfig commands provide an easy way to encapsulate disks and partitions. However, if you need a finer degree of control, use the manual encapsulation procedure to tailor the encapsulation to the specific needs of your configuration. See Section 4.4.4 for information to help you encapsulate UNIX style partitions manually.

To begin the encapsulation process, you supply the name of a physical device (for example, rz3) or a partition name (for example, rz3g) as input to the volencap command. For example:

```
# /usr/sbin/volencap rz3
```

The LSM encapsulation process uses information in the disk label and the /etc/fstab file to find out if a partition is in use, for example if it contains a UFS file system or a database. If the partition does not have information in a disk label or /etc/fstab file to indicate that it is being used by an application, the partition must be encapsulated using the partition name.

The /usr/sbin/volencap command generates scripts containing the necessary LSM commands and files to create LSM volumes. You run these scripts by executing the /sbin/vol-reconfig command, as shown here:

```
# /sbin/vol-reconfig
```

If any partition or disk that has been encapsulated is still in use, reboot the system.

Instead of executing /sbin/vol-reconfig manually, you can add /sbin/vol-reconfig to the the /etc/inittab file by running the volinstall command. Then, when the system is rebooted, the encapsulation commands generated by /usr/sbin/volencap take effect.

Use the above method if any disk or partition that was encapsulated was in use.

The results of the encapsulation process are as follows:

- If the encapsulation process is performed on a disk partition, LSM encapsulates the partition as a nopriv disk because there is no free space available to store the configuration database. A nopriv disk can be added to a disk group only if other disks in the disk group have a defined metadata region.
- If the encapsulation process is performed on a physical disk, LSM adds the physical disk to the LSM environment as a simple disk when the following conditions are true:
  - All user data and, hence, the public region of the LSM simple disk, can be defined as a contiguous region.
  - There is room for a small private region at the beginning or end of the disk.
- If the encapsulation process is performed on a physical disk, and the disk cannot be encapsulated as a simple disk, LSM adds each active partition on the physical disk to the LSM environment as an individual nopriv disk.

After the encapsulation, LSM converts each partition that is in use (for example, as a UFS file system) to a subdisk. LSM then uses the subdisk to create a plex and, in turn, uses the plex to generate an LSM volume. Entries in the `/etc/fstab` or `/sbin/swapdefault` file are changed to use the LSM volume name instead of the block device name of the physical disk partition.

4.4.3 Preserving Block 0

Block 0 on a Digital UNIX disk device is read-only by default. UFS does not use block 0 when putting data on device partitions. To preserve the LBN mapping, the LSM nopriv disk must start at LBN block 0. As long as this disk is used for UFS volumes, this does not present a problem. However, if the disk is reused for other applications which write to block 0, then a write failure will occur. To help avoid such failures, earlier releases of LSM labeled the LSM nopriv disk with the unique administration name *device-name* `_blk0`. You should remove this nopriv disk from the LSM disk group and redefine it without block 0 if the volume is no longer needed.

Starting with Version 4.0, the `voldiskadd` and `voldisksetup` utilities automatically map out block 0. Digital recommends that you use these utilities to add disks to LSM. Note that if `volencap` is used to add a disk to LSM, it will not preserve block 0. This can cause problems if an application writes to that part of the disk.

4.4.4 Encapsulating UNIX Partitions Using Individual Commands

You need to perform a manual encapsulation only when the automatic `volencap` encapsulation process does not apply to your configuration. Before beginning the encapsulation process, do the following:

1. Ensure that all the disks or partitions that you intend to encapsulate are not in use. If a partition is currently mounted, it should be unmounted.
2. Save the original `/etc/fstab` file. Change the file to use the LSM volume names instead of the partition names.
3. Make sure the `rootdg` disk group exists and is active.

To encapsulate a partition follow these steps. This example encapsulates the `/dev/rz3h` partition that is being used as `/usr/staff`.

1. The following example shows how `rz3h` appears in the `/etc/fstab` file:

   ```
   /dev/rz3h       /usr/staff    ufs rw 1 2
   ```

2. Add the `rz3h` partition as a nopriv LSM disk. For example:

   ```
   # voldisk -f init rz3h type=nopriv
   ```

3. Add the `rz3h` partition to a disk group using the following instructions:

   – To add `rz3h` to `rootdg`, enter the following command:

   ```
   # voldg -g rootdg adddisk rz3h
   ```

   – To add the `rz3h` partition to the existing disk group `dg1`, enter the following command:

   ```
   # voldg -g dg1 adddisk rz3h
   ```

4. Using the information in the disk label, find the size of partition `h`. Use the following command to display the disk label information for `rz3`:

   ```
   # disklabel -r rz3:
   #   size    offset fstype [fsize bsize cpg]
   a:  131072       0 4.2BSD 1024 8192 16 # (Cyl.    0 - 164*
   b:  262144  131072 unused 1024 8192    # (Cyl.  164*- 492*)
   c: 2050860       0 unused 1024 8192    # (Cyl.    0 - 2569)
   d:  552548  393216 unused 1024 8192    # (Cyl.  492*- 1185*)
   e:  552548  945764 unused 1024 8192    # (Cyl. 1185*- 1877*)
   f:  552548 1498312 unused 1024 8192    # (Cyl. 1877*- 2569*)
   g:  819200  393216 4.2BSD 1024 8192 16 # (Cyl.  492*- 1519*)
   h:  838444 1212416 4.2BSD 1024 8192 16 # (Cyl. 1519*- 2569*)
   ```

   The example output shows that the size of partition `h` is 838444 sectors.

5. Find a unique name for the LSM volume corresponding to partition `rz3h`. Use a volume name that uses the partition name (for example, `vol-rz3h`).

6. Create an LSM volume. Provide the correct disk group name. The following example uses `rootdg` as the disk group name:

   ```
   # /sbin/volassist -g rootdg  make vol-rz3h 838444s rz3h
   ```

7. Change the `/etc/fstab` file to look as follows:

   ```
   /dev/vol/rootdg/vol-rz3h       /usr/staff    ufs rw 1 2
   ```

To encapsulate a complete disk to LSM and convert partitions that are in use into LSM volumes follow these steps.

In the following example for the disk `rz4`, two partitions store user data by means of a UFS file system, and one partition allows applications to directly access and store user data using the device interface.

1. The `/etc/fstab` file includes the following lines that correspond to the `rz4` disk:

   ```
   /dev/rz4d       /data1    ufs rw 1 2
   /dev/rz4e       /data2    ufs rw 1 2
   ```

   Partition `rz4f` is used by applications to store user data directly using the device interface. The following example shows the command and display for the disk label information for the `rz4` disk:

   ```
   # disklabel -r rz4
   #     size  offset  fstype [fsize bsize cpg]
   a:  131072       0  unused  1024 8192 16 # (Cyl.    0 - 164*
   b:  262144  131072  unused  1024 8192    # (Cyl.  164*- 492*)
   c: 2050860       0  unused  1024 8192    # (Cyl.    0 - 2569)
   d:  552548  393216  4.2BSD  1024 8192    # (Cyl.  492*- 1185*)
   e:  552548  945764  4.2BSD  1024 8192    # (Cyl. 1185*- 1877*)
   f:  552548 1498312  unused  1024 8192    # (Cyl. 1877*- 2569*)
   g:  819200  393216  unused  1024 8192 16 # (Cyl.  492*- 1519*)
   h:  838444 1212416  unused  1024 8192 16 # (Cyl. 1519*- 2569*)
   #
   ```

2. Edit the disk label before beginning the encapsulation process.

   In the example for the `rz4` disk, the output shows that partitions `d` and `e` are in use for UFS file system. The `f` partition is marked as unused even though it is in use. Applications that access data directly on partitions using the device interface must edit the disk label using the command `disklabel -e` to change the disk label before encapsulation. Because no appropriate disk label tags are provided, use of disk label tag 4.1BSD (shown in the following example) is suggested:

   ```
   #     size  offset fstype [fsize bsize cpg]
   a:  131072       0 unused 1024 8192 16 # (Cyl.    0 - 164*
   b:  262144  131072 unused 1024 8192    # (Cyl.  164*- 492*)
   c: 2050860       0 unused 1024 8192    # (Cyl.    0 - 2569)
   d:  552548  393216 4.2BSD 1024 8192    # (Cyl.  492*- 1185*)
   e:  552548  945764 4.2BSD 1024 8192    # (Cyl. 1185*- 1877*)
   ```

```
f:  552548 1498312 4.1BSD 1024 8192    # (Cyl. 1877*- 2569*)
g:  819200  393216 unused 1024 8192 16 # (Cyl.  492*- 1519*)
h:  838444 1212416 unused 1024 8192 16 # (Cyl. 1519*- 2569*)
```

3. An LSM disk can be created only when at least 512 sectors (or the length of the private region) are free either at the beginning or at the end of the disk.

   Use partition `c` to store the offset and size for the public and private regions of the LSM disk.

   If there is not enough space at either the beginning or end of the disk for the private region you must encapsulate `rz4c` as a nopriv disk.

   Note

   Block 0 on a disk is write-locked. Therefore, do not use block 0 either for the public or private region of the disk.

4. Make sure that partition `c` covers the entire disk.
5. Initialize `rz4c` and add it to the `rootdg` disk group as shown here. Note that in this example the private region is at the beginning of the disk, because there is no space for it at the end of the disk.
   - For an LSM simple disk, enter the following:

     ```
     # voldisk -f init rz4c type=simple privoffset=1 \
     privlen=512 puboffset=513 publen=2050347
     ```

   - For an LSM nopriv disk, enter the following:

     ```
     # voldisk -f init rz4c type=nopriv
     ```

6. Add the LSM disk to `rootdg`, as follows:
   - For an LSM simple disk, enter the following:

     ```
     # voldg -g adddisk rz4c
     ```

   - For an LSM nopriv disk, enter the following:

     ```
     # voldg -g adddisk rz4c
     ```

7. Create LSM volumes for all partitions that are in use.

   For an LSM simple disk, perform the following:

   a. To convert partition `d`, specify the starting offset of partition `d` in the public region. Because the public region starts at block 513, subtract 513 from the offset. (Note that the calculation differs when the private region starts at the beginning of the disk.)

The following example shows the conversion calculation for partition `d`:

```
[ partition d offset - 513 ] or [ 393216 - 513 ]
```

Enter the following command for partition `d`:

```
# volassist make vol-rz4c-01 552548 rz4c,392703
```

b. To convert partition `e`, specify the starting offset [945764 - 513] in the public region. For example:

```
# volassist make vol-rz4c-02 552548 rz4c,945251
```

c. To convert partition `f`, specify the starting offset [1498312 - 513] in the public region. For example:

```
# volassist make vol-rz4c-03 552548 rz4c,1497799
```

For an LSM nopriv disk, the calculation for the partition offset does not need to be changed because nopriv disks do not contain any metadata. The LSM disk partitions are as follows:

– For partition `d`, enter the following:

```
# volassist make vol-rz4c-01 552548 rz4c,393216
```

– For partition `e`, enter the following:

```
# volassist make vol-rz4c-02 552548 rz4c,945764
```

– For partition `g`, enter the following:

```
# volassist make vol-rz4c-03 552548 rz4c,1498312
```

8. Change the `/etc/fstab` file as follows:

– For an LSM simple disk, enter the following:

```
/dev/vol/rootdg/vol-rz4c-01       /data1    ufs rw 1 2
/dev/vol/rootdg/vol-rz4c-02       /data2    ufs rw 1 2
```

– For an LSM nopriv disk, enter the following:

```
/dev/vol/rootdg/vol-rz4c-01       /data1    ufs rw 1 2
/dev/vol/rootdg/vol-rz4c-02       /data2    ufs rw 1 2
```

Applications that were using `/dev/rz4f` should now use `/dev/rvol/rootdg/vol-rz4c-03`.

4.5 AdvFS Domain Storage Encapsulation

Encapsulation of AdvFS user data is at the storage domain level. Each physical device in the domain is encapsulated into an LSM volume by changing the links in the domain tree to point to the LSM volumes.

4.5.1 Overview of AdvFS Support on Digital UNIX

An AdvFS domain is a single-storage container consisting of one or more physical disk partitions. File systems, called filesets, are created and defined in the domain and can expand and contract within the domain if space is available. Storage devices can be added or removed from a domain even when filesets are mounted. Active filesets and domains are determined as follows:

- A fileset is considered active when it is mounted at a file system mount point.
- A domain is active if one or more of its filesets are active.

Storage devices can be physical devices or logical volumes. Each domain has a directory tree in the `/etc/fdmns` file that describes the physical disk partitions that constitute the storage container. The root of the tree is the domain and each leaf node in the domain directory tree is a physical disk partition name. This is a soft link of the full-access path of the physical disk partition.

A typical system has a number of domains using a few physical devices as storage in each domain, and with many filesets created on these domains.

4.5.2 Encapsulating AdvFS Domains

For most configurations, you can encapsulate user data automatically using the `volencap` command. However, if a finer degree of control is desired, use the manual encapsulation procedure to tailor the encapsulation to the specific needs of your configuration. See Section 4.5.4 for information.

Note

> In previous releases of LSM, a separate command, `voladvdomencap`, was used for migrating AdvFS domains to LSM. Starting with Version 4.0, the functionality of this command is provided by the `volencap` command. The `voladvdomencap` command is supported only for backward compatibility and is scheduled to be withdrawn from a future release of LSM.

The goal of encapsulating AdvFS domains is to capture the data in the physical disk partitions of a domain into LSM volumes, and present the same data access to AdvFS by changing the soft links in the domain directory tree to point to the LSM volumes.

LSM volumes encapsulated from domain physical devices must reflect the exact data at the exact logical block number (LBN) location as the physical device. The entire LBN range of the LSM nopriv disk is defined as one LSM subdisk. A plex is created with this subdisk and an LSM volume is created with the plex.

No mount point changes are necessary during encapsulation, because the filesets that are mounted are abstractions to the domain. The domain can be activated normally after the encapsulation process completes. Once the domain is activated, the filesets remain unchanged and the encapsulation is transparent to users of the AdvFS domain.

To begin the encapsulation process, you supply the name of a domain as input to the `volencap` command. For example:

```
# /usr/sbin/volencap dom1
```

The `/usr/sbin/volencap` command generates scripts containing the necessary LSM commands and files to create LSM volumes. You run these scripts by executing the `/sbin/vol-reconfig` command, as shown here:

```
# /sbin/vol-reconfig
```

The domain should not be in use when you execute the `/sbin/vol-reconfig` command. All filesets in the domain should be unmounted.

Instead of executing the `/sbin/vol-reconfig` command manually, you can add `/sbin/vol-reconfig` to the `/etc/inittab` file by running the `volinstall` command. Then, when the system is rebooted, the encapsulation commands generated by `/usr/sbin/volencap` take effect.

The `/etc/fdmns` file is updated on successful creation of LSM volumes.

4.5.3 Preserving Block 0

Block 0 on a Digital UNIX disk device is read-only by default. AdvFS does not use block 0 when putting data on device partitions. To preserve the LBN mapping, the LSM nopriv disk must start at LBN block 0. As long as this disk is used for AdvFS volumes, this does not present a problem. However, if the disk is reused for other applications which write to block 0, then a write failure will occur. To help avoid such failures, earlier releases of LSM labeled the LSM nopriv disk with the unique administration name `advfs_`*`device-name`*. You should remove this nopriv disk from the LSM disk group and redefine it without block 0 if the volume is no longer needed.

Starting with Version 4.0, the `voldiskadd` and `voldisksetup` utilities automatically map out block 0. Digital recommends that you use these utilities to add disks to LSM. Note that if `volencap` is used to add a disk to LSM, it will not preserve block 0. This can cause problems if an application writes to that part of the disk.

4.5.4 Encapsulating AdvFS Domains using Individual Commands

This section describes how to manually encapsulate AdvFS storage domains to generate LSM volumes. You need to perform a manual encapsulation only when the automatic `volencap` encapsulation process does not work for your configuration. The following instructions describe the manual process:

1. Check the AdvFS domain by entering the AdvFS domain inquiry command `showfdmn` on the domain that is to be encapsulated. You should encapsulate the domain only if the AdvFS indicates the domain is inactive. The possible outcomes of the `showfdmn` command are as follows:

   – If the domain does not exist, AdvFS returns the following error message:

   ```
   # showfdmn dom2
   showfdmn: unable to get info for domain 'dom2'
   showfdmn: error = No such file or directory
   ```

   – If the domain is active, AdvFS returns the following information:

   ```
   # showfdmn dom2
   Id                 Date Created              LogPgs  Domain Name
   2d2b5782.0009cca0  Wed Jan  5 19:12:50 1994   512   dom2
    Vol   512-Blks        Free  % Used  Cmode  Rblks  Wblks  Vol Name
    1L    1024000     1015408      1%     on    256    256  /dev/rz3c
   ```

   – If the domain is inactive, AdvFS returns the following information:

   ```
   # showfdmn dom2
   Id                 Date Created               LogPgs  Domain Name
   2d2b5782.0009cca0  Wed Jan  5 19:12:50 1994     512  dom2
   showfdmn: unable to display volume info; domain not active
   ```

2. Check the LSM disk group by entering the LSM disk group inquiry command, `voldg list`, on the target disk group. For example, to encapsulate the AdvFS storage domain into the disk group `dg1`, check that `rootdg` is enabled as shown in the following example:

   ```
   # voldg list
   NAME         STATE    ID
   rootdg       enabled  761416202.1025.chowfun.zk3.dec.com
   dg1          enabled  761416202.1034.chowfun.zk3.dec.com
   ```

3. Save the following information in the event that a recovery is needed:

   – Save the LSM disk group name in the directory `/etc/vol/reconfig.d/domain.d/domain_name.d`.

For example, before encapsulating the AdvFS domain dom2, enter the following commands:

```
# mkdir -p /etc/vol/reconfig.d/domain.d/dom2.d
# echo "dg1" > /etc/vol/reconfig.d/domain.d/dom2.d/dg
# cat /etc/vol/reconfig.d/domain.d/dom2.d/dg
dg1
```

– Save the domain directory tree to the directory `/etc/vol/reconfig.d/domain.d/domain_name.d`. For example:

```
# cp -R /etc/fdmns/dom2 \
 /etc/vol/reconfig.d/domain.d/dom2.d
```

4. Encapsulate the physical devices of the AdvFS domain into the target disk group. For example:

```
# ls -R /etc/fdmns/dom2
rz3c    rz16g
# voldisk -f init rz3c type=nopriv
# voldisk -f init rz16g type=nopriv
# voldg -g dg1 adddisk advfs_rz3c=rz3c advfs_rz16g=rz16c
```

5. Define volumes to represent the user data. For example:

```
# volprint -g dg1 -F "%len" -d advfs_rz3c
4109967
# volprint -g dg1 -F "%len" -d advfs_rz16g
301986
# volassist -g dg1 make vol_rz3c 4109967 advfs_rz3c
# volassist -g dg1 make vol_rz16g 301986 advfs_rz16g
```

6. Change the AdvFS soft links. For example:

```
# rm /etc/fdmns/dom2/rz3c
# rm /etc/fdmns/dom2/rz16g
# ln -sf /dev/vol/dg1/vol-rz3c /etc/fdmns/dom2/vol-rz3c
# ln -sf /dev/vol/dg1/vol-rz16g /etc/fdmns/dom2/vol-rz16g
```

7. Once the encapsulation is complete, mount filesets using their regular names, as shown:

```
# mount -t advfs dom2#fset2 /mnt
```

8. If the encapsulation fails, try to recover the domain by restoring the soft links. For example:

```
# rm -rf /ec/fdmns/dom2/vol*
# cp -R /etc/vol/reconfig.d/domain.d/dom2.d/dom2 /etc/fdmns
```

4.6 Using voldisk for Manual Encapsulations

In some cases, you may want to encapsulate a disk that does not have any space that can be used for an LSM private region partition. The `voldisk` utility can be used to encapsulate disks that do not have available space. This is done using special types of disk devices, called nopriv devices, that do not have private regions.

To perform this type of encapsulation, create a partition on the disk device that maps all parts of the disk that you want to be able to access. See `disklabel`(8).

Then, add the partition device for that partition using the following command syntax:

voldisk define *partition-device* **type=nopriv**

Here, *partition-device* is the basename of the device in the `/dev` directory. For example, to use partition `h` of disk device `rz3`, use the command:

```
# voldisk define rz3h type=nopriv
```

To create volumes for other partitions on the disk drive, add the device to a disk group, figure out where those partitions reside within the encapsulation partition, then use `volassist` to create a volume with that offset and length.

A major drawback with using these special encapsulation partition devices is that LSM cannot track changes in the address or controller of the disk. Normally, LSM uses identifying information stored on the physical disk to track changes in the location of a physical disk. Because `nopriv` devices do not have identifying information stored on the physical disk, this cannot occur.

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

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

LSM Root and Swap Disk Mirroring 5

This chapter describes the Logical Storage Manager support for mirroring the root and swap partitions.

You can find additional information in the `volencap`(8), `volintro`(8), `volrootmir`(8), and `volunroot`(8) reference pages.

Refer to Chapter 14 for information about recovering after a failure of the boot disk and on issues with system reinstallation.

5.1 Introduction to Root and Swap Mirroring

LSM provides the ability to encapsulate the partitions used for the root file system and swap partition to LSM volumes. The root and swap devices then appear to applications as volumes and provide the same mirroring characteristics as other LSM volumes.

Using LSM to mirror the root and swap volumes provides complete redundancy and recovery capability in the event of boot disk failure. By mirroring disk drives that are critical to booting, you ensure that no single disk failure will leave your system unusable.

Digital suggests you mirror the original boot disk using the `volrootmir` command, which causes LSM to mirror volumes created on the original boot disk onto another available disk. The volume that contains the root file system is referred to as the root volume (`rootvol`). The volume that contains the primary swap area is referred to as the swap volume (`swapvol`).

Other volumes that you use for swap (secondary swap volumes) are treated as ordinary LSM volumes.

For information about creating secondary swap volumes, refer to Section 7.9. For more information on mirroring the boot disk, see Section 5.3.

5.1.1 Root and Swap Usage Types

The following special usage types exist for root and swap volumes:

- `root` usage type

  This type is intended for volumes used as root devices. Because the root volume contains a file system, the `root` usage type resembles the `fsgen` usage type. The `root` usage type restricts the configuration of the volume such that all plexes of the volume are accessible as a root device through normal disk drivers.

- `swap` usage type

  This type is intended for volumes used as the primary swap device. Unlike the `root` device, a swap device does not contain a file system; therefore, it generally resembles the `gen` usage type.

5.1.2 Boot Disk Availability

LSM root and swap disk mirroring help you provide system availability by ensuring that you can boot the system despite certain errors. For example, booting can proceed even if errors occur when you start the swap volume. To do this, configure swap volumes so that all plexes of the volume are accessible as a swap device through normal disk drivers.

To boot the system despite the failure of the primary boot disk, you can boot the system using a disk containing the root and swap volume mirrors. See Section 5.3 for more information.

5.2 Encapsulating the Root and Swap Partitions

You set up to mirror root and swap by first encapsulating the existing boot disk. Digital recommends that the root and swap partitions be encapsulated together. The steps for encapsulating the root file system are the same whether you are using UFS or AdvFS.

Refer to Section 14.6 if you have problems with this procedure. Refer to Section C.14 and Section C.15 for detailed examples.

Follow these steps to encapsulate the root and primary swap partitions:

1. Verify that the following conditions are met before beginning the encapsulation process:

   – The root and primary swap partitions are both located on the root disk.

   – The `rootdg` disk group has been initialized. (Note that if you are encapsulating the entire boot disk, `rootdg` does not have to be initialized. In this case, the encapsulation procedure will initialize it.)

   – There are at least two free partitions on the root disk.

     The LSM encapsulation process requires two free partition-table entries to store LSM disk label tags. Note that the encapsulation requires only the partition-table entry; it does not need the disk space associated with the partition.

   LSM uses space from the swap area to create an LSM private region for the root disk. After the encapsulation process has finished, the swap area is smaller by the size of the private region (by default 1024 sectors).

2. Execute the `volencap` utility on the root disk. Do not specify particular partitions; supply the name of the physical boot device, as follows:

```
# /usr/sbin/volencap rz0
```

   This command creates the necessary LSM command scripts to convert all the partitions contained on `rz0` to LSM volumes. Note that you use the `volencap` command whether the root file system is a UFS or an AdvFS file system. This converts all in-use partitions on `rz0` to LSM volumes.

3. Enter the `shutdown` command with the reboot switch to execute the command scripts created by the `volencap` utility:

```
# shutdown -r +2
```

4. As part of the root encapsulation, LSM automatically reboots the system twice to set up the LSM configuration. During the system reboot, watch the system console for error messages that indicate problems with the encapsulation process.

   During the first reboot, the following messages are displayed on the system console.

```
vm_swap_init: warning /sbin/swapdefault swap
device not found

vm_swap_init: in swap over commitment node
```

   You can safely ignore these messages.

The names `rootvol` and `swapvol` are automatically assigned to the root volume and swap volume. Do not change these names. The root and swap volumes have the following specific minor device numbers:

- `rootvol` is minor device 0
- `swapvol` is minor device 1

As part of the encapsulation, LSM creates configuration files and changes the following system files and directories:

- `/etc/fstab`
- `/sbin/swapdefault` (if it exists)
- `/etc/sysconfigtab`
- `/etc/fdmns/*` (if using AdvFS)

5.3 Mirroring the Boot Disk

To maintain system availability, you must mirror the data important to running and booting your system. However, traditional LSM mirroring (described in Section 2.4), cannot be used to mirror the root disk. This is because mirroring, which is typically used to mirror data disks, cannot access some of the data required for a system reboot (such as the boot track). In addition, root disk mirroring is restricted such that the plexes on the mirrored disk must be accessible as root and swap partitions. Therefore, you must create a secondary root disk to make LSM continue to function if the boot disk fails.

The following list describes how to ensure that the secondary root disk has a boot track in the proper location and the root and swap partitions are accessible as partitions:

- If you are mirroring the whole boot disk, the target mirror disk must be as large as the boot disk. If you are mirroring only the root and swap partitions, the target mirror disk must be at least as large as the sum of the sizes of the root and swap partitions on the original boot disk, plus the length of the private region.

  You can use a disk that is the same as the original, for example you can use an rz26 to mirror another rz26, or you can use a disk whose physical geometry is different from that of the original disk. For example, an rz28 can mirror an rz26, because the rz28 is larger than the rz26.
- Choose a disk that is not already being used by LSM.
- Ensure that the target disk has a disk label, and that all partitions on the disk have an `fstype` of `unused`. If any partition is incorrectly marked as being in use, reinitialize the disk label (refer to the `disklabel`(8) reference page).

5.3.1 Mirroring rootvol and swapvol

To mirror the `rootvol` and `swapvol` volumes, you can use any of the following methods. Refer to Appendix C for detailed examples.

- Execute the `volrootmir` utility

  For example, the following command creates a mirror of all the volumes on `rz0` on disk `rz1`:

```
# volrootmir -a rz1
```

This creates mirrors for `rootvol`, and `swapvol`, and for volumes associated with the `/usr` and `/var` file systems if they exist on the boot disk.

- Select the `mirror volumes` option from the `voldiskadm` utility Support Menu.

  The utility prompts you to enter the root disk physical device name (such as `rz0`, which is the boot disk), and then prompts you to enter the disk device on which to mirror the root and swap volume (such as `rz1`). The disk, `rz1` in the previous example, must not have any partitions in use, and must not be already added to LSM.

- Invoke the Visual Administrator (`dxlsm`) and choose:

  rootvol → Basic Ops → Volume Ops → Add Mirror → Simple Mirror

  The option for a root volume causes a form to pop up. Enter the physical device name of the disk on which you want to mirror the `rootvol` and `swapvol` volumes.

Example 5-1 shows how to add the the boot disk to LSM, on a system that has disks already under LSM control. In this example, you encapsulate `rz0`, which is the boot disk, and you use `rz1` to mirror `rz0`. Note that this example also sets the default boot path so that the system first tries to boot from `rz0` and, if that fails, from `rz1`.

Example 5-1: Mirroring the Boot Disk

```
# volencap nconfig=0 rz0 1
# shutdown -h +2  2
>>> set bootdef_dev dka0 dka100  3
>>> boot 4
# volrootmir -a rz1  5
```

1 Encapsulate the entire boot disk.

2 Shut down the system.

3 At the console prompt, set `dka0` as the primary boot device and `dka100` as the alternate boot device.

4 Reboot the system.

5 Mirror the boot disk onto `rz1`.

5.3.2 Mirroring Other Volumes on the Root Disk

If you used the `volrootmir` command without the `-a` option, you mirrored only the root and swap partitions of the boot disk. In this case, the layout of the mirrored root disk partition is similar to the layout of the original root disk. For example, if the `g` and `h` partitions on the original root disk were in use for `/usr` and `/usr/staff` file systems, you can add the `g` and `h` partitions on the mirrored root disk to LSM. Then, you can add partitions `g` and `h` to LSM as nopriv disks, and mirror these partitions to obtain higher data availability for the `/usr` and `/usr/staff` file systems.

See Section 12.3.5 (mirroring using the graphical user interface) or Section 7.3.2 (mirroring using the command line interface), and the `volassist`(8) reference page for information about creating mirrors for existing volumes.

5.4 Evacuating a Boot Disk

Because the data on the book disk in `rootvol` and `swapvol` must be accessed as partitions, you cannot use traditional methods to evacuate a disk or move a subdisk. Hence, you should not use `volevac`, `volassist move`, or `volsd mv`. Instead, perform one of the methods described in Section 5.3.1 to create a new mirror. Later, remove the plex on the disk to be evacuated, as described in Section 5.5.

5.5 Removing Root and Swap Volume Mirrors

To remove mirrors associated with the root and swap volumes, enter the `volplex disassociate` command. For example:

```
# volplex dis rootvol-01

# volplex dis swapvol-01
```

When you disassociate a plex from `rootvol`, you should also disassociate the corresponding plex from `swapvol`.

You can remove the last plex of the root and swap volumes only by using the `volunroot` utility, as described in Section 5.6.

Refer to `voledit`(8) for information about removing `rootvol-01` and `swapvol-01` from the `rootdg` disk group.

5.6 Removing rootvol and swapvol

If you want the system to revert back to using UNIX partitions instead of LSM volumes for the root file system and primary swap partition, use the `volunroot` utility.

When `volunroot` is executed, the LSM configuration for `rootvol` and `swapvol` is removed, and the system is set up to use disk partitions for the root file system and the swap partition. When `volunroot` is executed with the `-a` option, it unencapsulates all volumes on the boot disk that map directly to a physical disk partition.

After executing `volunroot`, you must reboot the system, using the disk that was last used for `rootvol` and `swapvol.`

Note

> Rebooting from the disk that was used for `rootvol` and `swapvol` may require that you first change the default boot device in the console.

The following considerations apply to using the `volunroot` utility:

- Only one plex can exist in `rootvol` and `swapvol`.
- The volumes in `rootvol` and `swapvol` should be on the same physical disk.

Follow the steps shown here to execute the `volunroot` utility:

1. Enter the following command:

   ```
   # volunroot
   ```

   or

   ```
   # volunroot -a
   ```

2. Reboot the system, using the disk that was last used for `rootvol` and `swapvol`.

The `volunroot` utility performs the following actions:

- Removes `rootvol` and `swapvol` support for the root disk.

  If you use the `-a` option, removes support for all LSM volumes on the root disk.

- Restores the use of partitions on the root disk for the root file system and primary swap partition.

  If you use the `-a` option, restores the use of partitions on the root disk for all LSM volumes.

- Makes changes to the following system files:
  - `/etc/fstab`

    This file is changed to use disk partitions instead of LSM volumes.

- /etc/sysconfigtab

  This file is changed to remove LSM root and swap mirroring capabilities.

- /sbin/swapdefault

  If this file exists, it is changed to be a link to the swap partition.

- /etc/fdmns/*

  If you are using AdvFS, any domain directories that have disk partitions associated with the root disk are updated.

The volunroot utility changes the /etc/fstab and /sbin/swapdefault files to use the physical disk associated with the remaining mirror in rootvol.

Part 2: Disks and Volumes, Command Line Interface

Disk and Disk Group Operations 6

This chapter presents LSM commands used for managing disks and disk groups. For details on any of the commands mentioned in this chapter, refer to the appropriate reference pages.

The functions described in this chapter can also be performed using the LSM Support Operations menu interface or the graphical user interface `dxlsm`.

6.1 Commands Summary

Table 6-1 introduces the commands and utilities discussed in this chapter.

Table 6-1: Disk and Disk Group Commands

Command	Description
`voldisk`	Defines special disk devices, initializes information stored on disks that LSM uses to identify and manage disks, and performs additional special operations.
`voldiskadd`	Adds standard disks to LSM. This utility leads you through the process of initializing a new disk by displaying information and asking questions.
`voldg`	Creates new disk groups, adds and removes disks from disk groups, and enables (imports) or disables (deports) access to disk groups.
`voledit`	Creates, removes, and modifies LSM records.
`volevac`	Moves subdisks off the specified disk *medianame* to the specified destination disks *new_medianame*. If *new_medianame* operands are not specified, then any nonvolatile, nonreserved disks can be used as destination disks. Subdisks that are part of unmirrored striped plexes will be moved by moving the entire plex to a new location.
`volmake`	Creates subdisk, plex, and volume records.
`volprint`	Displays complete or partial information from records in LSM disk group configurations.

6.2 Working with Disks

LSM provides commands for adding and removing disks from LSM control, displaying information about the disks, renaming disks, and reserving disks so that the `volassist` utility does not allocate space from them.

Note that commands default to using the `rootdg` disk group. See Section 6.3 for information about disk groups.

6.2.1 Adding a Disk

To add a disk to LSM, use the following command syntax:

voldiskadd [*devname***]**

If you omit the device name from the command line, `voldiskadd` prompts you for an entry.

The following example shows the steps involved in adding the device `rz3` to LSM control, using the default initialize operation, disk group, and disk name.

See also Section 3.3 for other considerations when initializing and adding disks for LSM.

1. Enter the following command:

   ```
   # voldiskadd rz3
   ```

2. If the disk has never been used with LSM, the `voldiskadd` script gives you the option of encapsulating the disk partition instead of initializing it:

   ```
     Disk device rz3 appears to have never been used with LSM.
     If rz3 contains valid data then you should encapsulate the
     disk partition instead of initializing it as a new LSM disk.

   Do you wish to encapsulate rz3? [y,n,q,?] (default: n)
   ```

   To preserve the data on the disk, answer `y`.

   To initialize the disk as a new LSM disk, answer `n`.

   The remaining steps in this example assume a response of `n` to this prompt (the disk is being initialized, not encapsulated).

3. LSM prompts you for a disk group name:

   ```
     You can choose to add this disk to an existing disk group, to
     create a new disk group, or you can choose to leave the disk
     available for future 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>,list,q,?] (default: rootdg)
```

Press Return to assign the disk to the default disk group `rootdg`.

Otherwise, enter the name of the disk group to which you want the disk assigned or enter `none` to assign the disk as a spare.

4. LSM now prompts you for a disk name (unless you entered `none` for a disk group, because spare disks do not get named):

```
  You must now select a disk name for the disk.
  This disk name can be specified to disk removal,
  move, or replacement operations.  If you move the
  disk, such as between host bus adapters, the disk
  will retain the same disk name, even though it will
  be accessed using a different disk device address name.

Enter disk name [<name>,q,?] (default: disk01)
```

Press Return to assign the default disk name `disk01`, or enter a name of your choice.

5. LSM presents the following confirmation prompt:

```
  The requested operation is to initialize disk device rz3
  and to add this device to disk group rootdg as disk disk01.

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

Press Return to continue with the operation.

6. If the initialization is successful, LSM displays the following message:

```
 Disk initialization for rz3 completed successfully.

Goodbye.
```

If the disk cannot be initialized because the specified partition or an overlapping partition on the disk is open (that is, a partition is actively in use by UFS, AdvFS, LSM or swap), `voldiskadd` exits with this error message:

```
Initialization of disk device rz3 failed.
  Error:
rz3 or an overlapping partition is open.
Quitting ....
```

If the `fstype` in the disk label of the specified partition or an overlapping partition is set, LSM displays the following warning message to inform you that initializing the disk might destroy existing data:

```
rz3 is marked in use for 4.2BSD in the disklabel.
If you continue with the operation you can
possibly destroy existing data.

Would you like to continue?? [y,n,q,?] (default: n)
```

If you are sure that the disk partition has no valid data and that the partition can be added to LSM, you can ignore the warning message and answer `y` to the prompt. The `voldiskadd` utility then proceeds to initialize the disk partition and add it to LSM.

When you add new disks to existing LSM disk groups, the size of the private region on the new disks is the same size as on the other disks in the disk group. As you add more disks to a disk group, the `voldiskadd` utility reduces the number of configuration copies and log copies that are initialized for the new disks.

When you add disks to the `rootdg` disk group, the `voldiskadd` utility automatically adds the disks to the `/etc/vol/volboot` file.

6.2.2 Displaying Disk Information

Before you use a disk, you need to know if it has been initialized. You need to know if the disk is part of a disk group, since you cannot create volumes on a disk that is not part of a disk group. The `voldisk 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 LSM, use the following command:

```
# voldisk list
```

LSM returns the following display:

```
DEVICE    TYPE      DISK         GROUP        STATUS
rz8       sliced    rz8          rootdg       online
rz9d      simple    -            -            online
```

To display details on a particular disk defined to LSM (for example, `rz8`) enter:

```
# voldisk list rz8
```

6.2.3 Renaming a Disk

It is not necessary to give your disks special names; LSM gives the disk a default name when you add it to LSM control. The disk name is used by LSM to identify the disk's location or type.

If you want to change the disk name to reflect a change of ownership or use, enter the old and new disk names with the following `voledit` command syntax:

voledit rename *old_diskname new_diskname*

For example, to rename `disk03` to `disk01` follow these steps:

1. List the current disk names using `voldisk list`:

```
# voldisk list

DEVICE      TYPE     DISK       GROUP      STATUS
rz8         sliced   rz8        rootdg     online
rz9         sliced   disk03     rootdg     online
rz9d        simple   -          -          online
```

2. Rename the disk:

```
# voledit rename disk03 disk01
```

3. Confirm that the name change was successful:

```
# voldisk list

DEVICE      TYPE     DISK       GROUP      STATUS
rz8         sliced   rz8        rootdg     online
rz9         sliced   disk01     rootdg     online
rz9d        simple   -          -          online
```

6.2.4 Reserving Disks

By default, `volassist` operations allocate space from any disk that has free space. You may want to reserve some 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, enter the `voledit` command using the following syntax:

voledit set reserve=yes *diskname*

After you enter this command, `volassist` does not allocate space from the selected disk unless that disk is specifically mentioned on the `volassist` command line. For example, if disk `disk03` is reserved, use the `volassist` command as shown in the following command to create a 20 megabyte volume on `disk03`:

```
# volassist make vol03 20m disk03
```

However, the following command will not use `disk03` because the specific disk name is not included on the command line:

```
# volassist make vol04
```

Note that `disk03` is not used even if there is no free space on any other disk.

To turn off reservation of a disk, enter:

```
# voledit set reserve=no diskname
```

6.2.5 Removing a Disk

You can remove a disk to move it to another system or because the disk is failing or has failed. However, before removing a disk from the current system, you must move the subdisks to a different physical disk (see `volevac`(8)). If there is not enough free space to move the subdisks, take the following actions:

- Unmount the file system.
- Stop the volumes.
- Back up the volumes to tape (see Section 7.3.5).

  Alternatively, you can remove the volumes if they are no longer needed.

Removing a disk involves the following steps:

1. Remove the disk from its disk group, using the following `voldg` command syntax:

   voldg [-g *groupname***] rmdisk** *diskname*

   The *groupname* is the name of the disk group to which the disk belongs. The command uses the `rootdg` disk group by default. The *diskname* is the name of the disk to be removed.

   For example, to remove `disk01` from `rootdg`, enter:

   ```
   # voldg rmdisk disk01
   ```

2. After removing a disk from its disk group, remove it from LSM using the following command syntax:

   voldisk rm *devname*

   The *devname* is the name of the physical device that you are removing from LSM.

For example, to remove rz9d from LSM control, enter the command as follows:

```
# voldisk rm rz9d
```

6.2.6 Disabling a Disk

You must disable a disk before moving the physical disk device to another system. Also, if a disk becomes corrupted, you need to disable it and remove it.

To disable a disk, first remove the disk from its disk group. Then place the disk in an offline state using the following voldisk command syntax:

voldisk offline *devname*

For example, to disable the device rz8 (which has the disk name, disk01), enter:

```
# voldg rmdisk disk01
# voldisk offline rz8
```

Note

The device name is used with voldisk because the disk has no disk name after removal from the disk group.

6.3 Working with Disk Groups

A disk group is a collection of LSM disks that share a configuration database. Disk groups provide a method for partitioning the configuration database so that its size is manageable and so that database modifications affect identifiable sets of drives. Disk groups also allow LSM to operate with groups of physical disk media that can be moved between systems.

Disk groups are useful in situations where all data related to a particular set of applications or to a particular group of users needs to be made accessible on another system. For example:

- 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.

LSM supports a default disk group, rootdg, in which all volumes are created if no further specification is given. All commands default to rootdg.

You can create additional disk groups as needed for specialized disk management tasks.

6.3.1 Specifying Disk Groups

Most LSM commands allow you to specify a disk group using the `–g` option. For example, to create a volume named `mktvol` in a disk group named `mktdg`, you would use the following command:

```
# volassist -g mktdg make mktvol 50m
```

The volume device for this volume is `/dev/vol/mktdg/mktvol`.

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

```
# volassist make mktvol 50m mktdg01
```

This works for many commands as long as two disk groups do not have objects with the same name. For example, LSM allows you to create volumes named `mktvol` in both `rootdg` and in `mktdg`. If you do this, you must add `–g mktdg` to any command where you want to manipulate the volume in the `mktdg` disk group.

6.3.2 Displaying Disk Group Information

To use disk groups, you need to know the names of the groups and what disks belong to each group.

To display information on existing disk groups, enter the following command:

```
# voldg list
```

LSM returns the following message:

```
NAME         STATE    ID
rootdg       enabled  730344554.1025.harley
newdg        enabled  731118794.1213.harley
```

6.3.3 Displaying Free Space in a Disk Group

Before you add volumes and file systems to your system, you may want to make sure you have enough free disk space to adequately meet your needs.

To display free space for a disk group, specify a disk group name with the voldg free command using the following syntax:

```
# voldg -g disk_group free
```

If you do not specify a disk group, voldg free displays the free space in the default disk group, rootdg:

```
# voldg free

GROUP    DISK     DEVICE   TAG    OFFSET  LENGTH   FLAGS
rootdg   rz8      rz8      rz8    726400  102672   -
rootdg   disk01   rz9      rz9    0       102128   -
```

The LENGTH column gives the amount of free space, measured in terms of the number of 512-byte sectors.

6.3.4 Initializing a New Disk Group

To create a new disk group, use the voldiskadd command, which you also use to add a disk to LSM.

For example, to create the disk group newdg, do the following:

1. Enter the following command:

   ```
   # voldiskadd rz9d
   ```

   Because rz9d has already been initialized, LSM asks if you want to reinitialize it.

   ```
   Add or initialize a disk
   Menu: Logical Storage Manager/Disk/AddDisk

     Use this operation to add a disk to a disk group.
     You can select an existing disk group or create
     a new disk group. You can also initialize a disk
     without adding it to a disk group, which leaves
     the disk available for use as a replacement disk.
     This operation takes, as input, a disk device,
     for example rz3, a disk group (or none to leave
     the disk available for as a replacement disk).
     If you are adding the disk to a disk group, you
     will be asked to give a name to the disk.

     Disk device rz9d appears to have been initialized
     already. The disk is currently available as a
     replacement disk.

   Do you wish to reinitialize rz9d? [y,n,q,?] (default: y)
   ```

   Enter n to add the disk to a disk group without reinitializing it.

2. LSM prompts you for a disk group:

```
  You can choose to add this disk to an existing disk
  group, to create a new disk group, or you can choose
  to 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)
newdg
```

   Enter the new disk group name and press Return.

3. LSM displays the following message and prompt:

```
  There is no active disk group named newdg.

Create a new group named newdg? [y,n,q,?] (default: y)
```

   Press Return to continue.

4. LSM asks for a disk name:

```
  You must now select a disk name for the disk.  This disk
  name can be specified to disk removal, move, or
  replacement operations.  If you move the disk, such as
  between host bus adapters, the disk will retain the same
  disk name, even though it will be accessed using a
  different disk device address name.

Enter disk name [<name>,q,?] (default: disk02)
```

   Enter the disk name of your choice or press Return to select the default name.

5. LSM displays a confirmation window:

```
  The requested operation is to create a new disk group
  named newdg containing disk device rz9d.  The disk
  will be named newdg01 within the disk group.

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

   Press Return to continue.

6. Once the operation is complete, LSM returns the following display:

```
  Disk initialization for rz9d completed successfully.

Goodbye.
```

7. Enter the following command to see if the disk group was created:

   ```
   # voldisk list
   ```

   LSM displays the following output, which shows that `disk02` is a part of the disk group:

   ```
   DEVICE       TYPE      DISK         GROUP        STATUS
   rz8          sliced    rz8          rootdg       online
   rz9          sliced    disk01       rootdg       online
   rz9d         simple    disk02       newdg        online
   ```

When creating a new disk group, the `voldiskadd` utility prompts you to estimate the number of disks that will be used in the disk group. Based on the number you provide, the utility assigns default values for the private region size, nconfig, and nlog. The values assigned are similar to the values shown in Figure 3-1.

See also Section 3.3.2.2 for other considerations when initializing and adding disks for LSM.

6.3.5 Removing a Disk Group

To remove a disk group, close or unmount any volumes in the disk group and then run the following command:

```
# voldg deport diskgroupname
```

Deporting a disk group does not actually remove the disk group; instead, 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.

If you want 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 `rz3g` (attached with the disk name `disk04`) from disk group `rootdg` and add it to disk group `mktdg`, you would use the following commands:

```
# voldg rmdisk disk04
# voldg -g mktdg adddisk mktdg02=rz3g
```

This can also be done using `voldiskadm` by selecting item 3, `Remove a disk,` from the main menu, and then selecting item 1, `Add or initialize a disk.`

6.3.6 Moving Disks and Volumes between Disk Groups

You can move a set of LSM disks and the volumes configured on those disks into a new disk group, keeping the data in the LSM volumes intact. In the example shown here, the devices `rz17`, `rz19c`, `rz34`, and `rz36` are configured into the `rootdg` disk group as LSM disks `disk1`, `disk2`, disk21, and disk22 and contain LSM volumes named Volume1 and Volume2. The following steps show how to move these LSM disks and volumes from the rootdg disk group to a new disk group called staffdg, while preserving the data in Volume1 and Volume2:

1. Back up the Volume1 and Volume2 data and LSM configuration information as shown here:

   ```
   # tar cvf ...
   # volprint -ht > vp.info
   # voldisk list >> vp.info
   ```

2. The rootdg disk group must exist. If disk1, disk2, disk21, and disk22 are currently the only disks in rootdg, you need to add additional disks before you move them, so that rootdg will not be empty. Add at least two disks to rootdg. Note that for ASE configurations, the disks in rootdg must be on a local (that is, not shared) I/O bus. Use the mount, disklabel, and swapon-s commands to find free partitions or disks on the local bus.

   If a disk is unused, add the whole disk, for example rz4, to LSM. Otherwise, you can add a partition, for example rz4h. Use the voldiskadd command to add the disks to LSM as shown here:

   ```
   # voldiskadd rz4
   # voldiskadd rz6g
   ```

3. Remove references to disk1, disk2, disk21, and disk22 from the /etc/vol/volboot file. To do this, first use the voldisk list command to get the DISK to DEVICE mapping, and the voldctl list command to see if any of those disks are present. Then use the voldctl rm disk command to remove the disks that are in the /etc/vol/volboot file, but are not in rootdg, as shown here:

   ```
   # voldisk list
   # voldctl list
   # voldctl rm disk rz17 rz19c rz34 rz36
   ```

4. Save the Volume1 and Volume2 configurations into a description file:

   ```
   # volprint -mvpsh Volume1 > v1.df
   # volprint -mvpsh Volume2 > v2.df
   ```

5. Unmount the volumes, if mounted:

   ```
   # mount
   # umount /dev/vol/rootdg/Volume1
   ```

6. Remove `Volume1` and `Volume2` from `rootdg`:

   ```
   # voledit -rf rm Volume1 Volume2
   ```

7. Save the names of the LSM disk name mappings, then remove the disks from `rootdg`:

   ```
   # voldisk list > list.out
   # voldg rmdisk disk1 disk2 disk21 disk22
   ```

8. Create the new disk group (for example, `staffdg`) and add the disks. Use the output in the columns labeled DISK and DEVICE from the `voldisk list` command shown previously. For example, if `disk1`, `disk2`, `disk21`, and `disk22` were on `rz17`, `rz19c`, `rz34`, and `rz36`, respectively, use these commands:

   ```
   # voldg init staffdg disk1=rz17
   # voldg -g staffdg adddisk disk2=rz19c disk21=rz34 \
   disk22=rz36
   ```

9. Restore the LSM configurations for `Volume1` and `Volume2` into the new disk group; note that `v1.df` and `v2.df` are the description files that were saved earlier in this example.

   ```
   # volmake -g staffdg -d v1.df
   # volmake -g staffdg -d v2.df
   ```

10. Start the volumes:

    ```
    # volume -g staffdg start Volume1 Volume2
    ```

11. `Volume1` and `Volume2` now have different device paths, as shown here.

    Old path:
    `/dev/vol/rootdg/Volume1`
    `/dev/rvol/rootdg/Volume2`

    New path:
    `/dev/vol/staffdg/Volume1`
    `/dev/rvol/staffdg/Volume2`

    Update the `/etc/fstab` file for UFS file systems and the `/etc/fdmns` file for AdvFS file systems, to use the new paths. For AdvFS, the commands to do this are as follows:

    ```
    # cd /etc/fdmns/domain_name
    # ls -l | grep Volume1 Volume2
    # rm -f Volume1 Volume2
    # ln -s /dev/vol/staffdg/Volume1 Volume1
    # ln -s /dev/vol/staffdg/Volume2 Volume2
    ```

    The data in `Volume1` and `Volume2` is now accessible as part of the new `staffdg` disk group.

6.3.7 Moving Disk Groups Between Systems (Deporting and Importing)

Disk groups 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 without having to respecify the configuration.

The following steps describe how to move a disk group between systems.

1. On the first system, close and unmount all volumes in the disk group, then disable local access to the disk group with the following commands:

   ```
   # voldg deport diskgroupname
   # voldisk rm ...
   ```

2. Move all the disks to the second system.
3. Make the disks accessible with the command:

   ```
   # voldisk define  diskname ...
   ```

4. Enable local access to the disk group on the second system with the following command:

   ```
   # voldg import diskgroupname
   ```

5. Start all volumes in the disk group with the following command:

   ```
   # volrecover -g diskgroupname -sb
   ```

You may want to move disks from a system that has crashed. In this case, you will not be able to 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. If you move disks from a system that has crashed or failed to detect the group before the disk is moved, the locks stored on the disks will remain and must be cleared. The system returns the following error message:

```
voldg:disk group groupname: import failed: Disk
        in use by another host
```

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

```
# voldisk clearimport diskdevicename ...
```

Be careful when using this command in the ASE environment.

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

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

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

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

```
# voldg -f import diskgroupname
```

6.3.8 Displaying Free Space in a Configuration Database

You can also use the `voldg list` command to display information about the configuration database.

The configuration database size and the log size of a disk group corresponds to the smallest configuration database size and log size of any disk in the disk group.

To display the amount of free space, enter the following command:

```
# voldg list rootdg
Group:     rootdg
dgid:      783105689.1025.lsm
import-id: 0.1
flags:
config:    seqno=0.1112 permlen=173 free=166 templen=6 loglen=26
config disk rz13 copy 1 len=173 state=clean online
config disk rz13 copy 2 len=173 state=clean online
config disk rz11g copy 1 len=347 state=clean online
config disk rz10g copy 1 len=347 state=clean online
log disk rz11g copy 1 len=52
log disk rz13 copy 1 len=26
log disk rz13 copy 2 len=26
log disk rz10g copy 1 len=52
```

In this display, the ‘‘free=166’’ record indicates that there is enough space in the `rootdg` disk group to create 166 additional LSM configuration records.

In addition, disk `rz13` has a configuration database size of 173 sectors, and disks `rz11g` and `rz10g` each have a configuration database size of 347 sectors. The configuration database size of the `rootdg` disk group is 173 sectors, which corresponds to the smallest configuration database.

6.3.9 Increasing Free Space in a Configuration Database

When the disk group runs out of space in the configuration database, LSM displays the following message when creating an LSM record.

```
# volmake plex pl1000
volmake: No more space in disk group configuration
```

When you run out of space, you can use the `voldisk` command to increase the size of the configuration database on each disk that is smaller than the current disk group configuration. In the display, note that disk `rz13` has a configuration database size of 173 sectors for each copy of the database. By reducing the number of configuration copies on `rz13` from 2 to 1, you can increase the size of its configuration database. For example:

```
# voldisk moddb rz13 nconfig=1 nlog=1
```

When reducing the number of configuration copies, use caution to make sure that there are sufficient copies of the database available for redundancy.

Note

> Digital recommends that you create a backup copy of the configuration database before changing the free space.
>
> Refer to Appendix C for information about creating a backup configuration copy using the `volsave` utility .

Volume and File System Operations 7

This chapter presents LSM commands used for managing volumes and file systems. For details on any of the commands mentioned in this chapter, refer to the appropriate reference pages.

The functions described in this chapter can also be performed using the LSM Support Operations menu interface (`voldiskadm`) or the graphical user interface (`dxlsm`).

7.1 Commands Summary

Table 7-1 introduces the commands and utilities discussed in this chapter.

Table 7-1: Volume Commands

Command	Description
`volassist`	A one-step utility that finds space for and creates simple volumes, adds simple plexes to existing volumes, extends and shrinks existing volumes, provides for the migration of data from a specified disk, and provides facilities for the online backup of existing volumes. The command is supplied a keyword that selects the action to perform.
`voledit`	Sets and changes various attributes for LSM configuration records that do not depend upon volume usage types. (See the `volume` command utility to set attributes that are dependent upon usage types.) Each invocation can be applied to only one disk group at a time.
`volmake`	Creates subdisk, plex, and volume records.
`volplex`	Performs LSM operations on plexes and volume-and-plex combinations. The `volplex` utility takes several operands. The first operand is a keyword that determines the specific operation to perform. The remaining operands specify the configuration objects to which the operation is to be applied. Each operation can be applied to only one disk group at a time.
`volprint`	Displays information about volumes, plexes, and subdisks.
`volrecover`	Recovers plexes and volumes after disk replacement.

Table 7-1: (continued)

Command	Description
`volrestore`	Restores an LSM configuration, using the information in the files saved with the `volsave` command.
`volsave`	Saves information about an LSM configuration in a set of files called a description set.
`volume`	Performs LSM operations on volumes, including initialization, set the read policy (round robin, preferential, select a default policy based on plex association), enable disabled or detached volumes, stop volumes, and change volume characteristics.

In addition, this chapter describes the commands for making and mounting UFS and AdvFS file systems on an LSM volume.

7.2 Displaying Volume Information

If you are administering a volume created by someone else, you may want to know how this volume is configured.

Use the `volprint` command to display the volume, plex, and subdisk record information for all volumes, as shown here:

```
# volprint -ht
```

For example:

```
DG NAME     GROUP-ID
DM NAME      DEVICE        TYPE     PRIVLEN  PUBLEN PUBPATH
V  NAME      USETYPE       KSTATE   STATE    LENGTH READPOL...
PL NAME      VOLUME        KSTATE   STATE    LENGTH LAYOUT...
SD NAME      PLEX          PLOFFS   DISKOFFS LENGTH DISK-MEDIA...

dg rootdg      730344554.1025.harley

dm rz8         rz8         sliced   512    829072 /dev/rrz8e
dm disk01      rz8d        simple   512    202240 /dev/rrz8d
dm disk02      rz9         sliced   512    202240 /dev/rrz9e

pl volmir-02   -           DISABLED -      10240  CONCAT -  RW
sd disk02-02   volmir-02   0        16384  10240  disk02   rz9

v  homevol     fsgen       ENABLED  ACTIVE 275184 SELECT -
pl homevol-01  homevol     ENABLED  ACTIVE 275184 CONCAT -  RW
sd rz8-04      homevol-01  0        410256 275184 rz8      rz8

v  rootvol     root        ENABLED  ACTIVE 314496 SELECT -
pl rootvol-01  rootvol     ENABLED  ACTIVE 314496 CONCAT -  RW
sd rz8-01      rootvol-01  0        95760  314496 rz8      rz8
```

```
v  snapvol     fsgen       ENABLED  -      20480  ROUND  -
pl voltest-03  snapvol     ENABLED  ACTIVE 20480  CONCAT -  RW
sd disk01-01   voltest-03  0        143088 20480  disk01   rz8d

v  standvol    gen         ENABLED  ACTIVE 31248  SELECT -
pl standvol-01 standvol    ENABLED  ACTIVE 31248  CONCAT -  RW
sd rz8-03      standvol-01 0        0      31248  rz8      rz8

v  swapvol     swap        ENABLED  ACTIVE 64512  SELECT -
pl swapvol-01  swapvol     ENABLED  ACTIVE 64512  CONCAT -  RW
sd rz8-02      swapvol-01  0        31248  64512  rz8      rz8

v  volcat      fsgen       ENABLED  ACTIVE 20480  SELECT -
pl volcat-01   volcat      ENABLED  ACTIVE 20480  CONCAT -  RW
sd rz8-06      volcat-01   0        705920 20480  rz8      rz8

v  volspec     fsgen       ENABLED  ACTIVE 6144   SELECT -
pl volspec-01  volspec     ENABLED  ACTIVE 6144   CONCAT -  RW
sd disk02-01   volspec-01  0        10240  6144   disk02   rz8d

v  volzebra    fsgen       ENABLED  ACTIVE 20480  SELECT volstripe

pl volzebra-01 volzebra    ENABLED  ACTIVE 20480  STRIPE 128 RW
sd disk01-02   volzebra-01 0        102128 10240  disk01    rz9
sd rz8-01      volzebra-01 10240    0      10240  disk02    rz9

v  voltest     fsgen       ENABLED  ACTIVE 20480  SELECT -
pl voltest-01  voltest     ENABLED  ACTIVE 20480  CONCAT -   RW
sd rz8-05      voltest-01  0        685440 20480  rz8       rz8
pl voltest-02  voltest     ENABLED  ACTIVE 20480  CONCAT -   RW
sd disk01-04   voltest-02  0        122608 20480  disk01   rz8d
```

This example uses the following abbreviations:

`dg`	Disk group
`dm`	Disk
`pl`	Plex
`sd`	Subdisk
`v`	Volume

You can list information related to volumes under LSM control, including the name of the volume, its usage type, state, length, user and group IDs, and mode.

To display information on a particular volume, specify the volume name on the command line:

```
# volprint -t volume_name
```

For example, to display the information about a volume named `volspec`, enter:

```
# volprint -t volspec
DG NAME  GROUP-ID
DM NAME  DEVICE   TYPE     PRIVLEN  PUBLEN  PUBPATH
V  NAME  USETYPE  KSTATE   STATE    LENGTH  READPOL PREFPLEX
PL NAME  VOLUME   KSTATE   STATE    LENGTH  LAYOUT ST-WIDTH MODE
SD NAME  PLEX     PLOFFS   DISKOFFS LENGTH  DISK-MEDIA ACCESS

v volspec fsgen   ENABLED  ACTIVE   6144    SELECT   -
```

To display the plexes for this volume, use the following command:

```
# volprint –e ’assoc="volspec"’
```

To list information on all volumes, use the following command:

```
# volprint –vt
```

To show detailed information for all volumes, use the following command:

```
# volprint -vl
```

7.3 Volume Operations Using volassist

The `volassist` command enables you to perform a variety of volume operations, such as creating and mirroring volumes. This utility is easy to use and is LSM’s most powerful program.

To get help on using `volassist`, display the online help information as follows:

```
# volassist help

volassist - Perform simple general administrative actions

Usage: volassist [-b] [-g diskgroup] [-U usetype] [-d file] keyword...

Recognized keywords:
    make volume_name length [options]
    mirror volume_name [options]
    move volume_name [options]
    [-f] growto volume_name new_length [options]
    [-f] growby volume_name length_change [options]
    [-f] shrinkto volume_name new_length [options]
    [-f] shrinkby volume_name length_change [options]
    snapstart volume_name [options]
    snapwait volume_name
    snapshot volume_name snapshot_name
    help [debug | flags | options]
```

Note

You can control the behavior of the `volassist` command by specifying parameters on the command line or in the defaults file `/etc/defaults/volassist`. See Section 8.4.5 for an example of this file and for instructions on creating and editing it.

7.3.1 Creating a Volume

A volume is a logical disk device on which file systems and databases can be created.

To create a volume using system defaults, use the following command:

```
# volassist make volume_name length
```

For example, to create the volume `voldef`, enter:

```
# volassist make voldef 10m
```

This creates a 10MB volume named voldef.

7.3.1.1 Creating a Concatenated Volume

A concatenated volume is a volume that uses one or more sections of disk space. On a fragmented disk, concatenation allows you to put together a volume larger than any individual section of free disk space available.

To create a concatenated volume, use the following command:

```
# volassist make volume_name length layout=concat
```

For example, to create the concatenated volume `volcat`, enter:

```
# volassist make volcat 10m layout=concat
```

You can set concatenation as the default in the `volassist` defaults file `/etc/defaults/volassist`. This file is described in Section 8.4.5.

7.3.1.2 Creating 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, because it takes space from more than one disk.

To create a spanned volume, use the following command:

```
# volassist make volume_name length
```

The concatenated, spanned layout is the default.

For example, to create the spanned volume `volspan`, enter:

```
# volassist make volspan 1000m
```

7.3.1.3 Creating a Striped Volume

A striped volume consists of a number of equal sized subdisks, each located on a separate disk drive. To create a striped volume, use the following command:

```
# volassist make volume_name length layout=stripe
```

For example, to create the striped volume `volzebra`, enter:

```
# volassist make volzebra 10m layout=stripe
```

This command line creates a volume with the default stripe width on the default number of drives.

7.3.1.4 Creating a Volume on a Specific Disk

LSM automatically selects the disk or disks each volume will reside on, unless you specify otherwise. If you want a volume to reside on a specific disk, you must designate the disk for LSM.

To create a volume on a specific disk, use the following command:

```
# volassist make volume_name length diskname [...]
```

For example, to create the volume `volspec` on `disk03`, enter:

```
# volassist make volspec 3m disk03
```

You can specify more than one disk.

7.3.2 Mirroring a Volume

A plex contains a copy of the volume's data. A volume must have two plexes for the data to be mirrored. The mirrored copy is not stored on the same disk as the original copy of the volume. Mirroring a volume assures you that the data in that volume will not be lost if one of your disks fails.

To create a new volume with a plex, use the following command:

```
# volassist make volume_name length mirror=yes
```

For example, to create the mirrored volume, `volmir`, enter:

```
# volassist make volmir 5m mirror=yes
```

To create a plex for an existing volume, use the following command:

```
# volassist mirror volume_name
```

For example, to create a plex of the volume `voltest`, enter:

```
# volassist mirror voltest
```

7.3.3 Extending a Volume

If the volume is not large enough for the amount of data that needs to be stored in it, you need to extend the volume's length.

To extend a volume to a specific length, use the following command:

```
# volassist growto volume_name length
```

For example, to extend `volcat` to 2000 512-byte sectors, enter:

```
# volassist growto volcat 2000
```

To extend a volume by a specific length, use the following command:

```
# volassist growby volume_name length
```

For example, to extend `volcat` by 100 sectors, enter:

```
# volassist growby volcat 100
```

Notes

File systems such as AdvFS and UFS cannot currently take advantage of an enlarged volume.

You cannot use the `volassist growto` and `volassist growby` commands with striped volumes.

An extend operation will fail if a volume is open when you perform the operation. To override the failure, use the `-f` force option with `volassist`.

7.3.4 Shrinking a Volume

If you find that a volume is much larger than needed, you can shrink its size. However, be aware that shrinking a volume containing data (perhaps in the form of a file system or database) can result in the loss of any data residing on the part of the volume that is removed.

To shrink a volume to a specific length, use the following command:

```
# volassist shrinkto volume_name length
```

This command can be safely used on empty volumes.

For example, to shrink `volcat` to 1300 sectors, enter:

```
# volassist shrinkto volcat 1300
```

To shrink a volume by a specific length, use the following command:

```
# volassist shrinkby volume_name length
```

For example, to shrink `volcat` by 300 sectors, enter:

```
# volassist shrinkby volcat 300s
```

Notes

File systems such as AdvFS and UFS cannot currently take advantage of an enlarged volume.

You cannot use the `volassist shrinkto` and `volassist shrinkby` commands with striped volumes.

A shrink operation will fail if a volume is open when you perform the operation. To override the failure, use the `-f` force option with `volassist`.

7.3.5 Reducing Backup Down Time Using volassist

LSM provides the ability to perform snapshot backups of volume devices. This capability is provided through `volassist` and other utilities. There are various procedures for doing backups, depending upon 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 it is smaller, an incomplete copy results.

The recommended approach to volume backup involves the use of the `volassist` utility. The `volassist` procedure is convenient and relatively simple.

The `volassist snapstart`, `snapwait`, and `snapshot` operations provide a way to do online backup of volumes with minimal interruption of data change and access activity.

The `snapstart` operation 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 plex. The end of the update procedure is signified by the new snapshot plex changing its state to SNAPDONE. This change can be tracked by the `volassist snapwait` operation, which waits until at least one of the `snapshot` plexes changes its state to SNAPDONE. If the attach process fails, the snapshot plex is removed and its space is released.

Once the snapshot plex is synchronized, it continues being updated until it is detached. The system administrator can then select a convenient time at which to create a snapshot volume as an image of the existing volume. The system administrator 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 plex is long and indefinite, in contrast to the brief amount of time that it takes to create the snapshot volume.

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

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

Follow these steps to perform a complete backup using `volassist`:

1. Create a snapshot plex for a volume as follows:

   ```
   # volassist snapstart volume_name
   ```

2. When the snapstart operation is complete and the plex is in a SNAPDONE state, select a convenient time to complete the snapshot operation. Inform users of the upcoming snapshot and warn them to save files and refrain from using the system briefly during that time.

   Notes

   For UFS volumes, Digital recommends the file system be unmounted briefly to ensure the snapshot data on disk is consistent and complete.

   For volumes used in AdvFS domains, Digital recommends an AdvFS backup method such as cloning.

3. Create a snapshot volume that reflects the original volume as follows:

   ```
   # volassist snapshot volume_name temp_volume_name
   ```

   After the snapshot is completed, you can resume normal use of the volume. For example, if the volume contained a UFS, it can be remounted.

4. Use `fsck`, or some utility appropriate to the application running on the volume, to clean the temporary volume's contents. For example:

   ```
   # fsck -p /dev/rvol/temp-volume-name
   ```

5. Copy the temporary volume to tape, or to some other appropriate backup media. For example, a UFS on the temporary volume could be backed up on the temporary volume with the following command:

   ```
   # dump 0 /dev/rvol/temp-volume-name
   ```

6. After the backup has been completed, remove the temporary volume as follows:

   ```
   # volume stop temp_volume_name
   # voledit -r rm temp_volume_name
   ```

Note

The `-r` option of the `voledit` command removes multiple objects. Exercise caution when using it.

7.4 Saving an LSM Configuration

LSM provides utilities to save and restore configuration information, so that you can recreate your LSM configuration, if necessary.

The `volsave` command saves information about an LSM configuration in a set of files called a description set. By default, the description set is saved to a timestamped directory in `/usr/var/lsm/db` (for example, `/usr/var/lsm/db/LSM.19951226203620.skylark`). Or, you can specify a directory of your choice by using the `-d` option with `volsave` (as described in Section C.21).

The description set contains the following files:

- `allvol.DF`

  A `volmake` description file for all volumes, plexes, and subdisks in a disk group. The `volsave` utility creates a separate subdirectory and description file for each disk group on the system.

- `voldisk.list`

  A description of the disks. This file is the output of the `voldisk list` command.

- `volboot`

  The contents of the `/etc/vol/volboot` file.

- `header`

A header file for the description set, containing a checksum, a magic number, the date of the file's creation, and the version of the `volsave` command.

The `volsave` command does not save the data in LSM volumes; it only saves information about the LSM configuration. You should back up the description set, as you would any other files.

See Section C.21 for detailed examples of saving an LSM configuration.

7.5 Restoring an LSM configuration

The `volrestore` command restores an LSM configuration, using the information that you saved with the `volsave` command. If you enter the `volrestore` command with no options, `volrestore` attempts to restore all disk groups. If you use the `-i` (interactive) option, `volrestore` prompts you before restoring each disk group.

Before the `volrestore` command restores the LSM configuration, it validates the checksum that is part of the description set.

By default, the `volrestore` command restores the whole configuration, using the description set in the directory under `/usr/var/lsm/db` that has the latest timestamp. You can specify options to the command to use a different directory and to restore a specific volume or disk group. For example, this command restores only the volume named `myvol01` in the `staffdg` disk group:

```
# volrestore -g staffdg -v myvol01
```

When it restores a specific disk group, the `volrestore` command attempts to reimport the disk group based on configuration information on disks that belong to that disk group. If the import fails, `volrestore` recreates the disk group by reinitializing all disks within that disk group and recreating all volumes, unassociated plexes, and unassociated subdisks, based on information in the `volmake` description file, `allvol.DF`

Notes

The `volrestore` command does not restore volumes associated with the `root`, `/usr`, and `/var` file systems and the primary swap area. These partitions must be reencapsulated to use LSM volumes.

Before using `volrestore` in ASE or clusters, please refer to your ASE or clusters documentation.

When you restore a complete LSM configuration, `volrestore` attempts to reenable the `vold` daemon based on all `rootdg` disks in the saved copy of the `volboot` file. If the complete LSM configuration needs to be restored, you can use the -i (interactive) option with volrestore. The volrestore command prompts you before restoring each file, enabling you to skip specific disk groups.

If vold cannot be enabled, you are given the option of recreating the rootdg disk group and any other disk groups using the other files in the saved LSM description set. The rootdg disk group is recreated first, and vold is put in the enabled mode. Then, the other disk groups are enabled. The disk groups are recreated by first attempting to import them based on available disks in that disk group. If the import fails, the disk group is reinitialized and all volumes in that disk group are also recreated based on the volmake description files.

When volumes are restored using the volmake description file, the plexes are created in the "DISABLED EMPTY" state. The volrestore command does not attempt to start or enable such volumes. You must use volmend or volume to set the plex states appropriately before starting the volume. The volrestore command warns you to check the state of each disk associated with a volume before using volmend to set plex states; to carefully find out which disks in the LSM configuration could have had failures since saving the LSM configuration; and to use volmend to mark plexes on those disks to be "STALE." In addition, any plex that was detached or disabled at any point during or after the LSM configuration was saved should be marked "STALE" using volmend.

See Chapter 14 for information on how to resolve conflicts that may occur when restoring a configuration.

See Section C.23 for detailed examples of restoring a configuration.

7.6 Volume Operations Using Individual Commands (Bottom-Up Approach)

This section shows how to use the volume command and other low-level LSM commands to do the following tasks:

- Create a volume
- Initialize a volume
- Change volume attributes
- Start and stop volumes
- Mirror a volume
- Remove a mirrored plex
- Remove a volume

The LSM reference pages provide detailed descriptions about the commands used to perform volume operations.

7.6.1 Creating a Volume with volmake

Volumes can be created with either the `volassist` or `volmake` utility.

The length of a new volume can be specified in sectors, megabytes, or kilobytes. The unit of measure is indicated by adding the appropriate suffix to the length (`s`, `m`, or `k`). If no unit is specified, sectors are assumed.

To create a volume using `volmake`, use the following command:

```
# volmake –U usage_type vol volume_name len=length  \
plex=plex_name,...
```

Either the `plex=` parameter or the `len=` parameter can be omitted. If the `len=` parameter is specified with the `plex=` parameter, then the volume created from an existing plex takes the length of that plex. If the `len=` parameter is specified without the `plex=` parameter, then at least one plex of the appropriate length must later be specified.

The following examples show commands for creating an `fsgen-type` volume called `vol01`:

```
# volmake –Ufsgen vol vol01 len=100000
```

or

```
# volmake vol vol01 use_type=fsgen plex=vol01-01,vol01-02
```

Note that the first command shown above creates a volume that has no plex.

The usage type for a volume can be specified in either of two ways: `–Ufsgen` or `use_type=fsgen`. If a length is not specified (for example, `len=100000`) or associated plexes are not identified (for example, `plex=vol01-01,vol01-02`), the volume length will be zero.

Instead of specifying parameters on the command line, you can use a `volmake` description file to create a volume, as well as associated subdisks and plexes, by using the following command:

```
# volmake –d description_file
```

For detailed information about how to use `volmake`, and an example of the `volmake` description files, refer to Section 8.4.3 or to the `volmake`(8) reference page.

7.6.2 Initializing Volumes

LSM automatically sets the states of plexes and volumes based on usage type. The `volume start` command performs a default volume initialization on a newly created volume; the `volume init` command is used for spanned volumes. When a multiple-plex volume is first configured and needs to be started, you must initialize the volume manually. Use a command like the following:

```
# volume init how volume_name [plex_name]
```

The *how* variable determines what the initialization does and what condition the volume and plexes have after the volume has been initialized. The most common form of manual initialization is setting the state of the volume and one of its plexes to CLEAN. To set the state of a volume and plex to CLEAN, use the following command:

```
# volume init clean volume1 plex
```

The following commands illustrate a specific example of how to set a volume and plex to the CLEAN state during manual initialization:

```
# volmake -Ufsgen home plex=home-1,home-2
# volume init clean home home-2
# volume start home
```

In this example, the system administrator has created a new two-plex volume, and then realized that the data that existed on plex `home-2` was the data that needed to be mirrored onto the plex labeled `home-1`. By setting the state of `home-2` to CLEAN, LSM will correctly copy the data onto (revive) `home-1` during the `volume start` operation.

To make all the plexes in a volume ACTIVE and enable the volume, use the following command:

```
# volume init active volume_name
```

To enable a volume to temporarily allow I/O to dump the contents of a tape into a volume, use the following command:

```
# volume init enable volume_name
```

This enables the volume and all its plexes, but leaves the plex utility state as EMPTY.

After the I/O is complete, issue the `volume init active` version of the command.

Note

> When LSM is restarted (for example, after a system reboot) a volume that has been initialized using `init enable` will not start.

To zero out all the plexes, set them to ACTIVE, and enable the volume, use the following command:

```
# volume init zero volume_name
```

Caution

For both mirrored and nonmirrored volumes, this action destroys any existing data residing on the volume. Do not use the `zero` option unless you are sure that no critical data exists on any plex in the volume. Most initializations will destroy data on some plexes. To preserve data on a volume, use either `volume init active` or `volume init clean`, as appropriate.

7.6.3 Changing Volume Attributes

Volume attributes such as read policy, error policies, ownership, permissions, and the values in the comment and utility fields for existing volumes can be changed. These attributes are changed whenever the use of the volume or users' needs change. Two LSM commands are used to set volume attributes:

- The `voledit` command sets those attributes that are not usage-type-dependent.
- The `volume` command sets only those attributes that are usage-type-dependent.

Use the following command syntax to set volume attributes:

```
# volume set field=value ... volume_name ...

# voledit set field=value ... volume_name ...
```

Table 7-2 describes the attributes that each command can set.

Table 7-2: Setting Volume Attributes

Command	Attribute	Description
`voledit`		The comment field
	`tutil0`, `tutil1`, `tutil2`	Descriptive string of volume contents
	`putil0`, `putil1`, `putil2`	
	`fstype`	String indicating file system type
	`writeback`	Boolean (on/off) specifying read error correction mode

Table 7-2: (continued)

Command	Attribute	Description
	`user`	Owner of volume
	`group`	Group of volume
	`mode`	Permission mode for volume
`volume`	`len`	Numeric length of volume
	`log type`	(blkno/undef) specifier of block-change logging mode for volume
	`log`	Length of the block-change logging log
	`start opts`	Options to be executed to the `volume start` operation

For example, to change the owner of `vol01` to `susan`, the group to `staff`, and the permissions to read/write for owner, group, and other:

```
# voledit set user=susan group=staff mode=0666 vol01
```

7.6.3.1 Resizing a Volume with the volume Command

Resizing a volume is an instance of changing volume attributes that can be handled using either `volassist` or `volume`.

Striped volumes cannot be resized.

The new size of a volume can be specified in sectors, megabytes, or kilobytes. The unit of measure is indicated by adding the appropriate suffix to the length (`s`, `m`, or `k`). If no unit is specified, sectors are assumed.

To change the length of a volume using `volume set`, use the following command:

```
# volume set len=value ... volume_name ...
```

For example, to change the length to 100000 sectors, use the following command:

```
# volume set len=100000 vol01
```

Note

Note the following when using the `volume` command:

- The `volume set len` command cannot increase the size of a volume unless the needed space is available in the plexes of the volume.
- When a volume's size is reduced using the `volume set len` command, the freed space is not released into the free space pool.

7.6.3.2 Changing Volume Read Policy

LSM offers the choice of three read policies:

- The `round` policy reads each plex in turn in round-robin fashion.
- The `prefer` policy reads preferentially from a plex that has been labeled as the preferred plex. The read policy can be changed from `prefer` to `round` (or vice versa in the case of `prefer`) or to a different preferred plex.
- The `select` policy chooses a default policy based on plex association to the volume.

The `volume rdpol` command sets the read policy for a volume. To set a read policy, use one of the following commands:

```
# volume rdpol round volume_name
```

or

```
# volume rdpol prefer volume_name preferred_plex_name
```

For example, the command line to set the read policy for volume `vol01` to a round-robin read looks like this:

```
# volume rdpol round vol01
```

The command line to set the policy for the same volume to read preferentially from the plex `vol01-02` looks like this:

```
# volume rdpol prefer vol01 vol01-02
```

7.6.4 Starting and Stopping Volumes

Like mounting and unmounting a file system, starting and stopping a volume affects its availability to the user. Starting a volume changes its state and makes it available for use. Stopping a volume makes it unavailable.

Starting a volume changes the volume state from DISABLED or DETACHED to ENABLED. The success of this operation 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:

```
# volrecover -s volume_name ...
```

To start all DISABLED volumes, use the following command:

```
# volrecover -s
```

Stopping a volume changes the volume state 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:

```
# volume stop volume_name ...
```

For example, the command line to stop a volume labeled `vol01` looks like this:

```
# volume stop vol01
```

To stop all ENABLED volumes, use the following command:

```
# volume stopall
```

If all plexes of the volume become STALE, put the volume in maintenance mode so that the plexes can be looked at 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:

```
# volume maint volume_name
```

To assist in choosing the revival source plex, list the unstarted volume and display its plexes.

To take plex `vol01-02` off line, use the following command:

```
# volmend off vol01-02
```

For ENABLED volumes, save a step by using `volplex att` without first invoking `volmend on`. This command works on an OFFLINE plex of an ENABLED volume (designated as `vol01` in the example):

```
# volplex att vol01 vol01-02
```

The `volmend` utility can change the state of an OFFLINE plex of a DISABLED volume to STALE, after which running `volume start` on the volume would revive the plex. To put a plex labeled `vol01-02` in the STALE state, use the following command:

```
# volmend on vol01-02
```

To make other state changes in a plex or a volume, refer to the subsequent sections on volume recovery options.

7.6.5 Mirroring a Volume with the volplex Command

To mirror an existing volume using individual commands, first create a plex, then associate it to a volume. Use the following command syntax:

```
# volmake plex plex_name sd=subdisk_name ...
# volplex att volume_name plex_name
```

You can also use the `volassist` utility to mirror existing volumes.

7.6.6 Removing a Plex (Mirror)

A plex contains a copy of the volume's data. A volume is mirrored when there are two or more plexes attached to the volume. If you no longer want to mirror a volume, you must disassociate the extra plex copy or copies.

To remove a plex, first disassociate the plex from the volume, then remove the plex and any associated subdisks. Use the following command syntax:

```
# volplex dis plex_name

# voledit -r rm plex_name
```

For example, the following output of `volprint` shows the volume `volmir` as a mirrored volume with attached plexes `volmir-01` and `volmir-02`.

```
# volprint -h volmir

TYPE NAME         ASSOC        KSTATE     LENGTH COMMENT
vol  volmir       fsgen        ENABLED     10240
plex volmir-01    volmir       ENABLED     10240
sd   disk01-03    volmir-01    -           10240
plex volmir-02    volmir       ENABLED     10240
sd   disk02-02    volmir-02    -           10240
```

To ''unmirror'' the `volmir` volume, you would need to remove the volume `volmir-02` as follows:

1. Disassociate `volmir-02` from the parent volume:

   ```
   # volplex dis volmir-02
   ```

2. Remove `volmir-02` from the LSM configuration database:

   ```
   # voledit -r rm volmir-02
   ```

3. Verify that the operation was successful:

   ```
   # volprint -h volmir
   ```

```
TYPE NAME         ASSOC        KSTATE     LENGTH COMMENT
vol  volmir       fsgen        ENABLED     10240
plex volmir-01    volmir       ENABLED     10240
sd   disk01-03    volmir-01    -           10240
```

Note that `volmir-02` and its associated subdisk `disk02-02` are no longer listed. Because the `volmir` volume now has only one plex attached to it, `volmir` is no longer mirrored.

7.6.7 Removing a Volume

Once a volume is no longer necessary (it is inactive and archived, for example), you can remove the volume and free up the disk space for other uses.

To remove a volume, all references to the volume must be closed (that is, the file system in the volume must be unmounted, and the volume stopped using the `volume stop` command). The `volume stop` command stops all LSM activity to the volume.

After stopping the volume, enter the following command:

```
# voledit -r rm volume_name
```

To remove the volume `volspan`, enter:

```
# voledit -r rm volspan
```

Note

The `–r` option of the `voledit` command removes multiple objects. Exercise caution when using it.

7.7 Using LSM Volumes with UFS

The following sections summarizes how to create file systems on volumes.

7.7.1 Make and Mount a File System

A file system is created on a volume and cannot be larger than the volume on which it is created.

To create a file system on a volume, use the following syntax:

newfs [*specific_options***]** *special_file disktype*

The following table describes the qualifiers you can set with this command:

Variable	Description
specific_options	Options specific to this file system. See the `newfs`(2) reference page.
special_file	The full pathname of the volume on which to create the file system (for example, `/dev/rvol/rootdg/pubs`).
disktype	The disk whose characteristics map closely to the logical volume.

To create a file system on a volume called `/dev/rvol/rootdg/pubs`, first create the volume called `pubs` using `volassist`. Once the volume is created, enter the following command:

```
# newfs /dev/rvol/rootdg/pubs rz29b
```

After creating the file system, mount it using the following command:

```
# mount block_special mount_point
```

For example, to mount the file system `/dev/vol/rootdg/pubs`, enter:

```
# mount /dev/vol/rootdg/pubs  /pubs
```

7.7.2 Unmount a File System

If you no longer need to access the data in a file system, you can unmount it using the following command syntax:

umount *block_special | mount_point*

To unmount the `/dev/vol/rootdg/pubs`, file system, enter:

```
# unmount /dev/vol/rootdg/pubs
```

7.7.3 Display Mounted File Systems

It is important to keep track of which file systems are mounted and which are not. This saves users from trying to access unmounted file systems.

You can look at the status of file systems by entering the `mount` command. For example:

```
# mount

/dev/rz3a on / type ufs (rw)
/proc on /proc type procfs (rw)
/dev/vol/rootdg/vol-rz3g on /usr type ufs (rw)
/dev/vol/rootdg/vol01 on /ufs1 type ufs (rw)
```

7.8 Using LSM Volumes with AdvFS

You can use LSM volumes in AdvFS domains in the same manner that you use disk partitions. The LSM volume names used with AdvFS should be unique, regardless of the LSM disk group that the LSM volume is from.

7.8.1 Creating a New AdvFS Domain

Use the LSM volume's full device pathname when using an LSM volume with an AdvFS domain. For example, to create a new AdvFS domain called `lsmdom` using an existing LSM volume, `vol01`, in the `rootdg` disk group, enter:

```
# mkfdmn /dev/vol/rootdg/vol01 lsmdom
```

To create a file set called `lsmfset` with the `lsmdom` domain, enter:

```
# mkfset lsmdom lsmfset
```

To mount the AdvFS fileset `lsmfset` on `/advfs`, enter:

```
# mount -t advfs lsmdom#lsmfset /advfs
```

For further information about setting up and using domains and file sets, refer to the AdvFS documentation set.

7.8.2 Adding an LSM Volume to an Existing Domain

When configuring an LSM volume with AdvFS, use the full pathname of the LSM volume's device file. For example, to add the LSM volume `vol02` in the LSM `rootdg` disk group to the existing AdvFS domain called `lsmdom`, enter:

```
# addvol /dev/vol/rootdg/vol02 lsmdom
```

7.9 Using LSM Volumes for Secondary Swap Space

The system swap space is a vital system resource. In fact, if disk errors occur in the swap space, a system crash is likely to occur. You can guard against disk I/O errors in the secondary swap space by using an LSM mirrored volume for the secondary swap.

To do this, create an LSM volume and add it as secondary swap space using the `swapon` command and set the volume's start options to `norecov`. If you are adding multiple disks as LSM volumes to secondary swap, add the disks as several individual LSM volumes rather than striping or concatenating them into a single, larger LSM volume. Adding multiple, individual LSM volumes is preferable because the swapping algorithm automatically distributes its data across multiple disks to improve performance.

The following example demonstrates the commands you use to create and add a mirrored volume (`swapvol1` with a size of 102400 sectors) to the secondary swap space:

```
# volmake sd disk01-01 disk01,0,102400

# volmake sd disk02-01 disk02,0,102400

# volmake plex swapvol1-01 sd=disk01-01

# volmake plex swapvol1-02 sd=disk02-01

# volmake vol swapvol1 plex=swapvol1-01 usetype=gen \
start_opts=norecov

# volplex att swapvol1 swapvol1-02

# volume start swapvol1

# volprint -ht swapvol1
```

You can add an LSM volume for secondary swap space by editing the `/etc/fstab` file to add the following line:

```
/dev/vol/rootdg/swapvol1 swap2 ufs sw 0 2
```

Then, enter the following commands:

```
# swapon /dev/vol/swapvol1

# swapon -s
```

See the `swapon`(8) and the `fstab`(4) reference pages for complete information about these commands and setting up secondary swap.

7.10 Using LSM Volumes with Databases

LSM volumes have block and character device special files in the /dev/vol and /dev/rvol directories, respectively. For example, the volume dbvol in the disk group dbgrp, would have the block device file /dev/vol/dbgrp/dbvol and the character special device file /dev/rvol/dbgrp/dbvol. If the database uses the raw interface, you can specify the character special device file for an LSM volume for use with the database.

To change the owner, group, or mode of the special device files, use the LSM voledit command. Do not change these attributes using standard UNIX commands (such as chown, chgrp, or chmod).

The following example shows how to change user and group to dba and the mode to 0600 for the disk group dbgrp and volume vol_db:

```
# voledit -g dbgrp set user=dba group=dba mode=0600 vol_db
```

7.11 Unencapsulating LSM Volumes

When you unencapsulate an LSM volume, the LSM nopriv disk is removed, and the disk label's fstype field is marked unused. You should use the disklabel command to set the fstype field to the appropriate file system type. For example, if the disk partition is used for AdvFS, it should be labeled AdvFS. If it is used for UFS, it should be labeled 4.2BSD. See the disklabel(8) reference page for information on setting the fstype field.

For information on unencapsulating the /usr and /var file systems, refer to Section C.20.

For information on unencapsulating the root file system and primary swap area, refer to Section C.19.

The following sections describe how to unencapsulate LSM volumes and return to using disk partitions or AdvFS domains.

7.11.1 Unencapsulating UNIX Partitions

The following steps describe how to unencapsulate LSM volumes and return to using disk partitions.

In the example shown here, the volume named vol-rz4g, in the staffdg disk group on the rz4g partition, is being removed from LSM control. This example assumes that a UFS file system on rz4g was previously encapsulated. In addition, it assumes that the LSM configuration for volume vol-rz4g has not been changed (for example, moved to a different disk or striped).

1. Disable the vol-rz4g volume in the staffdg disk group:

```
# volume -g staffdg stop vol-rz4g
```

2. Remove the volume vol-rz4g from the staffdg disk group, recursively deleting the volume, plexes, and subdisks:

```
# voledit -g staffdg -rf rm vol-rz4g
```

3. Remove the disk media records for rz4g from LSM control:

```
# voldg rmdisk rz4g
```

4. Remove the disk access records for rz4g from LSM control:

```
# voldisk rm rz4g
```

   At this point, LSM no longer knows about the rz4g nopriv disk.

5. Edit the /etc/fstab file to mount the file system that is on the rz4g partition.

7.11.2 Unencapsulating an AdvFS Domain

The following steps describe how to unencapsulate LSM volumes and return to using AdvFS domains. In the example shown here, the domain named dom1, which includes the partition rz4e, was encapsulated into a volume named vol-rz4e in the staffdg disk group.

1. Disable the vol-rz4e volume in the staffdg disk group:

```
# volume -g staffdg stop vol-rz4e
```

2. Remove the volume from the disk group:

```
# voledit -g staffdg -rf rm vol-rz4e
```

3. Remove the disk media records and disk access records from LSM control:

```
# voldg rmdisk rz4e
# voldisk rm rz4e
```

4. Change the domain disk link:

```
# cd /etc/fdmns/dom1
# rm vol-rz4e
# ln -s /dev/rz4e rz4e
```

8 Advanced Volume Operations

This chapter describes advanced Logical Storage Manager (LSM) system administration using the LSM command line interface. Information is included about how to manipulate subdisks and plexes, as well as information on common LSM administration utilities.

8.1 Commands Summary

Table 8-1 introduces the commands and utilities discussed in this chapter.

Table 8-1: Subdisk and Plex Commands

Command	Description
`voledit`	Sets and changes various attributes for LSM configuration records that do not depend upon volume usage types. Each invocation can be applied to only one disk group at a time.
`volmake`	Creates subdisks, plex, and volume records.
`volmend`	Mends simple problems in configuration records.
`volplex`	Performs LSM operations on plexes and volume-and-plex combinations. The `volplex` utility takes several operands. The first operand is a keyword that determines the specific operation to perform. The remaining operands specify the configuration objects to which the operation is to be applied. Each operation can be applied to only one disk group at a time.
`volprint`	Displays information about volumes, plexes, and subdisks.
`volsd`	Performs LSM operations on subdisks and on plex-and-subdisk combinations. The `volsd` utility takes several operands. The first operand is a keyword that determines the specific operation to perform. The remaining operands specify the configuration objects to which the operation is to be applied. Each operation can be applied to only one disk group at a time.

8.2 Subdisk Operations

LSM volumes are composed of two types of objects, plexes and subdisks. A plex is composed of a single subdisk or a series of subdisks linked together in an address space. A subdisk is a portion of a physical disk and is defined by disk media, offset, and length. Subdisks are low-level building blocks of an LSM configuration. The following sections describe each of the operations that can be performed in relation to subdisks. These subdisk operations are:

- Creating a subdisk
- Removing a subdisk
- Displaying a subdisk
- Associating a subdisk
- Disassociating a subdisk
- Changing a subdisk
- Moving a subdisk
- Splitting a subdisk
- Joining a subdisk

8.2.1 Using volsd to Perform Subdisk Operations

The `volsd` utility is used to maintain subdisk-plex associations. The `volsd` utility can associate a subdisk with a plex or disassociate a subdisk from its associated plex, to move the contents of a subdisk to another subdisk, to split one subdisk into two subdisks that occupy the same space as the original, or to join two contiguous subdisks into one.

Note

> Some `volsd` operations can take a considerable amount of time to complete.

For detailed information, refer to the `volsd`(8) reference page.

8.2.2 Creating Subdisks

The command to create LSM objects is `volmake`. The steps to create a subdisk include specifying the following:

1. Name of the subdisk
2. Length of the subdisk
3. Starting point (offset) of the subdisk within the disk

4. Disk media name

To create a subdisk, use either of the following commands:

```
# volmake sd name disk, offset, len
```

or

```
# volmake sd name len=length offset=disk_offset disk=lsm_disk
```

For example, use either of the following command lines to create a subdisk labeled `disk02-01`, that starts at the beginning of disk `disk02`, and has a length of 8000 blocks:

```
# volmake sd disk02-01 disk02,0,8000
```

or

```
# volmake sd disk02-01 len=8000 offset=0 disk=disk02
```

Note

Commands take sizes in blocks. Adding a suffix changes the unit of measure. A `k` suffix specifies 1024-byte blocks. To preserve (encapsulate) data that exists on the disk, a plex and volume must be created to cover that data.

8.2.3 Removing Subdisks

You can remove subdisks when they are no longer in use or needed. To remove a subdisk, use the following command:

```
# voledit rm subdisk_name
```

For example, the command line to remove a subdisk labeled `disk02-01` looks like this:

```
# voledit rm disk02-01
```

8.2.4 Displaying Subdisks

The `volprint` utility displays information about LSM objects. To display general information for all subdisks, use the following command:

```
# volprint –st
```

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

To display complete information about a particular subdisk, use the following command:

```
# volprint –l subdisk_name
```

For example, the command line to obtain all database information on a subdisk labeled `disk02-01` looks like this:

```
# volprint –l disk02-01
```

8.2.5 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. In all cases, the entire area that the subdisk fills must not be occupied by any portion of another subdisk. There are several different ways that subdisks can be associated with plexes, depending on the overall state of the configuration.

If the system administrator has already created all the subdisks needed for a particular plex, subdisks are associated at plex creation by using a command similar to the following:

```
# volmake plex home-1 sd=disk02-01,disk03-00,disk03-01
```

This command creates a plex `home-1` and associates subdisks `disk02-01`, `disk03-00`, and `disk03-01` with the plex `home-1` during the plex creation process. Subdisks are associated in order starting at offset 0. Using a command like this one eliminates the need to specify the multiple commands necessary to create the plex and then associate each of the subdisks with that plex. In the example, the subdisks are associated with the plex in the order they are listed (after the `sd=`); the disk space defined as `disk02-01` will be first, the disk space of `disk03-00` is second, and `disk03-01` is third.

This method of associating subdisks is convenient during the initial configuration. Subdisks can also be associated with a plex that already exists. any subdisks that are already associated with the plex, unless the `–l` option is specified with the `volsd` command. The `–l` option provides a way to associate subdisks at a specific offset within the plex.

The `–l` option is needed in a case where a system administrator has created a sparse plex for a particular volume, and wants to make this plex complete. To make the plex complete, it is necessary to create a subdisk of exactly the size needed to fill the hole in the sparse plex, and then associate the subdisk with the plex by specifying the offset of the beginning of the hole in the plex.

Use the following command to accomplish this task:

```
# volsd -l offset assoc sparse_plex_name exact_size_subdisk
```

Note

> The subdisk must be exactly the right size because LSM does not allow for the space defined by two subdisks to overlap within a single plex.

8.2.6 Associating Logging Subdisks

Logging subdisks are one-block long subdisks that are defined for and added to a plex that is to become part of a volume using block-change logging. Block-change logging is enabled for a volume when the volume has at least two active plexes that include a logging subdisk. Logging subdisks are ignored as far as the usual plex policies are concerned, and are only used to hold the *block-change log* (described in Section 13.1.4).

Because block-change logs are one block in length, only a limited number of recent changes can be kept in the log. Older entries are removed from the log as newer I/O changes take place and are added to the log. Because all I/O must follow a log write, if no unused or completed log entries are available for use in the log block, then further I/O is delayed until some of the pending I/O completes and releases log space. This delay could adversely affect system performance.

Note

> Only one logging subdisk can be associated with a plex. Because this subdisk is frequently written, care should be taken to position it on a disk that is not heavily used. Placing a logging subdisk on a heavily-used disk can result in severe degradation of system performance.

To add a logging subdisk to a plex, use the following command:

```
# volsd aslog plex subdisk
```

For example, the command line to associate a subdisk labeled `disk02-01` with a plex labeled `vol01-02` (which is already associated with volume `vol01`) looks like this:

```
# volsd aslog vol01-02 disk02-01
```

8.2.7 Disassociating Subdisks

To break an established relationship between a subdisk and the plex to which it belongs, the subdisk is *disassociated* from the plex. A subdisk is disassociated when the subdisk is to be removed or used in another plex. To disassociate a subdisk, use the following command:

```
# volsd dis subdisk_name
```

To disassociate a subdisk labeled `disk02-01` from the plex with which it is currently associated, use the following command:

```
# volsd dis disk02-01
```

Subdisks can also be removed with the command:

```
# volsd -orm dis subdisk_name
```

8.2.8 Changing Subdisk Information

The `voledit` utility changes information related to subdisks. To change information relating to a subdisk use the following command:

```
# voledit set field=value ... subdisk_name ...
```

For example, the command line to change the comment field of a subdisk labeled `disk02-01` looks like this:

```
# voledit set comment="New comment"  disk02-01
```

The following subdisk fields can be changed using `voledit`:

- The `putil` *[n]* fields
- The `tutil` *[n]* fields
- The `len` field (only if the subdisk is disassociated)

Table 8-2 describes these fields.

Table 8-2: The putil and util Fields

Field	Description
`putil0`	This utility field is reserved for use by the LSM utilities and is retained after a reboot.
`putil1`	This utility field is reserved for use by high-level utilities such as the Visual Administrator interface (`dxlsm`) and the LSM Support Operations interface (`voldiskadm`). This field is retained after a reboot.
`putil2`	This utility field is reserved for use by the system administrator or site-specific applications. This field is retained after a reboot.
`tutil0`	This utility field is reserved for use by the LSM utilities and is cleared after a reboot.
`tutil1`	This utility field is reserved for use by high-level utilities such as `dxlsm` and `voldiskadm`. This field is cleared after a reboot.
`tutil2`	This utility field is reserved for use by the system administrator or site-specific applications. This field is cleared after a reboot.

Note

Entering data in the `putil0` field prevents the subdisk from being used as part of a plex, if it is not already being so used.

8.2.9 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 disassociated from the plex, and the new subdisk is associated with the plex, at the same offset within the plex as the source subdisk. To move a subdisk, use the following command:

```
# volsd mv old_subdisk_name new_subdisk_name
```

For the subdisk move operation to perform 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.

8.2.10 Splitting Subdisks

Splitting a subdisk divides an existing subdisk into two subdisks. The `–s` option is required to specify the size of the first of the two subdisks that will be created. To split a subdisk, use the following command:

```
# volsd –s size split sd newsd newsd2
```

In this example, *sd* is the name of the original subdisk, *newsd* is the name of the first of the two subdisks that will be created, and *newsd2* is the name of the second subdisk to be created.

If the existing subdisk is associated with a plex before the `split` operation, upon completion of the split, both of the resulting subdisks will be associated with the same plex.

8.2.11 Joining Subdisks

Joining a subdisk 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:

```
# volsd join subdisk1 subdisk2 new_subdisk
```

8.3 Plex Operations

Plexes are logical groupings of subdisks that create an area of disk space independent of any physical disk size. Replication of disk data can be accomplished by defining multiple plexes that will be attached to one volume. The replication provided by multiple plexes prevents data loss in the event of a single-point disk-subsystem failure. Multiple plexes also provide increased data integrity and reliability.

Plex operations include:

- Creating a plex
- Removing a plex
- Associating a plex
- Disassociating a plex
- Listing all plexes
- Displaying plexes
- Changing plex attributes
- Changing plex status

- Moving plexes
- Copying plexes

8.3.1 Using volplex to Perform Plex Operations

The `volplex` utility performs LSM operations on plex or on volume-and-plex combinations. The first operand is a keyword that determines the specific operation to perform. The remaining operands specify the configuration objects to which the operation is to be applied. The `volplex` utility can attach a plex to a volume and detach a plex from a volume. A detached plex does not participate in I/O activity to the volume, but remains associated with the volume. The semantics of a detached plex are somewhat different dependent upon usage type, but in the case of the standard usage types, `fsgen` and `gen`, a detached plex is reattached when a volume is next started.

Another capability provided by `volplex` is disassociation of a plex from the volume with which it is associated. When a plex is disassociated, its relationship to the volume is completely broken. At that point, the plex is available for other uses and can be associated with a different volume. This functionality is useful as part of a backup procedure.

Additionally, `volplex` provides options to copy the contents of the specified volume onto all of the named plexes, moving the contents of one plex onto a new plex, and allows for other usage-type-dependent operations to be added.

For detailed information about how to use `volplex`, refer to the `volplex`(8) reference page.

8.3.2 Creating Plexes

Plexes are created by identifying subdisks and associating them to the plex that you want to create. The `volmake` command creates LSM objects. To create a plex from existing subdisks, use the following command:

```
# volmake plex plex_name sd=subdisk_name,...
```

For example, the command line to create a plex labeled `vol01-02` using two existing subdisks labeled `disk02-01` and `disk02-02` looks like this:

```
# volmake plex vol01-02 sd=disk02-01,disk02-02
```

8.3.3 Backup Using a Plex

If a volume is mirrored, backup can be performed on that volume by taking one of the volume's plexes off line for a period of time. This eliminates the need for extra disk space for the purpose of backup only. However, it also eliminates redundancy of the volume for the duration of the time needed for the backup.

Perform a backup of a mirrored volume on an active system as follows:

1. Stop I/O activity and flush any buffers to improve the consistency of the backup. For example, Digital recommends briefly unmounting the UFS volume in order to create a complete and consistent backup.
2. Disassociate one of the volume's plexes (`vol-01`, for this example):

   ```
   # volplex dis vol-01
   ```

   This operation should only take a few seconds. It will leave the device `/dev/plex/vol-01` available as an image of the volume frozen at the time of the disassociation.
3. At this point, resume I/O activity. For example, if the volume contained UFS, it can be remounted.
4. Create a temporary volume. For example:

   ```
   # volmake -Ufsgen vol vtmp plex=vol-01
   # volume start vtmp
   ```

5. Check the temporary volume, if necessary:

   ```
   # fsck -p /dev/rvol/vtmp
   ```

6. Create a backup using the temporary volume. For example:

   ```
   # dump 0 /dev/rvol/vtmp
   ```

7. Reattach the plex to the volume in order to regain redundancy of the volume. For example:

   ```
   # volplex dis vol-01
   # volplex att volume-name vol-01
   ```

8.3.4 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:

```
# volplex att volume_name plex_name
```

For example, the command line to associate a plex labeled `vol01-02` with volume labeled `vol01` looks like this:

```
# volplex att vol01 vol01-02
```

Alternately, 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:

```
# volmake -U usetype vol vol_name plex=plex_name1, plex_name2...
```

For example, the command line to create a mirrored, `fsgen`-type volume labeled `home`, and associate two existing plexes labeled `home-1` and `home-2` to the volume `home` looks like this:

```
# volmake -Ufsgen vol home plex=home-1,home-2
```

8.3.5 Removing Plexes

When a plex is no longer needed, it can be removed. Examples of operations that require plexes to be removed are:

- Reducing the number of plexes in a volume to increase the length of another plex and its associated volume; the plex and subdisks are removed, then the resulting space can be added to other volumes
- Removing a temporary plex that was created to backup a volume and is no longer required
- Changing the layout of a plex from concatenated to striped, or vice versa

Caution

> To save the data on a plex that is to be removed, you need to know the original configuration of that plex. Several parameters from that configuration, such as stripe width and subdisk ordering, are critical to the construction of a new plex which would contain the same data. Before such a plex is removed, its configuration should be recorded.

A plex can be disassociated from a volume and removed with the following command:

```
# volplex -o rm dis plex_name
```

To disassociate and remove a plex labeled `vol01-02` use the following command:

```
# volplex -o rm dis vol01-02
```

This removes the plex `vol01-02` and all associated subdisks.

Note

> Without the `-o rm`, the `volplex` command disassociates the plex and subdisks, but does not remove them. To remove the disassociated plex and subdisks, use the following command:
>
> ```
> # voledit -r rm plex_name
> ```
>
> Because the `-r` option of `voledit` removes multiple objects. Exercise caution when using it.

8.3.6 Listing All Plexes

Listing plexes helps identify free plexes that can be used for building volumes. Using the `volprint` utility with the plex (`–p`) option lists information about all plexes; the `–t` option prints a single line of information about the plex. To list free plexes, use the following command:

```
# volprint –pt
```

8.3.7 Displaying Plexes

To display detailed information about all plexes, use the following command:

```
# volprint –lp
```

To display detailed information about a specific plex, use the following command:

```
# volprint –l plex_name
```

8.3.8 Changing Plex Attributes

The `comment` field and the `putil` and `tutil` fields are used by the utilities after plex creation. The `putil` fields attributes are maintained after a reboot; the `tutil` fields are temporary and are not retained after a reboot. Both `putil` and `tutil` have three uses and are numbered according to those uses. These fields can be modified as needed. LSM uses the utility fields marked `putil0` and `tutil0`; other products use those marked `putil1` and `tutil1`. Those marked `putil2` and `tutil2` are user fields. Table 8-2 describes the uses for the `putil` and `tutil` fields.

To change plex attributes, use the following command:

```
# voledit set field=value ... plex_name ...
```

The following steps demonstrate how you can use the `voledit` command to modify comment fields:

1. Set the `comment` field that identifies what the plex (`vol01-02`) is used for to `my_plex`.
2. Enter `u` in `tutil2` to indicate that the subdisk is in use.
3. Use the following command to change the user ID to `admin` in the `putil2` field:

```
# voledit set comment="my plex" tutil2="u" putil2="admin" \
vol01-02
```

To prevent a particular plex from being associated with a volume, set the `putil0` field to a non-null string as specified in the following command:

```
# voledit set putil0="DO-NOT-USE" vol01-02
```

8.3.9 Changing Plex Status: Detaching and Attaching Plexes

Once a volume has been created and placed on line (ENABLED), LSM provides mechanisms by which plexes can be temporarily disconnected from the volume. This is useful, for example, when the hardware on which a plex resides needs repair or when a volume has been left unstartable and a source plex for the volume revive operation must be chosen manually.

Resolving a disk or system failure includes taking a volume off line and attaching and detaching its plexes. The two commands used to accomplish disk failure resolution are `volmend` and `volplex`.

To take a plex OFFLINE so that repair or maintenance can be performed on the physical disk containing that plexes' subdisks, use the following command:

```
# volmend off plex_name ...
```

If a disk drive suffered a head crash, the system administrator should put all plexes that have associated subdisks represented on the affected drive OFFLINE. For example, if plexes `vol01-02` and `vol02p1` had subdisks on a drive to be repaired, use the following command:

```
# volmend off vol01-02 vol02p1
```

This command places `vol01-02` and `vol02-01` in the OFFLINE state, and they remain in that state until explicitly changed.

8.3.9.1 Detaching Plexes

To temporarily detach one plex in a mirrored volume, use the following command:

```
# volplex det plex_name ...
```

For example, the command line to temporarily detach a plex labeled `vol01-02` and place it in maintenance mode looks like this:

```
# volplex det vol01-02
```

This command temporarily detaches the plex, but maintains the association between the plex and its volume; however, the plex will not be used for I/O. A plex detached with the preceding command will be recovered when the system is rebooted. The plex state is set to STALE, so that if a `volume start` command is run on the appropriate volume (for example, after a system reboot), the plex will be revived and made ACTIVE.

When the plex is ready to return as an active part of its volume, follow this procedure:

1. If the volume is not ENABLED, start it as follows:

   ```
   # volume start vol_name.
   ```

   If it is unstartable, set one of the plexes to CLEAN as follows:

   ```
   # volmend plex clean plex_name,
   ```

   Then start the volume.

2. If the plex does not yet have a *kernel state* of ENABLED, issue the following command:

   ```
   # volplex att volume_name plex_name ...
   ```

   As with returning an OFFLINE plex to ACTIVE, this command starts a revive operation of the plexes stated, and when each revive operation is completed, sets the plex state to ACTIVE.

8.3.9.2 Attaching Plexes

When the disk has been repaired or replaced and is again ready for use, the plexes must be put back on line (plex state set to ACTIVE). The following steps describe putting plexes back on line.

1. If the volume is currently ENABLED, use the following command:

   ```
   # volplex att volume_name plex_name ...
   ```

   For example, the command for a plex labeled `vol01-02` on a volume labeled `vol01` looks like this:

   ```
   # volplex att vol01 vol01-02
   ```

   This starts a revive operation of the plex `vol01-02` and, after the revive operation is completed, sets the plex utility state to ACTIVE.

2. If the volume is not in use (not ENABLED), use the following command:

   ```
   # volmend on plex_name
   ```

   For example, the command for a plex labeled `vol01-02` looks like this:

   ```
   # volmend on vol01-02
   ```

   In this case, the state of `vol01-02` is set to STALE, so that when the volume is next started, the data on the plex will be revived from the other plex, and incorporated into the volume with its state set to ACTIVE.

If it becomes necessary to manually change the state of a plex, refer to Section 14.9.2. See the `volmake`(8) and `volmend`(8) reference pages for more information about these commands.

8.3.10 Moving Plexes

Moving a plex copies the data content from the original plex onto a new plex. In order for a move operation 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 should 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 important implications. If the new plex is smaller, or more sparse, than the original plex, an incomplete copy of the data on the original plex results. If this is the desired action, then the `-o force` option is required.

If the new plex is longer, or less sparse, than the original plex, then the data that exists on the original plex will be copied onto the new plex. Then, any area that was not on the original plex, but is represented on the new plex, will be filled from other complete plexes 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 will not be initialized.

The command to move data from one plex to another is:

```
# volplex mv original_plex new_plex
```

8.3.11 Copying Plexes

This operation copies the contents of a volume onto a specified plex. The volume to be copied must not be enabled. The plex must not be associated with any other volume.

To copy a plex, use the following command:

```
# volplex cp vol_name new_plex
```

After the copy operation is completed, *new_plex* will not be associated with the specified volume *vol_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, otherwise an incomplete copy of the data results. For this same reason, *new_plex* also should not be sparse.

8.4 Individual Utility Descriptions

The following sections describe LSM utilities that are commonly used to perform system administration and maintenance functions.

8.4.1 Using the voldctl Command

The volume configuration daemon (`vold`) is the interface between the LSM utilities and the kernel `volconfig` device driver. The `volconfig` device is a special device file created by LSM that interacts with `vold` to make LSM configuration changes.

The `voldctl` utility is the interface to `vold`. Some `voldctl` operations involve modifications to the `volboot` file, which indicates the locations of root configuration copies.

The `voldctl` utility is used for the following tasks:

- Performing administrative tasks related to the state of the daemon
- Managing boot information and various aspects of the LSM root configuration initialization
- Manipulating the contents of the `volboot` file, which contains a list of disks containing root configuration databases

For detailed information about how to use `voldctl`, refer to the `voldctl`(8) reference page.

8.4.2 Using voledit to Remove and Modify LSM Objects

The `voledit` utility has two functions:

1. Allows the system administrator to modify certain records in the LSM configuration databases. Only fields that are not volume usage-type-dependent can be modified.

2. Removes or renames LSM objects.

In general, LSM objects that are associated are not removable. This means that:

- A subdisk that is associated with a plex cannot be removed.
- A plex that is attached to a volume cannot be removed.

Note

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

Because the `-r` option of `voledit` removes multiple objects, exercise caution when using it.

For detailed information about how to use `voledit`, refer to the `voledit`(8) reference page.

8.4.3 Using volmake to Create LSM Objects

The `volmake` utility is used to add a new volume, plex, or subdisk to the set of objects managed by LSM. The `volmake` command adds a new record for that object to the LSM database. Records can be created entirely from parameters specified on the command line, or they can be created using a description file.

If operands are specified, then the first operand is a keyword that determines the kind of object to be created, the second operand is the name given to that object, and additional operands specify attributes for the object. If no operands are specified on the command line, then a description file is used to specify what records to create.

A *description file* is a file that contains plain text describing the objects to be created with `volmake`. A description file can contain several commands, and can be edited to perform a list of `volmake` utility operations. You can create a description file of existing LSM objects with the `volprint` utility, and then later recreate these LSM objects using the `volmake` utility. The following example shows a sample description file:

```
#rectyp   #name#options
sd   3s1-01    dmname=disk03 offset=0 len=20480
sd   4s1-01    dmname=disk04
plex db-dsk    layout=STRIPE st_width=16k sd=3s1-01,4s1-01
sd   mem1-01   dmname=memdisk01 len=640h
          comment="Hot spot for dbvol"
plex db-mem    sd=mem1-01:40320

vol  db   use_type=gen plex=db-dsk,db-mem
          read_pol=PREFER pref_name=memdbplx
          comment="Uses mem1 for hot spot in last 5m"
```

By default, this description file is read from standard input. However, by using the `-d` option, a file name can be specified. For detailed information about how to use `volmake`, as well as detailed descriptions and definitions of the object-specific fields specified with `volmake`, refer to the `volmake`(8) reference page.

8.4.4 Using the volmend Utility

The `volmend` utility performs miscellaneous LSM usage-type-specific operations on volumes, plexes, and subdisks. These operations are used to:

- Clear utility fields
- Change the state of a volume or plex
- Take a volume or plex off line
- Place a volume or plex on line
- Perform specialized actions for objects with a particular usage type

Use the `volmend` utility primarily to escape from a state that was accidentally reached. The offline and online functions are also available with disk-related commands.

For detailed information about how to use `volmend`, refer to the `volmend`(8) reference page.

8.4.5 Defaults File for volassist

The `volassist` command provides a convenient, one-step interface to LSM and is especially useful for basic and commonly used administrative operations. The `volassist` command automatically finds space for and creates simple volumes or plexes for existing volumes, resizes volumes, and provides online backup of volumes.

The `volassist` command uses a set of tunable parameters that you can specify in defaults files or on the command line. The tunable parameters default to reasonable values if they are not defined anywhere. The tunable parameters are specified as follows:

- Internal defaults

  The built-in defaults are used when the value for a particular parameter 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 may be altered by the system administrator. These values are used for tunable parameters that are not specified on the command line or in the user's defaults file.

- User defaults file

  The user can create a personal defaults file. If a personal defaults file exists, the values therein are used for tunable parameters that are not specified on the command line. These values override those in the system-wide defaults file.

- Command line

  The values specified on the command line override any values specified internally or in defaults files.

Tunable parameters are specified in the `/etc/defaults/volassist` file, which you must create. 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 command line.

The following is a sample `volassist` defaults file:

```
# LSM Vn.n
# volassist defaults file.  Use '#' for comments

# layout
layout=concat,noncontig,span

# mirroring
nmirror=2
mirror=no

# allocation policies
align=4k
alloc=20m

# striping
stripewidth=64k

# logging
logtype=none

# volume usage type
```

```
usetype=fsgen
```

For detailed information about how to use `volassist`, refer to the `volassist`(8) reference page.

Part 3: Menu Interface

Menu Interface (voldiskadm) 9

This chapter describes the LSM Support Operations designed to help you perform LSM disk and disk group operations in the menu interface. You enable LSM Support Operations using the `voldiskadm` command, which starts up a menu-driven interface. The menus are easy to use and provide information about each step to help you decide the correct response for each prompt.

The following sections describe how to start up and use the LSM Support Operations menu interface for some of the most common disk management tasks.

9.1 Starting LSM Support Operations

To start LSM Support Operations, enter the following command:

```
# voldiskadm
```

LSM brings up a Support Operations main menu shown in Figure 9-1 that displays the disk operation options available to you.

Figure 9-1: LSM Support Operations Main Menu

```
Logical Storage Manager Support Operations
Menu LogicalStorageManager/Disk

 1     Add or initialize a disk
 2     Encapsulate a disk
 3     Remove a disk
 4     Remove a disk for replacement
 5     Replace a failed or removed disk
 6     Mirror volumes
 7     Move volumes from a disk
 8     Enable access to (import) a disk group
 9     Remove access to (deport) a disk group
10     Enable (online) a disk device
11     Disable (offline) a disk device
list   List disk information
?      Display help about menu
q      Exit from menus

Select an operation to perform:
```

Table 9-1 describes the main menu selections.

Table 9-1: Selections on the Main Menu

Option	Description
1	Identifies a disk to LSM and prepares the disk for LSM use. Use this option to initialize disks that are new or that do not contain valid data.
2	Prepares a disk for LSM usage while preserving existing data on the disk.
3	Removes a disk from LSM. This option does not retain the disk name for future disk replacement.
4	Removes a failed disk, retaining the disk name, and replaces it with another disk.
5	Specifies a replacement disk for a disk that either failed or was removed using menu option 4.
6	Duplicates data to give you one or more copies on another disk.
7	Moves data on a disk to another disk or disks on the system.

Table 9-1: (continued)

Option	Description
8	Enables access by this system to a disk group.
9	Disables access to a disk group that is currently enabled (imported) by this system.
10	Places the disk access record in an online state.
11	Places the disk access record in an offline state.
list	Lists disks that are available to the system.
?	Provides help when using the LSM menus. The output of the ? command is a list of operations and a definition of each.
q	Enter the q command: • From any menu to return to the main menu. • From the main menu to exit the menu interface.

9.2 Disk Operations

This section describes the disk operations available with the LSM Support Operations.

9.2.1 Initializing a Disk

Disk initialization identifies a disk to LSM and prepares the disk for LSM use. This operation involves installing a disk header and writing an empty configuration on the disk. A disk access record is created for the disk, unless such a record already exists.

To initialize a disk for use with LSM, perform the following steps:

1. At the main menu prompt (shown in Figure 9-1), select menu item 1 to enable the ‘‘Add or initialize a disk’’ operation.
2. At the prompt on the following ‘‘Add or initialize a disk’’ screen, enter the address of the disk to be added.

```
Add or initialize a disk
Menu:  LogicalStorageManager/Disk/AddDisk

Help information about this screen
          :

Select disk device to add [<disk/partition name>, list, q, ?]
```

If you do not know the address of the disk you want to add, enter the letter `l` or type the word `list` at the prompt. LSM displays a list of the disks that are available as replacement disks. For example:

```
DEVICE     DISK       GROUP      STATUS

rz1        disk00     rootdg     online
rz2        -          -          online
rz3        -          -          error
```

3. Once you have entered the disk name, LSM displays this screen that asks you to supply the name of the disk group you want the disk to be a part of:

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

Note that you can press the Return key to accept the default disk group name, `rootdg`. You can also enter the word none if either of the following conditions are true:

– The disk group you want the disk to be a part of does not exist yet.

– You want to keep this disk available as a spare to be used as a replacement disk.

4. Depending on your response to the ‘‘Which disk group...’’ prompt, LSM displays one of the following screens.

– If you entered none, LSM displays the following screen:

```
The requested operation is to initialize disk rz3 and
to leave this disk free for use as a spare disk.


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

– If you selected `rootdg` as the disk group, LSM displays the following screen:

```
The requested operation is to initialize disk device rz3 and
to add this device to disk group rootdg as disk disk01.


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

Press the Return key to continue.

5. If LSM successfully completes the disk initialization, the following screen appears:

```
Disk initialization for rz3 completed successfully.


Add or initialize another disk?  [y,n,q,?]  (default: n)
```

Press the Return key to return to the main menu.

If the `fstype` in the disk label of the specified partition or an overlapping partition is set, LSM displays a warning message to inform you that initializing the disk might destroy existing data.

If you are sure that the disk partition has no valid data and that the partition can be added to LSM, you can ignore the warning message and answer `y` to the prompt. The `voldiskadm` utility then proceeds to initialize the disk partition and add it to LSM.

If the disk cannot be initialized because the specified partition or an overlapping partition on the disk is open (that is, a partition is actively in use by UFS, AdvFS, LSM or swap), the initialization process fails and `voldiskadm` exits with an error message informing you of the problem.

9.2.2 Displaying Disk Information

The following steps describe how to find information about disks available on the system.

1. From the main menu (see Figure 9-1), enter the letter `l` or type the word `list` to display a list of disks available on the system.

   LSM displays a list of devices similar to the following screen, and prompts you to enter the address of the disk for which you want to obtain detailed information.

```
DEVICE      DISK        GROUP       STATUS

rz3         disk00      rootdg      online
rz4         –           –           online
rz5         –           –           online

Device to list in detail [<disk/partition name>,none,q,?]  (default: none)
```

2. The following screen displays information for the disk device `rz3`:

```
Device:      rz3
devicetag:   rz3
type:        sliced
hostid:
disk:        name= id=723605502, 1095.mysys
group:       name= id=
flags:       online  ready  private  autoconfig
pubpaths:    block = /dev/rz3g  char = /dev/rrz3g
privpaths:   block = /dev/rz3h  char = /dev/rrz3h
version:     1.1
iosize:      512
public:      slice = 13  offset = 0  len = 660992
private:     slice = 14  offset = 0  len = 512
update:      time = 723605503  seqno = 0.1
header:      0  248
configs:     count = 2  len = 173
logs:        count = 2  len = 26
Defined regions:
   Config  priv  000017-000189[000173]: copy = 01  offset = 000000

   log  priv  000190-000215[000026]: copy = 01  offset = 000000
   log  priv  000296-000321[000026]: copy = 02  offset = 000000
   config  priv  000322-000494[000173]: copy = 02  offset = 000000

List another disk device?  [y,n,q,?]  (default: n)
```

3. Press the Return key to return to the main menu.

9.2.3 Adding a Disk to a Disk Group

You may want to add a new disk to an already established disk group. Perhaps the current disks 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 performing these steps:

1. Follow the instructions documented in Section 9.2.1.
2. When the add disk operation adds a disk to a disk group, LSM checks to see if the disk is already initialized. If the disk has been initialized, LSM displays the following screen and asks whether or not you want to reinitialize the disk:

```
Disk device rz3 appears to have been initialized already.
The disk is currently available as a replacement disk.


Do you wish to reinitialize?  [y,n,q,?]  (default: y)  n
```

Use the information in the following table to determine whether or not you should reinitialize the disk.

If...	Then...
The disk is new	Initialize the disk before placing it under the control of LSM.
The disk was previously in use and contains useful data	Do not initialize the disk. Instead, use the LSM encapsulation function to add the disk to the LSM system while still preserving the existing data.
The disk was previously in use but it does not contain useful data	Initialize the disk before placing the disk under LSM control.

9.2.4 Moving Volumes from a Disk

Before you disable or remove a disk, you may want to move the data from that disk to other disks on the system. Use this operation immediately prior to removing a disk, either permanently or for replacement (described in Section 9.2.5).

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, two consecutive move operations could move volumes from one disk to another.
>
> Also note that you need to make sure the other disks in the disk group have sufficient space available.

To move volumes from a disk, do the following:

1. Select menu item 7 from the main menu (shown in Figure 9-1).

2. From the ‘‘Move volumes from a disk’’ screen, enter the name of the disk whose volumes you want to move:

```
Move volumes from a disk
Menu:  LogicalStorageManager/Disk/Evacuate

Help information about this screen
          :

Enter disk name  [<disk>, list, q,?]     disk01
```

3. Verify the information that LSM displays on the following screen:

```
  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)
```

   Press Return to move the volumes.

4. As LSM moves the volumes from the disk, it displays the status of the operation:

```
  Move volume voltest  ...

  Move volume voltest-bk00  ...
```

5. When the volumes have all been moved, LSM displays the following success screen:

```
Evacuation of disk disk01 is complete.

Move volumes from another disk?  [y,n,q,?]  (default: n)
```

Press Return to return to the main menu.

9.2.5 Removing a Disk from a Disk Group

This operation involves removing the LSM disk associated with the selected partitions from LSM control by removing the associated disk access records. LSM Support Operations provides two methods (menu items 3 and 4 on the main menu shown in Figure 9-1) for removing disks. These two operations remove a disk as follows:

- Menu item 3, Remove a disk — Removes a disk completely from LSM control and does not retain the disk name.
- Menu item 4, Remove a disk for replacement — Removes a failed disk and retains the disk name so it can be replaced with another disk.

See also Section 9.2.4 which describes how to move data from a disk to another disk on the system, and see Section 9.2.6 which describes how to replace a failed or removed disk.

Note

You must disable the disk group before you can remove the last disk in that group. Disabling a disk group, also referred to as deporting a disk group, is described in Section 9.3.2.

9.2.5.1 Removing a Disk Without Replacement

To remove a disk from its disk group, perform the following steps:

1. Select menu item 3 from the main menu (shown in Figure 9-1).
2. LSM displays the following ‘‘Remove a disk’’ screen and prompts you to enter the disk name of the disk to be removed.

```
Remove a disk
Menu:  LogicalStorageManager/Disk/RemoveDisk

Help information about this screen
        .
        .
        .

Enter disk name [<disk>, list, q, ?]  disk02
```

This example removes the disk device `disk02`.

3. LSM displays a verification screen and asks whether or not to continue:

```
Requested operation is to remove disk disk02 from group newdg.

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

Press the Return key to continue.

4. LSM removes the disk from the disk group and then displays the following screen when the operation has completed:

```
Removal of disk disk02 is complete.

Remove another disk?  [y,n,q,?]  (default: n)
```

Press the Return key to return to the main menu.

9.2.5.2 Removing a Disk for Replacement

You may occasionally need to replace a disk in a disk group. This operation involves initializing the disk for LSM use, and replacing the old disk and associated disk media records with the new disk and its information. Perform the following steps to replace a disk while retaining the disk name:

1. Select menu item 4 from the main menu (shown in Figure 9-1). LSM displays the ‘‘Remove a disk for replacement’’ screen.

```
Remove a disk for replacement
Menu:  LogicalStorageManager/Disk/RemoveForReplace

Help information about this screen
        .
        .
        .
Enter the disk name [<disk>, list, q, ?]  1
```

2. Enter the name of the disk to be replaced if you know it. Otherwise, enter the letter `l` for a list of disks. LSM displays a screen similar to the following:

```
DM  NAME      DEVICE    TYPE     PRIVLEN   PUBLEN    PUBPATH
dm  disk00    rz2       sliced   512       143632    /dev/rrz2g
dm  disk01    rz4       sliced   512       660992    /dev/rrz4g

Enter disk name  [<disk>,list,q,?]  disk01
```

3. If there are any initialized disks available that are not part of a disk group, LSM displays the following screen and gives you the option of using one of these disks as a replacement. Select the replacement disk from the list provided.

```
The following devices are available as replacements:

rz3

You can choose one of these disks now, to replace disk01.
Select "none" if you do not wish to select a replacement disk.

Choose a device, or select "none"
[<device>,none,q,?] (default: rz3)
```

Press the Return key if you want to use the default disk.

4. LSM then displays the following verification screen:

```
Requested operation is to remove disk disk01 from group rootdg.
The removed disk will be replaced with disk device rz3.

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

Press the Return key to continue.

5. When LSM successfully replaces the disk, LSM displays the following screen:

```
Removal of disk disk01 completed successfully.
Proceeeding to replace disk01 with device rz3.
Disk replacement completed successfully.
Remove another disk?   [y,n,q,?]  (default: n)
```

Press the Return key to return to the main menu.

9.2.6 Replacing a Failed or Removed Disk

Use this menu operation to specify a replacement disk for a disk that you removed with the ‘‘Remove a disk for replacement’’ menu operation (see Section 9.2.5.2), or for a disk that failed during use. To replace a disk, use the following instructions:

1. Select menu item 5 from the main menu (shown in Figure 9-1).
2. The following ‘‘Replace a failed or removed disk’’ screen asks you to enter the name of the disk to be replaced. You can choose an uninitialized disk, or you can choose a disk that you have already initialized (skip to the next step).

```
Replace a failed or removed disk
Menu:  LogicalStorageManager/Disk/ReplaceDisk

Help information about this screen
          .
          .
          .

Select a removed or failed disk  [<disk>, list, q,?]  disk01
```

3. If there are any initialized disks available that are not part of a disk group, LSM displays the following screen to list the disks available to be used as a replacement. Select the replacement disk from the list provided.

```
The following devices are available as replacements:

rz3

You can choose one of these disks now, to replace disk01.
Select "none" if you do not wish to select a replacement disk.

Choose a device, or select "none"
[<device>,none,q,?] (default: rz3)
```

   Press Return to select the default device or enter the device name of the device of your choice.

4. LSM displays a confirmation screen:

```
The requested operation is to use the initialized device rz3
to replace the removed or failed disk disk01
in disk group rootdg.

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

   Press Return to replace the disk.

5. LSM displays the following success screen:

```
Replacement of disk disk01 in group rootdg with disk
device rz3 completed successfully.

Replace another disk?  [y,n,q,?]  (default: n)
```

Press Return to return to the main menu.

9.2.7 Renaming a Disk

To rename a disk, perform the following steps:

1. Perform the steps to remove the disk as described in Section 9.2.5.
2. Select menu item 1 from the main menu (shown in Figure 9-1) to bring up the ‘‘Add or initialize a disk’’ menu.
3. Select the address of the disk you just removed. This is the disk that you will add with a new name:

```
Add or initialize a disk
Menu:  LogicalStorageManager/Disk/AddDisk

Help information about this screen
            :
Select disk device to add [<disk/partition name>, list, q, ?] rz3
```

4. LSM displays a screen that tells you the device is already initialized, and is available as a replacement disk, and asks whether or not you want to reinitialize the disk. Press the Return key to reinitialize the disk. Otherwise, enter `n` to avoid reinitialization.
5. Enter the name of the disk group in which you want to include the disk. In the following example, assume that the disk group, `newdg`, has already been created.

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

6. Enter the new disk name. The following example uses `disk03`:

```
Enter disk name:  [<diskNN>,*,q,?]  (default: disk02)  disk03
```

7. LSM then displays the following verification screen:

```
The requested operation is to add disk device rz3 to disk
group newdg as disk disk03.

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

8. Press the Return key if the information is correct and you want LSM to continue.

9. Look for a message screen similar to the following that displays when LSM finishes the renaming operation:

```
Disk initialization for rz3 completed successfully.

Add or initialize another disk?  [y,n,q,?]  (default: n)
```

   Press the Return key to return to the main menu.

9.2.8 Disabling a Disk

This operation places the disk access record in an offline state. During searches for disk IDs or members of a disk group, offline disks are ignored. To disable a disk, perform the following steps:

1. Select menu item 11 from the main menu (shown in Figure 9-1) to disable (offline) a removable disk.
2. On the next screen, select the disk you want to disable:

```
Disable (offline) a disk device
Menu:  LogicalStorageManager/Disk/OfflineDisk

 Help information about this screen
          :

Select a disk device to disable [<disk/partition name>, list, q, ?]  rz3
```

   This example shows that disk `rz3` has been selected.
3. LSM disables disk `rz3` and then asks if you want to disable another device:

```
    Disable another device?  [y,n,q,?]  (default: n)
```

   Press the Return key to return to the main menu.

9.3 Disk Group Operations

This section describes the disk group operations that can be performed with the LSM Support Operations.

9.3.1 Importing (Enabling) a Disk Group

Use this menu operation to enable access by this system to a disk group. This operation can be used to move a disk group from one system to another. If you want to move a disk group from one system to another you must first disable (deport) it on the original system (see Section 9.3.2), then move the disk between systems and enable (import) the disk group.

Note

If two hosts share a SCSI bus, make sure that the other host really failed or deported the disk group. If two hosts import a disk group at the same time, the disk group will be corrupted and become unusable.

To import a disk group, do the following:

1. Select menu item 8 from the main menu (shown in Figure 9-1). From the ‘‘Enable access to (import) a disk group’’ menu, select the name of the disk group to import:

```
Enable access to (import) a disk group
Menu:  LogicalStorageManager/Disk/EnableDiskGroup

Help information about this screen
        :

Select disk group to import [<group>, list, q,?] (default: list)  newdg
```

2. Once the import is complete, LSM displays the following success screen:

```
  The import of newdg was successful.

Select another disk group?  [y,n,q,?]  (default: n)
```

Press Return to return to the main menu

9.3.2 Deporting (Disabling) a Disk Group

Use this operation 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 some new purpose.

Note

For removable disk devices on some systems, it is important to disable all access to the disk before removing the disk.

To deport a disk group, perform the following steps:

1. Select menu item 9 from the main menu (shown in Figure 9-1).
2. From the following ‘‘Remove access to (deport) a disk group’’ menu, enter the name of the disk group to be deported:

```
Remove access to (deport) a disk group
Menu:  LogicalStorageManager/Disk/CreateDiskGroup

Help information about this screen
        :

Enter name of disk group  [<group>, list, q,?] (default: list)  newdg
```

3. Verify that you want LSM to disable the disk group on the displayed confirmation screen:

```
  The requested operation is to disable access to the
  removable disk group named newdg.  This group is stored
  on the following disks:

     newdg on device rz3

  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)
```

   Press Return to deport the disk group.

4. On the next screen display, verify that you want to continue with the operation:

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

   Press Return to continue.

5. Look for the following screen that displays once the disk group is deported:

```
Removal of disk group newdg was successful.

Disable another disk group?  [y,n,q,?]  (default: n)
```

Press Return to go to the main menu.

9.4 Volume Operations: Mirroring Volumes on a Disk

Mirroring the volumes on a disk gives you one or more copies of your volumes in another disk location. By creating mirror copies of your volumes, you protect yourself against loss of data in case of a disk failure.

LSM's `voldiskadm` utility cannot be used to mirror volumes that are already mirrored, or that are comprised of more than one subdisk.

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:

1. Select menu item 6 from the main menu (shown in Figure 9-1).
2. On the ''Mirror volumes on a disk'' menu, enter the name of the disk whose volumes you want to mirror:

```
Mirror volumes on a disk
Menu:  LogicalStorageManager/Disk/Mirror

Help information about this screen
        :

Enter disk name  [<disk>, list, q,?]   disk02
```

3. Select the target disk name (this disk must be the same size or larger than the originating disk). Volumes can be mirrored onto another disk or onto any available disk space.

```
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
```

4. LSM displays the verification screen:

```
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)
```

   Press Return to make the mirror.

5. LSM displays the status of the operation as it performs the mirroring:

```
  Mirror volume voltest-bk00 ...
  Mirroring of disk disk01 is complete.

Mirror volumes on another disk?  [y,n,q,?]  (default: n)
```

6. Once LSM has completed the mirroring operation, it asks if you want to mirror volumes on another disk. Press Return to go back to the main menu.

9.5 Exiting LSM Support Operations

When you have completed all of your disk administration activities, exit the LSM Support Operations by selecting menu option `q` from the main menu.

Part 4: Graphical Interface

Introduction to the Visual Administrator (dxlsm) 10

The Visual Administrator, also called `dxlsm`, is a graphical user interface for LSM. This interface is designed primarily for disk and volume operations, but also provides a limited set of file-system operations. You should understand the principles of LSM operations before you use the Visual Administrator. See Chapter 1 for an overview of LSM.

This chapter gives an overview of Visual Administrator features. See Chapter 12 for information on using the Visual Administrator for LSM operations.

10.1 Mouse Buttons

A two- or three-button mouse is required to use the Visual Administrator. Table 10-1 describes the mouse buttons, referred to throughout this chapter as MB1, MB2, and MB3.

Table 10-1: Default Mouse Buttons

Virtual Mouse Button	3-Button Access	2-Button Access	Function
MB1	Left	Left	Selects a single icon.
MB2	Middle	Ctrl -Left	Selects either one or multiple icons simultaneously.
MB3	Right	Right	Displays either the properties form or the analysis statistics form for that object, depending on whether the icon is undergoing analysis.
Shift -MB1	Shift -Left	Shift -Left	Toggles between minimizing or maximizing an icon.
Shift -MB2	Shift -Middle	Ctrl -Right	Toggles between starting or stopping projection on the selected icon.

Table 10-1: (continued)

Virtual Mouse Button	3-Button Access	2-Button Access	Function
Shift -MB3	Shift -Right	Shift -Right	Displays the properties form for the object, regardless of whether analysis is in effect.

An icon can be deselected by positioning the pointer over that icon and clicking MB2. This works regardless of which mouse button was used to select the icon.

Note

> The examples in this document assume that you are using a three-button mouse, set up according to Table 10-1. It is possible to redefine mouse buttons (using the `xmodmap` command, for instance). Refer to your X Window System documentation for details.

10.2 Icons

The Visual Administrator interface uses icons to represent LSM volumes, plexes, subdisks, and disks.

Disk groups are represented as view windows rather than icons. The icons representing LSM disks, volumes, and other objects belonging to a particular disk group are all displayed within the view of the disk group.

10.2.1 Icon Characteristics

Figure 10-1 illustrates the icons that `dxlsm` uses to represent LSM objects. Table 10-2 describes the icons and their characteristics

Figure 10-1: Icons That Represent LSM Objects

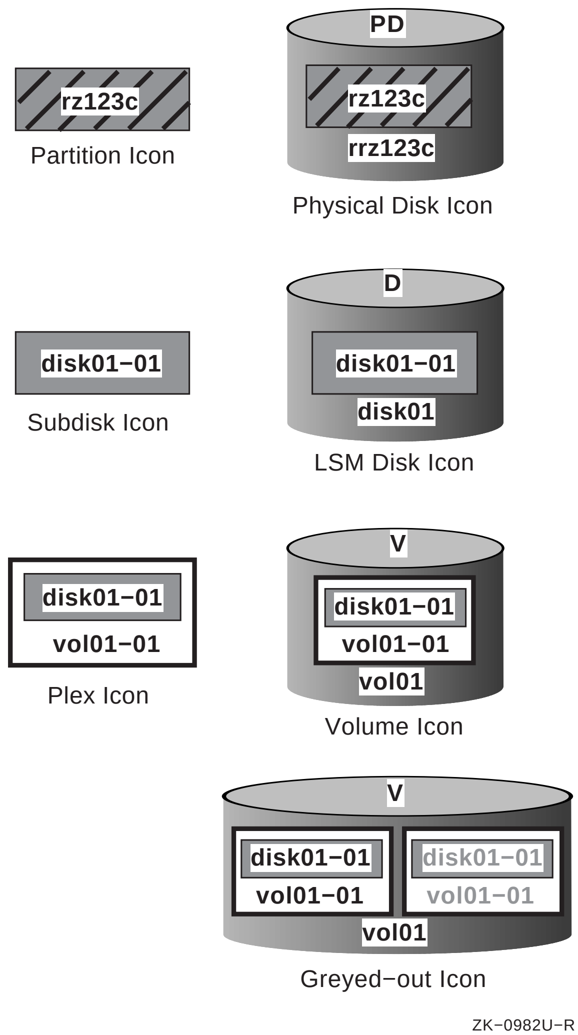


Table 10-2: Visual Administrator Icon Characteristics

Icon	Description
Physical Disk	Physical disks appear as cylindrical icons labeled `PD`. These icons represent physical disks known to dxlsm. Physical disk icons appear in the View of Disks window.
Partition	Partitions appear as rectangular icons within physical disk icons. The partition icon is labeled with the device name. If a disk has been added to a disk group, the corresponding partition icon is shaded. Partition icons appear in the View of Disks window.
LSM Disk	LSM disks appear as cylindrical icons labeled `D`. They usually contain subdisks, which are represented as rectangles. LSM disk icons represent disks that are both under LSM control and assigned to a disk group. LSM disk icons are labeled with the disk name, by default. LSM disk icons typically appear in a disk group view.
Subdisk	Subdisks appear within LSM disks (and often within plexes) as rectangular shaped icons. Subdisk icons typically appear in disk group views or in the View of Volumes window. Log subdisks (used to log recent disk activity) have icons with double borders to distinguish them from regular subdisk icons.
Plex	Plexes appear either alone or within volumes as relatively large rectangles containing subdisks. Plex icons have a heavy border to distinguish them from partition or subdisk icons. Plex icons typically appear in disk group views or in the View of Volumes window.
Volume	Volumes appear as cylindrical icons labeled `V`. These icons often contain plex and subdisk icons. Volume icons are distinguished from disk icons by a heavy border. Volume icons typically appear in disk group views or in the View of Volumes window.
Greyed out	With some operations, icons are updated almost instantly to reflect the results of the operation just performed. During other operations, it may take awhile for a particular icon to update itself. While being updated, icons are prevented from accepting input or undergoing configuration changes. Since an icon that is busy being updated should not be selected or manipulated, dxlsm greys out the text in that icon so that the user is aware that it is temporarily inaccessible. No input is accepted by an icon while it is greyed out. As soon as the icon is fully updated, it returns to its normal visual state and accepts input again. Icons that are temporarily greyed out in this manner are also referred to as *blocked icons.*

10.2.2 Manipulating Icons

There are two ways to manipulate icons:

Style	Description
Select-Operate	An icon representing an LSM object is selected and then the desired operation is performed on that object via menus and forms.
Drag and Drop	An object is manipulated by dragging its icon and then dropping it elsewhere (such as on another object in any view window).

10.2.2.1 Select-Operate Operation

The select-operate operation works as follows:

1. Select an icon representing a LSM object by positioning the pointer over the icon and clicking the MB1 button (when selecting a single icon) or the MB2 button (when selecting multiple icons).
2. From the Basic-Ops and Advanced-Ops pull-down menus, choose and activate the operation to be performed on the selected icons. Some operations require that a form be completed before the operation is activated.

10.2.2.2 Drag and Drop Operations

The drag and drop operation works as follows:

1. Drag an icon by placing the pointer on it, pressing and continuing to hold down the MB1 mouse button, and moving the mouse until the outline of the icon is superimposed on its destination.
2. Drop an icon by releasing the MB1 mouse button when the icon reaches its drop location.

When an icon is dragged and dropped, the resulting operation depends on the icon type and drop location. The drop location can be another icon or a different location.

Note

When dropping an icon onto another icon, the dragged icon must be positioned so that the pointer (in the image of a hand) is directly over an unobscured portion of the icon on which it is to be dropped.

Some icons can be copied from one user-created view window to another. Only one copy of a particular icon may appear in a view, however. Attempts to drop a second copy are ignored.

Table 10-3 summarizes the possible drag and drop operations.

Table 10-3: Drag and Drop Operations

Icon Type	Drop Location	Action
Free subdisk	View window	Create a plex and associate the subdisk to the plex.
Free subdisk	LSM disk	Create an identically-sized subdisk on the disk.
Free subdisk	Plex	Associate the subdisk to the plex.
Associated subdisk	Free subdisk	Swap the associated subdisk with the free subdisk. The free subdisk becomes associated and replaces the original subdisk, which is removed.
Associated subdisk	LSM disk	Create an identical free subdisk on the LSM disk, then swap the associated subdisk with the new free subdisk. The free subdisk becomes associated and replaces the original subdisk, which is removed.
Associated subdisk	View window	Dissociate the subdisk.
Associated plex	View window	Dissociate the plex.
Dissociated plex	User's view window	Copy the plex icon to the user's view.
Plex	Volume	Associate the plex to the volume.
Disk	User's view window	Copy the physical or LSM disk icon to the user's view.
Volume	User's view window	Copy the volume icon to the user's view.
Partition	Disk group view	Add a LSM disk (corresponding to the partition) to that disk group.

Note

If you perform a drag and drop operation that is not recognized by the Visual Administrator, that drag and drop operation is either ignored or a warning message box appears. In some cases, the Command Info Window displays an error relating to an unacceptable drag and drop attempt.

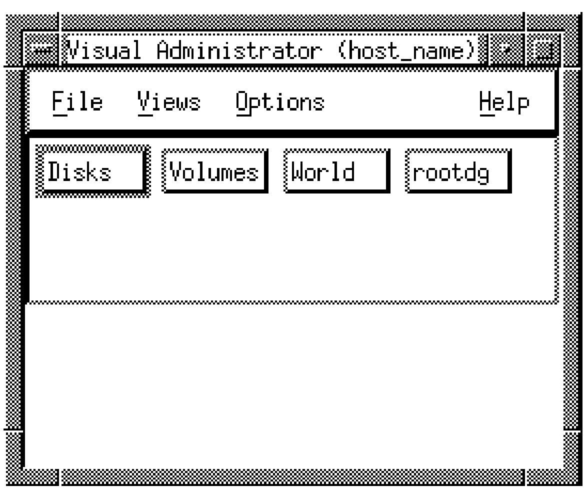
10.3 Windows and Views

You can use `dxlsm` to graphically display various aspects of LSM objects. For example, you can display all the disks under LSM control. You can also create a display for a particular disk group. This section describes the windows accessible through `dxlsm`.

10.3.1 Root Window

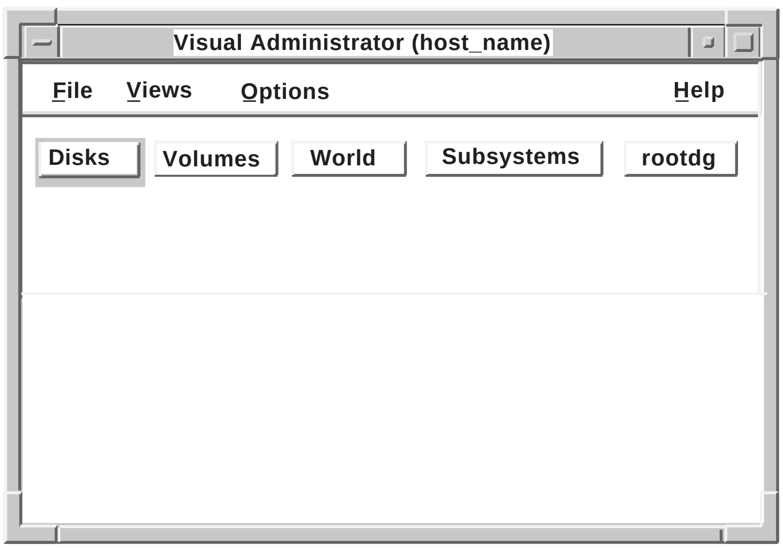
When the Visual Administrator comes up, it displays the main Visual Administrator window, also known as the *root window*. The root window contains a menu bar and a set of buttons. The set of buttons varies slightly depending on whether you have RAID (Redundant Arrays of Independent Disks) subsystems installed on your system. If you do not have RAID subsystems installed, the root window displays as shown in Figure 10-2.

Figure 10-2: Visual Administrator Root Window



If you have RAID subsystems installed, the root window displays as shown in Figure 10-3.

Figure 10-3: Visual Administrator Root Window for RAID Access



The menu bar contains the following pull-down menu items. See Chapter 11 for more information on the operations you can perform using these menus.

Item	Description
File	Closes the current window or exits the Visual Administrator interface completely
Views	Creates and manipulates user view windows
Options	Sets user preferences for dxlsm and displays the command window
Help	Accesses the help facility

Immediately after the root window displays, the ‘‘View of rootdg’’ window also appears automatically. This view displays icons representing everything that currently exists in the `rootdg` disk group.

10.3.2 Views

From the Visual Administrator root window, you can display the *view windows*, also referred to as *views.* View windows allow you to examine and manipulate different parts of the physical and logical storage systems. Each view window title includes the name of the machine on which the session is running.

The Visual Administrator root window provides a view button area containing a button for every view on the system. The buttons for the Disks, Volumes, World, and rootdg views windows are shown in Figure 10-2. To access a view window, click MB1 on one of the view buttons.

The Visual Administrator allows for two types of views: default and user-created. Both types function identically, but certain restrictions are placed on default views. Default views cannot be removed or renamed, as user-created views can.

10.3.2.1 Default Views

Click MB1 on the view buttons to access the following default view windows:

Menu Buttons	Window	Access
Disks	View of Disks	Displays all physical disks on the system
Volumes	View of Volumes	Displays all volumes, as well as plexes and associated subdisks, on the system
World	View of World	Displays everything on the system including physical and LSM disks, volumes, and other objects
rootdg	View of rootdg	Displays everything in the default disk group, `rootdg`, including LSM disks, volumes, and other objects

10.3.2.2 User-created Views

A user-created view is a view window that focuses on a particular part of a physical and a logical mass storage system, as defined by the system administrator. For example, a system administrator can create a view window for each disk group.

User-created views differ from default views in that they contain copies of icons from default views. Operations performed on these icon copies are reflected in the default views that display the affected icons. However, icons that appear in user-created views are not always updated whenever those icons are altered in the corresponding default view.

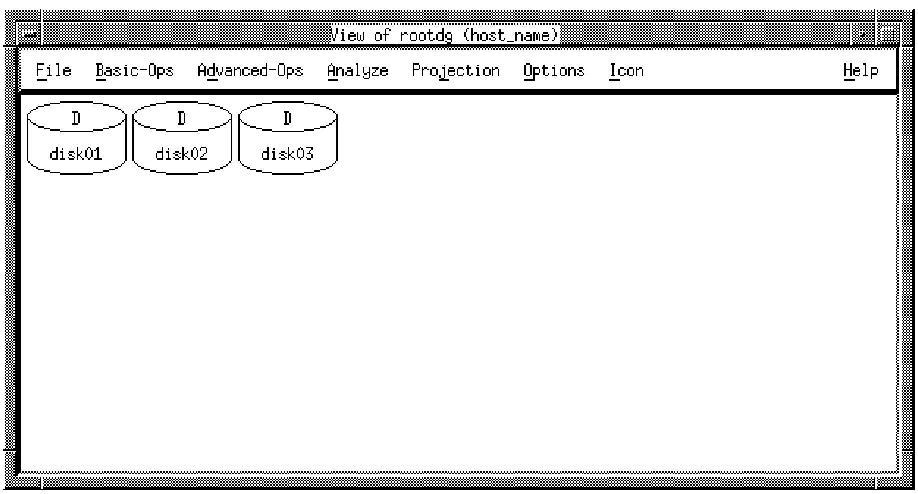
Use the Views menu from the Visual Administrator root window to create a new view window. Once the view is created, you can use its Icon menu to add icons, by copying them over from existing views.

10.3.2.3 View of rootdg

The ‘‘View of rootdg’’ window contains objects belonging to the `rootdg` disk group. By default, it appears immediately after the Visual Administrator window displays (see Figure 10-4). Whenever possible, perform operations in the ‘‘View of rootdg’’ window, or in another disk group view.

Figure 10-4 shows a ‘‘View of rootdg’’ window that does not yet contain any volumes. See Chapter 11 for information on the operations that can be performed from the ‘‘View of rootdg’’ and other disk group windows.

Figure 10-4: View of rootdg



10.4 Pull-Down Menus

The Visual Administrator provides pull-down menus that provide access to various Visual Administrator features. Once you have accessed a pull-down menu you can select any available option. Unavailable options are displayed within the menu in greyed-out text to distinguish them from available ones. Some menu selections lead to other submenus; some lead to forms that can be completed and then applied; some simply execute a specific operation immediately. Help is available on every menu to assist you in understanding the current operation.

Each window contains a title bar that identifies the window. Below the title bar is a menu bar, which contains the names of each available pull-down menu. One of the characters of each menu name is underlined; this is the mnemonic associated with that particular menu.

Figure 10-5 illustrates both the title bar and menu bar of a typical view window.

Figure 10-5: Title and Menu Bars

10.5 Forms

The Visual Administrator uses forms to present textual information for you to examine and change. These forms provide useful information about existing objects and configurations.

There are two types of forms:

- General forms tend to appear during operations or setup requests and usually accept or require user input. You access general forms via certain menu selections.
- Properties forms display detailed information about a specific object's current characteristics, some of which can be modified directly through the properties form. You access properties forms by placing the pointer on the chosen icon and then clicking MB3 (unless the icon is undergoing analysis, in which case you must use the Shift-MB3 key instead).

The following subsections describe form fields, buttons, and error messages.

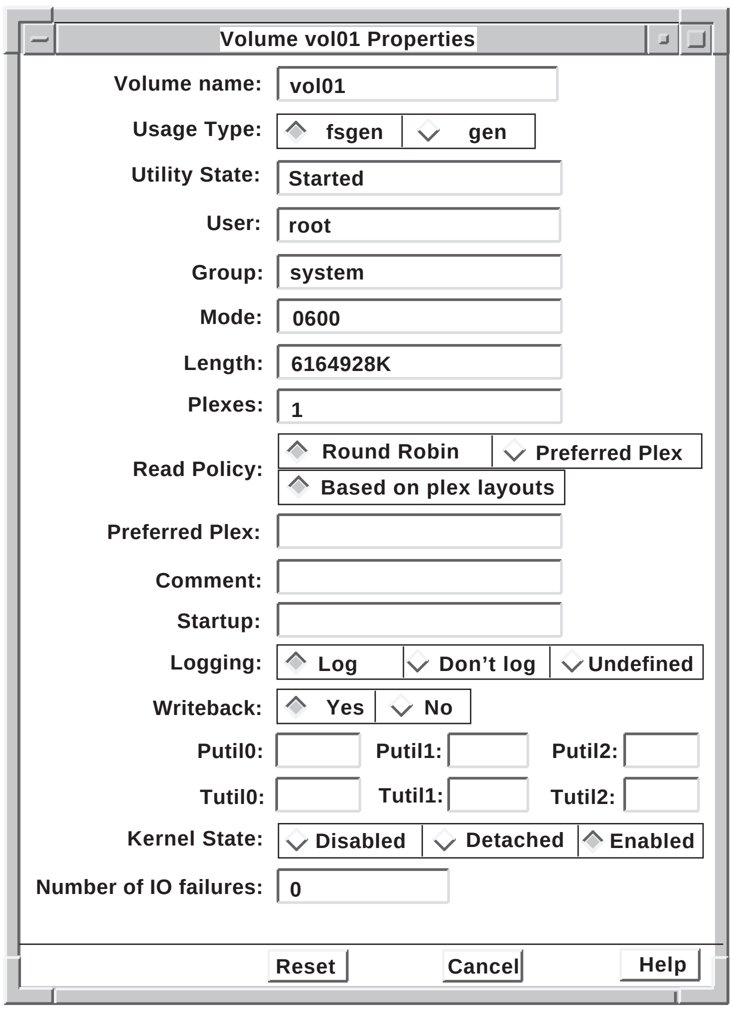
10.5.1 Fields

Many forms require input to proceed with an operation. If a required form field is either blank or incorrect, an error message will result. Other form fields already contain information (such as default values), which you may either alter or accept. Yet other form fields are read-only and therefore cannot be changed; these fields beep if you attempt to change them.

Form fields can be altered or completed in various ways, depending on the type of field. Some require text to be typed in, while others make use of assorted buttons that can be toggled or selected.

Figure 10-6 illustrates a volume properties form.

Figure 10-6: Volume Properties Form



10.5.2 Buttons

All forms have buttons on the bottom of the screen that perform standard functions:

Apply	Accepts the information on this form, checks for errors, and continues with an operation. Pressing the Return key is equal to selecting the Apply key with a form. For a properties form, the Visual Administrator issues the appropriate commands to make the changes.
Reset	Fills in the fields of the form with its default values. If the form is a properties form, the Visual Administrator uses the values that were present when the form first displayed.
Cancel	Ignores all changes made on the form and closes it. If the form was brought up as part of an operation, that operation is canceled.
Help	Displays the Help window with information about the form.

Some forms are read-only; only the the Help key and the Cancel buttons are provided on these forms.

Note

On some displays, certain forms may be too long to fit entirely on the screen and the buttons and fields at the bottom of the form may not be visible. If this is the case, the window manager's move window function (Alt - F7 , by default) can be used to move the form to view the desired buttons or fields.

10.5.3 Form Error Messages

Error messages are displayed if you select the Apply button with one or more fields incorrect on the form. A message is printed at the bottom of the form, and you can correct the values for those fields. If the error cannot be corrected or the operation is no longer desired, you should select the Cancel button.

10.6 Error and Warning Messages

The Visual Administrator uses dialog boxes to present error or warning messages. When a message is displayed in this manner, you must acknowledge it by selecting one of the buttons displayed in the error dialog box before proceeding. Some warning boxes announce that a prerequisite has not been met and require you to acknowledge this by clicking the displayed Continue button before reattempting the operation (see Figure 10-7).

Figure 10-7: Warning Box for dxlsm

10.7 Help Windows

You can access on-line help text from the menu bar of the Visual Administrator root window and the view windows. In addition, help text is available through the Help option in submenus or through the Help button at the bottom of a form.

Once you invoke Help, a Help window containing information relevant to the current window, menu, form, or operation appears. The Help window is equipped with both horizontal and vertical scrollbars, which can be used to scroll through the available Help text.

At the bottom of each Help window is a SEE ALSO area. This area lists similar or related Help topics. To access any of the listed Help topics, simply click the MB1 button on the appropriate words in the SEE ALSO list. The Help facility keeps track of the order in which Help topics are visited, so you can return to a previous topic. Selecting Previous from the menu bar displays the previous topic. Selecting Next displays the next topic.

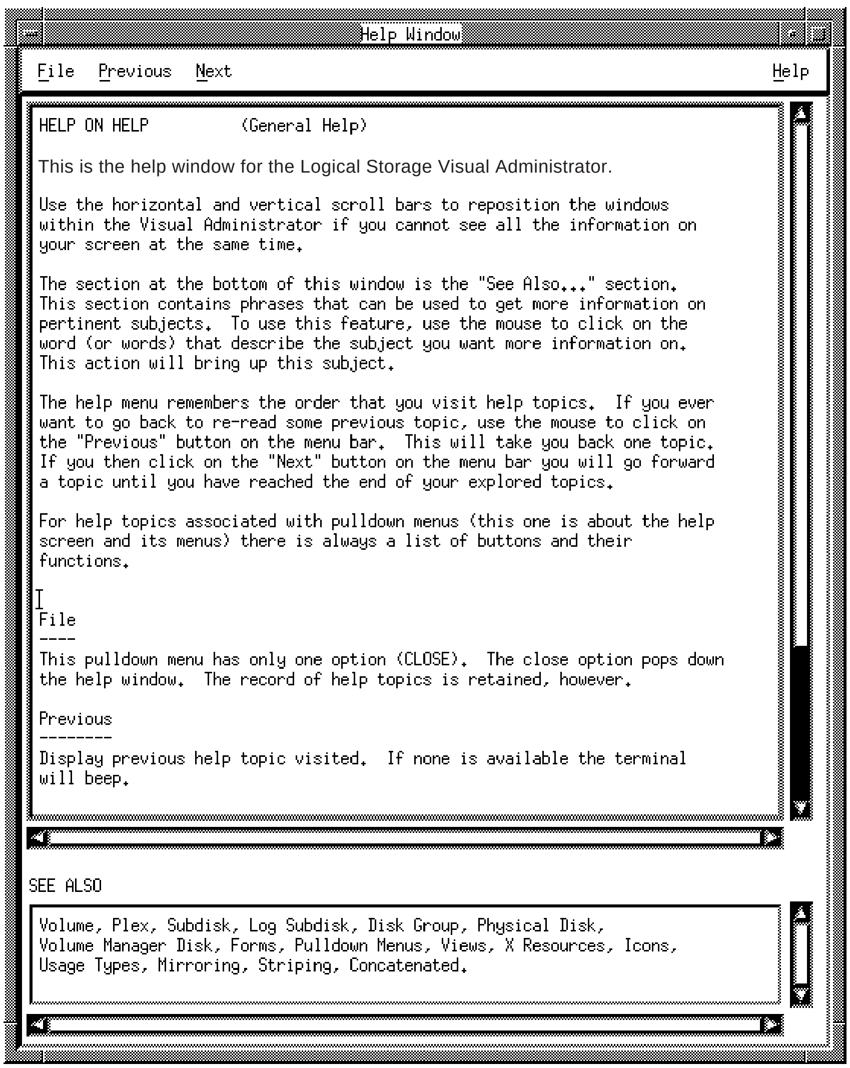
The Help menu in the menu bar of the Help window itself provides access to the following information:

General Help	Accesses Help text that includes general information on the Visual Administrator Help facility and how it is used.
Help Index	Accesses a complete listing of the available Help topics, arranged in logical groupings. Once a topic is identified from this list, that topic can be directly accessed from the SEE ALSO section of this Help window, which lists all topics alphabetically.

To close the Help window, select the Close option from the File menu. The record of help topics visited is retained.

Figure 10-8 illustrates a typical Help window.

Figure 10-8: Help Window



Visual Administrator Menus 11

This chapter gives an overview of the menus that are available from the Visual Administrator root window and the view windows. See Chapter 12 for information on individual operations.

The Visual Administrator root window has a menu bar containing the following items:

Item	Description
File	Closes the current window or exits the Visual Administrator interface completely
Views	Creates and manipulates user view windows
Options	Sets user preferences for `dxlsm` and displays the Command window
Help	Accesses the help facility

The ‘‘View of rootdg’’ window and user-created view windows have a menu bar containing the following menu items:

Item	Description
File	Closes the current window or exits the Visual Administrator completely.
Basic-Ops	Accesses basic volume, file system, and disk operations.
Advanced-Ops	Accesses advanced operations involving volumes, disks, and other LSM objects.
Analyze	Analyzes and displays the activity level of objects.
Projection	Illustrates the relationships between certain objects.
Options	Sets user preferences for using `dxlsm`. Also displays the Command Info window.
Icon	Manipulates icons.
Help	Accesses the help facility.

The following sections provide information about these menus. See Chapter

12 for information on individual operations.

11.1 File Menu

The File menu, available from both the Visual Administrator root window and the view windows, lets you close the current window or exit the Visual Administrator completely.

11.2 Views Menu

The Views menu, available only from the Visual Administrator root window, lets you create new view windows, as well as rename and remove existing user-created view windows.

11.3 Basic Operations Menu

The Basic-Ops menu, available from view windows like the ‘‘View of rootdg’’ window, can be used to perform volume management and other administrative operations. The Basic-Ops menu uses the "top-down" approach, described in Section 1.8.1.1. This approach typically makes use of the `volassist` utility, the LSM one-step automated interface. Because the `volassist` command operates automatically, it takes information about what you want to accomplish and then performs the necessary underlying tasks. This approach requires only minimal user input, but does permit more detailed specifications.

The basic operations available through the Basic-Ops menu are as follows:

Operation	Description
File System Operations	Operations involving general file system maintenance. File systems can be created, mounted, and unmounted.
Volume Operations	Operations involving general volume maintenance. Volumes can be created from LSM disks, removed, mirrored, resized, and backed up.
Disk Operations	Operations involving general disk maintenance. Disks can be initialized, removed, or replaced.

11.4 Advanced Operations Menu

The Advanced-Ops menu, available from view windows like the ‘‘View of rootdg’’ window, can be used to perform volume management and other administrative operations. The Advanced-Ops menu uses the "bottom-up" approach, described in Section 1.8.1.2. This approach typically makes use of the LSM command set, which generally requires you to specify detailed input. The LSM commands use a building block approach that requires you to have a detailed knowledge of the underlying LSM structure and components to manually perform the sequences of commands necessary to accomplish a certain task.

Note

> You should not attempt to create and manipulate volumes using the Advanced-Ops menu unless you have a solid understanding of LSM concepts.

The advanced operations available through the Advanced-Ops menu are as follows:

Operation	Description
Volume Operations	Perform operations on one or more volumes. These include creating a new volume, removing a volume, and changing the state of an existing volume.
Plex Operations	Perform operations on one or more plexes. These include creating a new plex, removing a plex, associating a plex with a volume, disassociating a plex from a volume, and detaching a plex.
Subdisk Operations	Perform operations on one or more subdisks. These include creating a new subdisk, removing a subdisk, associating a subdisk with a plex, disassociating a subdisk from a plex, and joining/splitting subdisks.
Disk Group Operations	Perform various operations on disk groups. These include creating a new disk group, adding disks to a disk group, and removing disks from a disk group.
Disk Operations	Perform various operations on disks. These include initializing a disk, removing a disk from LSM control, and onlining or offlining a disk.

11.5 Analyze Menu

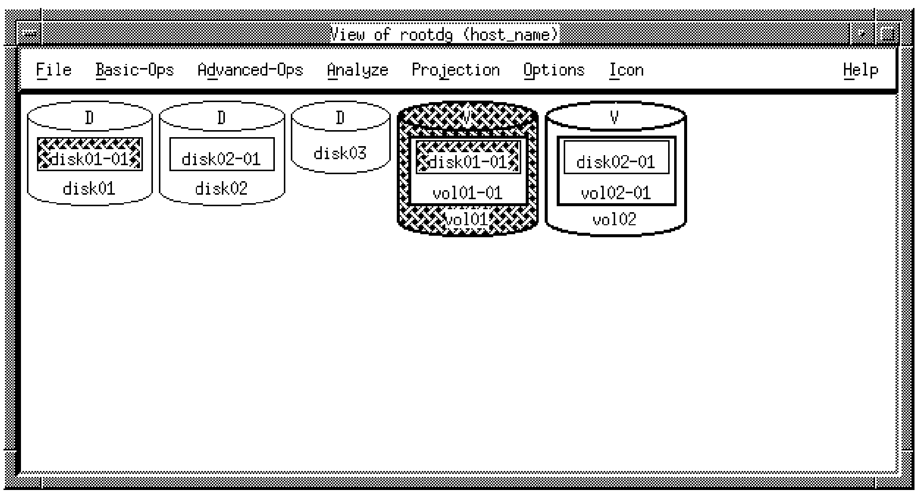
The Analyze menu, available from view windows like the ‘‘View of rootdg’’ window, lets you display statistics on the performance of LSM objects. When you start analysis, the volumes, LSM disks, and subdisks that are associated with the selected icons change their colors or patterns to reflect their relative activity levels (high, medium, or low). Once analysis is started, you can analyze additional objects by selecting their disk or volume icons.

The operations available through the Analyze menu are as follows:

Operation	Description
Start	Starts analysis on the selected LSM objects.
Stop	Stops analysis on the selected LSM objects.
Stop All	Stops analysis on all LSM objects in all views.
Parameters	Accesses the Analysis Statistics form, which sets user preferences for how analysis is performed.

Figure 11-1 illustrates analysis for a selected volume.

Figure 11-1: Analysis



11.6 Projection Menu

The Projection menu, which is available from any view window, provides access to the following operations:

Operation	Description
Icon Projection	Leads to a submenu from which you select Start or Stop for icon projection.

Operation	Description
Show Free Subdisks	Leads to a submenu from which you select Start or Stop to highlight free subdisks.

The next sections provide more information on these features.

11.6.1 Projection

Projection is the technique that the Visual Administrator uses to show relationships between icons that represent LSM objects. In addition to selecting Start from the Icon Projection submenu, you can start projection by clicking the MB2 button on an icon while holding down the Shift key (Shift-MB2). To stop projection, press Shift-MB2 again.

Projection is illustrated using color or bitmap patterns. It highlights those objects that the selected object is composed of and illustrates the relationship between the objects. For example, if a volume is selected for projection, the corresponding subdisks are highlighted within the volume icon and also on the appropriate disk icons.

When an icon is highlighted by projection, all icons representing that object in all view windows where it appears are highlighted. If the selected icon has no associated objects, the Visual Administrator issues a warning to this effect.

Volume, plex, subdisk, and LSM disk icons can be selected for projection. Projection does not apply to physical disk or partition icons.

Table 11-1 summarizes the projection relationships that are highlighted for particular icon types. If no icons of the correct type are associated with the selected icon, then nothing is highlighted.

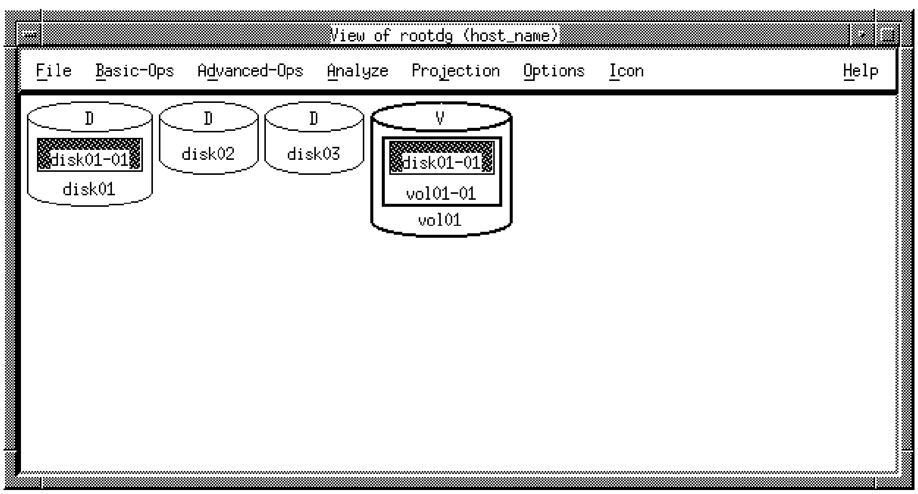
Table 11-1: Projection Behavior

Icon Selected	Icons Highlighted
Volume	All subdisks associated with any plex associated with the volume
Plex	All subdisks associated with the plex
Subdisk	Associated plex and volume, and all other subdisks associated with the plex
LSM Disk	All plexes associated with the subdisks that reside on the disk

When projection is turned on and left on from two different objects, any icon that happens to be related to both of these objects receives two layers of projection highlighting. Projection must then be turned off from both objects that started it (or all projection in the current session must be stopped) before the double-highlighted icon returns to its normal state. For example, if projection is turned on from both a volume and a plex related to the same subdisk, then that subdisk is highlighted twice even though it only appears to have one layer of highlighting.

Figure 11-2 illustrates highlighting that results from the selection of a volume icon for projection.

Figure 11-2: Projection



11.6.2 Free Subdisks

The Projection menu provides access to a feature that highlights any free subdisk icons. This is useful for identifying subdisks that are not currently associated with any plexes and should either be used or removed to free up the space that they occupy.

To turn on highlighting of free subdisks, access the Projection menu, select Show Free Subdisks, then select Start. Once turned on, free subdisks will continue to be highlighted until this feature is turned off.

11.7 Options Menu

The Options menu is available in both the Visual Administrator root window and in views windows, like the ‘‘View of rootdg’’ window. This menu allows you to specify preferences on how the Visual Administrator should operate. These preferences are saved in the file `.dxlsm_pref`, which is automatically created in the your home directory. (Each user has a personal `.dxlsm_pref` file.)

The Options menu also gives you access to the Command Info Window. The Command Info Window is a special window that displays the command history, along with the status and results of those commands. See Section 11.8 for more information about this window.

The options that can be set through the Options menu are as follows:

Option	Description
Show Command	Specifies whether the Command Info Window is to be shown before every command is executed (Show at Start) or only when a command fails (Show on Error). The default is Show on Error.
When Commands are Ready	Specifies what the Visual Administrator should do when it is ready to run an LSM or system command. The command can either be run immediately (Execute Commands) or brought up in the Command Info Window for inspection (Show Commands Only). The command can be executed directly from the Command Info Window once it is approved. The default is Execute Commands.
Logging	Logging keeps a record on disk of all commands sent to LSM or the system by the Visual Administrator. Logging can be started or stopped at any time. When started, a window requesting a log file name displays. The default is that no logging is in effect.
Popup the Command Window	This option is used to bring up the Command Info Window on demand. Commands can also be executed or repeated from this window. The Command Info Window is discussed in more detail in a later section.

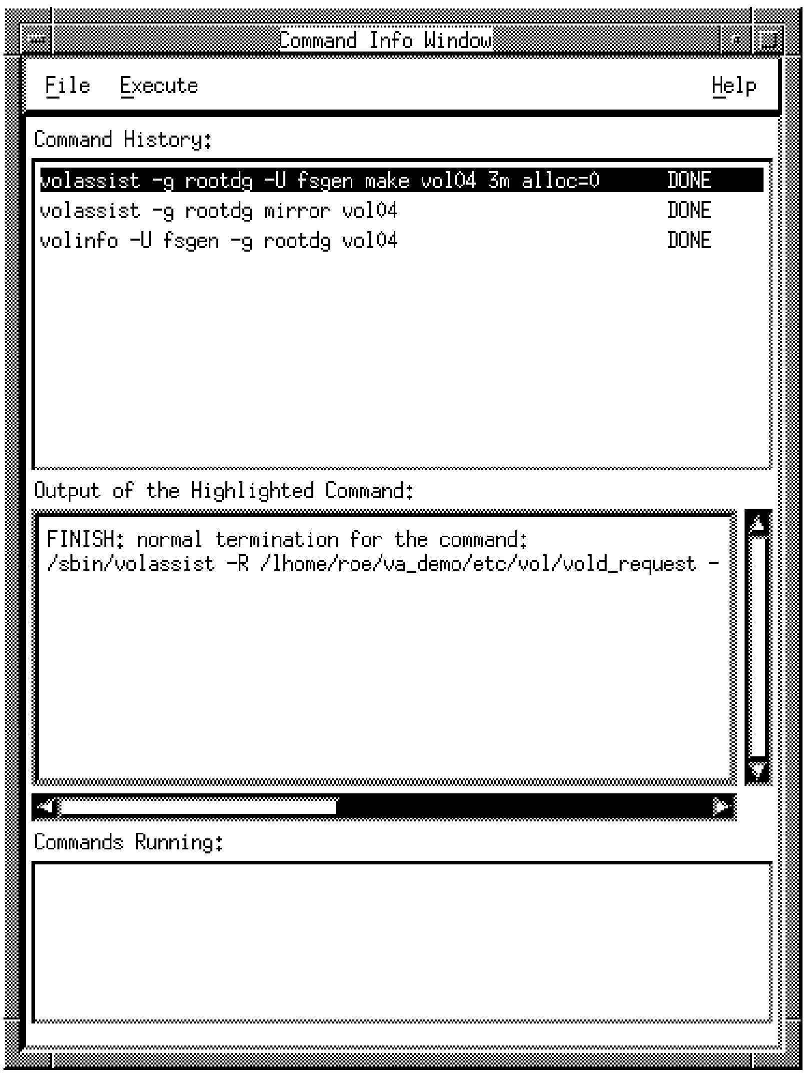
Option	Description
Format of Size	This specifies the units in which size-related output should be displayed (megabytes, kilobytes, or sectors). The unit of size applies to output only and is set to megabytes by default. In properties forms, length values are displayed as a number followed by an `s`, `m`, or `k` (representing sectors, megabytes, or kilobytes, respectively). If the size cannot be cleanly converted into megabyte or kilobyte format, it is displayed in sectors instead. The preferred unit of size does not apply to input. Input typically defaults to sectors, unless megabytes or kilobytes are specified.

11.8 Command Info Window

Command Info Window is accessed through the Options menu. It displays commands that are currently being executed by the Visual Administrator, as well as previous commands. Both LSM and system commands are displayed in this window as they are invoked by the Visual Administrator. The status and output of these commands is also displayed here. Previously executed commands can be executed again directly from this window.

Figure 11-3 illustrates the Command Info Window.

Figure 11-3: Command Info Window



The Command Info Window is divided into three sections: Command History, Output of the Highlighted Command, and Commands Running.

11.8.1 Command History

The Command History section displays a chronological listing of the commands sent to LSM or the system for execution and their status. The last ten commands are saved and displayed, with the most recent command at the bottom. If a command is too long, only the first few arguments are shown. The status of a command is one of the following:

Status	Description
DONE	Command successfully completed
BROWSE	Command not executed, just displayed here
ERROR	Command terminated with error condition
UNKNOWN	Command status cannot be determined by the Visual Administrator. This rarely occurs, but generally results from an internal Visual Administrator error or a command being interrupted unexpectedly.

If a command is selected in this window, it is shown in its entirety in the middle section of this window, along with its output.

11.8.2 Output of the Highlighted Command

When a command is highlighted in the Command History section, its information is displayed in the Output of the Highlighted Command section of the window. The results of the command are indicated here (regardless of whether it succeeded or failed), along with the full command. If the command terminated abnormally or exited with an error condition, the error message is also displayed.

11.8.3 Commands Running

The Commands Running section of the window displays the command that is currently running. This command has been sent to the system or LSM, but has not yet terminated. As soon as the command completes, it disappears from this section of the window.

11.8.4 Executing Commands

A command can be executed directly from the Command Info Window, as follows:

1. Select a command by clicking the MB1 button on the desired command in the Command History window. Once selected, the command is highlighted.

2. Access the Execute menu and choose the Execute option.

This procedure sends the selected (highlighted) command to the system or LSM for execution. This is useful for executing a command again, reexecuting a failed command that should now succeed, or executing a command that was only shown (in BROWSE status) before.

In some circumstances, the Execute with Force option may be used rather than Execute. This option adds `-f` to the executed command to force LSM to complete an operation that is considered unsafe and to disregard error messages. The `-f` option is available with some LSM commands only and does not apply to file system operations.

Note

> The Execute with Force option is a *very* dangerous operation, which can result in irreparable loss of data; it should only be used when the user is sure that an operation should succeed, even though LSM error checking prevents it.

11.9 Icon Menu

The Icon menu, available from the view windows like the ''View of rootdg'' window, lets you manipulate `dxlsm` icons. The following table summarizes the options available from the Icon menu. Note that if you have created a view of a disk group, you use the Create option from this menu to create icons for the disks in that group.

Icon Option	Description
Maximize Icons	Maximize the selected minimized icons, so that it shows all of its subicons.
Minimize Icons	Minimize the selected icons, so that it shrinks the size and hides all of its subicons.
Maximize All Icons	Maximize all icons in the current view window at once.
Create Icons	Create a copy of the icons selected from another view and place the icon copy in the current user-created view.
Remove Icons	Remove the selected icons from the current user-created view.

Using the Visual Administrator (dxlsm) 12

This chapter describes how to use the Visual Administrator to manage LSM disk configurations.

12.1 Starting the Visual Administrator

To start the LSM Visual Administrator, you must be logged into an account that has superuser privileges.

Enter the `dxlsm` command as follows:

```
# dxlsm
```

The system displays the following message in a pop up window:

```
dxlsm is coming up, please wait.
```

When `dxlsm` comes up, it displays the main LSM Visual Administrator window, called the root window, and the ''View of rootdg'' window. These windows are described in Section 10.3.

12.2 Disk Operations

The following sections describe the basic disk operations you can perform using the Visual Administrator. When managing disks, it is important that you recognize the difference between a *device name* and a *disk name*. Refer to Section 1.6.2 for more information about how LSM disks are named.

12.2.1 Initializing a Disk

Whenever you add a new disk, you must identify it to the system. This is also called initializing the disk. The system software cannot make use of the physical disk device without a software pointer to that disk. In order for LSM to be able to control a disk, that disk must also be identified to LSM.

To initialize a disk using dxlsm, follow these steps:

1. In the View of Disks window, select

   Basic-Ops → Disk Operations → Add Disks

2. In the Add Disks Form that displays, fill in the following fields:

Field	**Description**
New device name	Enter the new physical device name in rznp form. The name must be unique within the disk group.
Disk group	Enter the name of the disk group to which this disk is to be added. This field is optional.

3. Click the MB1 button on the Apply button, or press Return.

If the operation is successful, a new physical disk icon containing a partition icon appears in the View of Disks window. If a disk group was specified, a new LSM disk icon appears in the view window corresponding to the chosen disk group.

If the fstype in the disk label of the specified partition or an overlapping partition is set, LSM displays a warning message to inform you that initializing the disk might destroy existing data.

If you are sure that the disk partition has no valid data and that the partition can be added to LSM, you can use the Execute with Force option to override the warning message. See Section 11.8.4 for instructions on using this option.

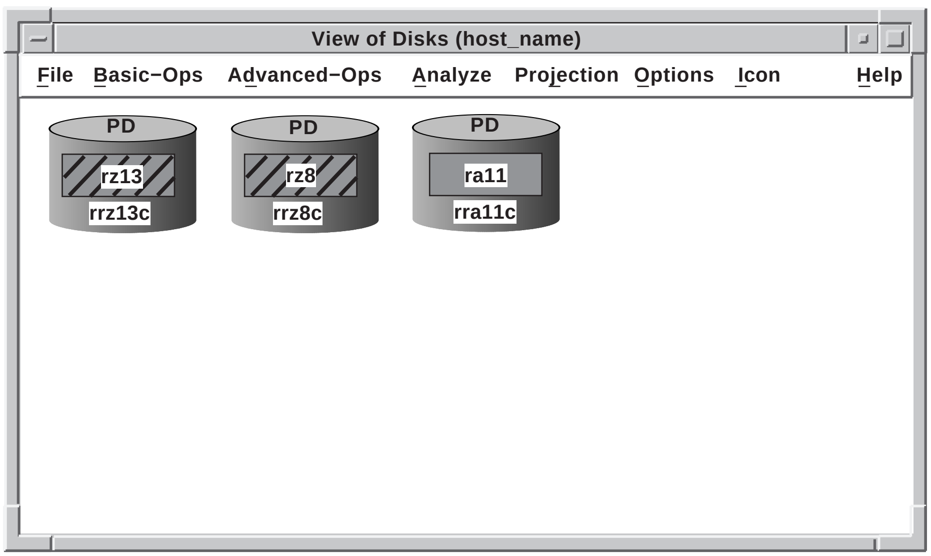
If the disk cannot be initialized because the specified partition or an overlapping partition on the disk is open (that is, a partition is actively in use by UFS, AdvFS, LSM or swap), the initialization process fails and dxlsm exits with an error message informing you of the problem.

12.2.2 Displaying Disk Information

Before you use a disk, you should confirm that it has been initialized. With the Visual Administrator, physical disk icons containing partition icons appear in the View of Disks window. Those disks that are under LSM control contain partition icons that are colored or patterned. The LSM disks corresponding to colored or patterned partition icons are displayed in the window for the disk group to which they belong. Disk groups are represented visually as disk group views rather than icons. The ‘‘View of rootdg’’ window contains icons representing all LSM disks that exist within the rootdg disk group.

To see the disk information displayed in the View of Disks window, click MB1 on the Disks button in the Visual Administrator window. A View of Disks window appears (see Figure 12-1).

Figure 12-1: View of Disks



The Visual Administrator allows you to see detailed information about a particular LSM disk in a properties form.

To display disk configuration information for a particular LSM disk, in the ‘‘View of rootdg’’ window (or appropriate disk group view), click MB3 on the disk icon whose properties you are viewing. The disk’s properties form appears, displaying detailed information about the disk.

It is possible to alter certain characteristics of the LSM disk (such as its name) via its properties form by editing the appropriate properties form field and then clicking MB1 on Apply.

12.2.3 Adding a Disk to a Disk Group

Disks are added to and grouped in disk groups for ease of administration.

To add another disk to an existing disk group, do the following:

1. In the View of Disks window, click MB1 on the partition icon representing the disk you want to add to a disk group.
2. Choose

   Advanced-Ops → **Disk Groups** → **Add Disk**

3. When the Add Disk Form appears, either leave rootdg as the default name in the Disk group field or enter the name of another existing disk group.
4. Enter the disk media name or leave the default name.
5. Click the MB1 button on Apply.

   If you look at the view window for the group you specified, you see that the new disk has been added to that disk group.

   Note

   You cannot put a disk in more than one disk group. If you attempt to do so, LSM returns an error message.

12.2.4 Renaming a Disk

Disks might be named according to their purpose (for example `docs0`), their owner (for example `smith02`), or their work group (for example `pubs01`). However, it is not necessary to give the disks special names (names such as `disk01` suffice). If the owner or purpose of the disk changes, you might want to change the disk name to reflect the change of ownership or use.

To rename a disk, do the following:

1. In the ‘‘View of rootdg’’ window (or the appropriate disk group view), click MB3 on the icon representing the disk to be renamed. This accesses the LSM disk properties form corresponding to that disk.
2. Enter the new disk name in the Logical Storage Manager disk name field in the properties form.
3. Click MB1 on Apply to change the disk name.

12.2.5 Initializing a New Disk Group

A disk group consists of one or more disks that share a common configuration. Disks are grouped together in disk groups for ease of administration.

To create a disk group, perform the following steps:

1. In the View of Disks window, select a disk by selecting the desired partition icon. This disk cannot already belong to a disk group.
2. Choose

   Advanced-Ops → Disk Groups → Initialize

3. In the Disk Group Initialize Form, fill in the Disk group field with the name you have chosen for the disk group.
4. Click MB1 on Apply to execute the initialize command.

12.2.6 Deporting a Disk Group

When a disk group is deported, access to that disk group is disabled. Disk groups may be deported if you intend to move or reuse the disks that are currently in the disk group.

To deport a disk group, do the following:

1. In the view of the disk group to be deported, choose:

   Advanced-Ops → **Disk Group** → **Deport Disk Group**

   In the Deport Disk Group Form, enter the name of the disk group to be deported.
2. Click MB1 on Apply to deport the disk group.

The deported disk group’s view disappears.

12.2.7 Importing a Disk Group

Importing a disk group enables access to a deported disk group. The disk group to be imported must have been deported at one time and at least one disk that belonged to this disk group before it was deported must remain unused.

To import a disk group, do the following:

1. In any view window, select:

   Advanced-Ops → **Disk Group** → **Import Disk Group**

   In the Import Disk Group Form, enter the name of the disk group to be imported.
2. Click the MB1 button on Apply to import the disk group.

The view button for the disk group that has been imported appears in the Visual Administrator root window.

12.2.8 Displaying Disk Group Information

With the Visual Administrator, disk groups are represented by view windows rather than icons. A view for each disk group that exists is accessible through a button in the Logical Storage Manager Visual Administrator root window.

To view the objects belonging to a particular disk group, click MB1 on the appropriate disk group button in the Logical Storage Manager Visual Administrator root window. The view window for the desired disk group appears, displaying an icon for each object that belongs in that disk group.

12.2.9 Displaying Free Space

Before you add volumes and file systems to your system, you may want to make sure you have enough free disk space to adequately meet your needs. LSM lets you request a display of free space.

To display the free space on a disk, in the ‘‘View of rootdg’’ window (or the appropriate disk group view), click MB3 on the desired LSM disk. The properties form for the selected disk appears. The last field in this window shows the maximum free space available on that particular disk. Confirm that the amount of free space is sufficient.

12.2.10 Removing a Disk

The disk hardware is removable and can be moved between systems to where it is needed the most. However, before removing the disk from the current system, you must remove the software connections the disk has with the system. First you must remove the disk from its disk group, then you can remove the disk.

To remove a disk from a disk group, do the following:

1. In the disk group view to which the disk belongs, select the LSM disk you want to remove by clicking MB1 on its icon.
2. Choose

   Advanced-Ops → **Disk Group** → **Remove Disks**

The disk icon disappears from its disk group view.

To remove a disk that no longer belongs to a disk group, do the following:

1. In the View of Disks window, select the disk you want to remove by clicking MB2 on all partitions in the disk.
2. Choose

   Advanced-Ops → **Disk** → **Remove Disks**

Note

LSM does not allow you to remove the last disk in a disk group. To remove the last disk from a disk group, deport that disk group first, then reuse the disk.

12.2.11 Replacing a Disk in Place

If a disk fails, you need to replace that disk with another. It is possible to replace a disk without disrupting the system; this is referred to as replacing a disk in place. It requires disabling and removing the failed disk and installing a new disk in its place.

To replace a disk in place, perform the following steps:

1. In the ''View of rootdg'' window (or the appropriate disk group view), click MB1 on the failed disk.
2. Select

   Basic-Ops → **Disk Operations** → **Replace Disks**

3. In the Replace Disks Form, enter the name of the replacement disk in the New physical disk name field.
4. Click the MB1 button on Apply to complete the replacement form.

12.3 Volume Operations

A volume is a virtual disk on which file systems or databases can be placed. This section describes the volume operations available with the Visual Administrator.

12.3.1 Creating a Simple Volume

To create a simple concatenated volume, perform the following steps:

1. In the ''View of rootdg'' window (or the appropriate disk group view), select

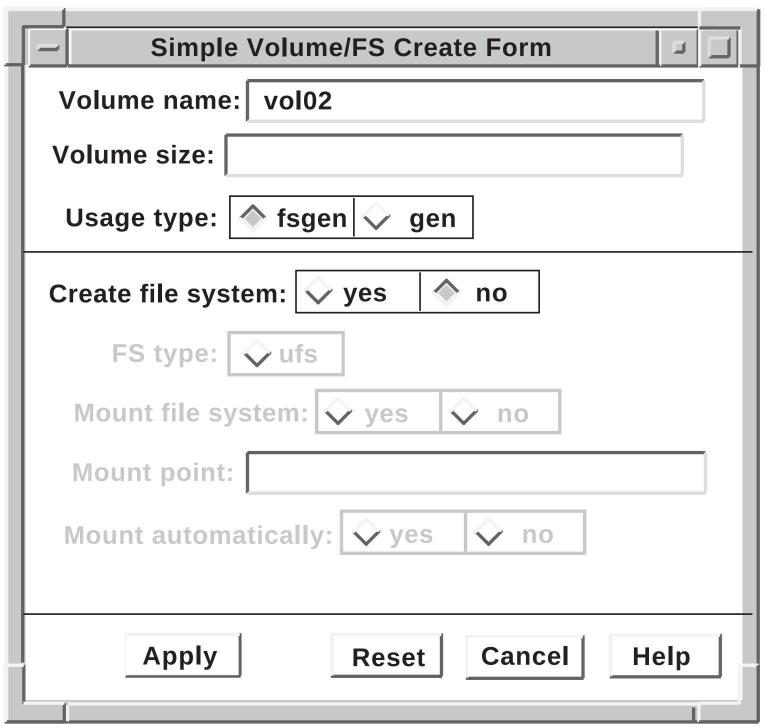
   Basic-Ops → **Volume Operations** → **Create** → **Simple**

2. In the Simple Volume/FS Create Form (Figure 12-2), either enter a name for the new volume or allow LSM to choose a default name (which appears in the Volume name field).
3. Enter a value in the Volume size field. If no unit is specified, sectors are assumed.

4. Select the Usage type. The default is `fsgen`. Use `gen` if you do not plan to create a file system on the volume.
5. Click the MB1 button on Apply to initialize the simple volume.

Figure 12-2 shows an example of this form.

Figure 12-2: Simple Volume/FS Create Form



Note

Because a file system is not created by default with this operation, the file system fields in the Simple Volume/FS Create Form are greyed out.

12.3.2 Creating a Striped Volume

A striped volume is one whose plex consists of a number of equal-sized subdisks, each located on a separate disk. Striped volumes provide faster average response time.

To create a striped volume, perform the following steps:

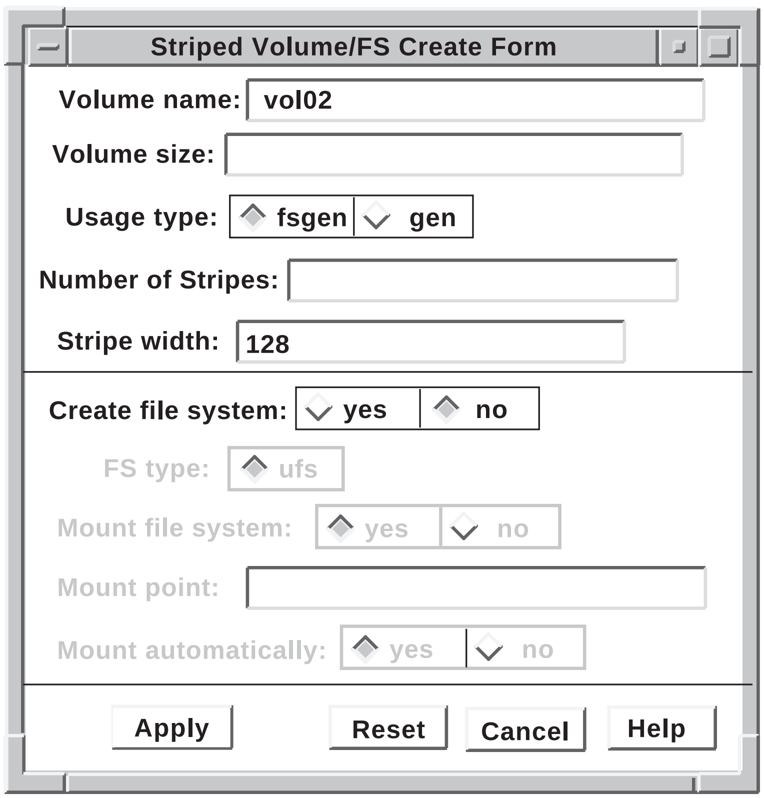
1. In the ‘‘View of rootdg’’ window (or the appropriate disk group view), select:

   Basic-Ops → **Volume Operations** → **Create** → **Striped**

2. In the Striped Volume/FS Create Form (Figure 12-3), either enter a name for the new volume or allow LSM to choose a default name (which appears in the Volume name field).
3. Enter a value in the Volume size field. If no unit is specified, sectors are assumed.
4. Select the Usage type. The default is `fsgen`. Use `gen` if you do not plan to create a file system on the volume.
5. Choose the number of stripes (which is equal to the number of disks on which the volume is to be created).
6. Choose the stripe width or let LSM use the default value.
7. Click the MB1 button on Apply to initialize the striped volume.

Figure 12-3 shows an example of this form.

Figure 12-3: Striped Volume/FS Create Form



Note

Because a file system is not created by default with this operation, the file system fields in the Striped Volume/FS Create Form are greyed out.

12.3.3 Creating a Volume on a Specific Disk

LSM automatically selects the disk or disks each volume will reside on, unless you specify otherwise. If you want a volume to reside on a specific disk, you must designate the disk for LSM.

To create a simple volume on a specific disk, do the following:

1. In the ‘‘View of rootdg’’ window (or the appropriate disk group view), click MB1 on the desired LSM disk icon.
2. Choose

   Basic-Ops → **Volume Operations** → **Create** → **Simple**

3. In the Simple Volume/FS Create Form either enter a name for the new volume or allow LSM to choose a default name (which appears in the Volume name field).
4. Enter a value in the Volume size field. If no unit is specified, sectors are assumed.
5. Choose the Usage type. `fsgen` is the default. Use `gen` if you do not plan to create a file system on the volume.
6. Click the MB1 button on Apply to initialize the simple volume.

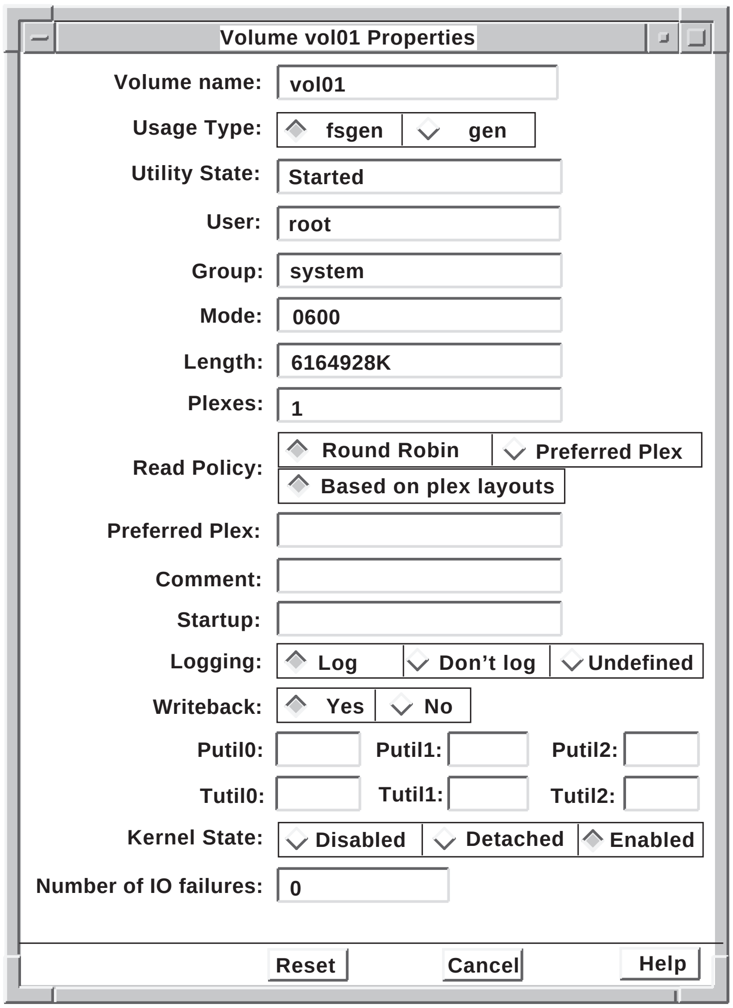
12.3.4 Displaying Volume Information

At times, you may want to see how a volume is configured. The Visual Administrator allows you to view detailed information about a particular volume in a volume properties form.

To display volume configuration information for a particular volume, in the ‘‘View of rootdg’’ window (or appropriate disk group view), click MB3 on the volume icon whose properties are to be viewed. The volume’s properties form appears, displaying detailed information about the volume (see Figure 12-4).

It is possible to alter certain characteristics of the volume (such as its name) via its properties form by editing the appropriate properties form field and then clicking MB1 on Apply.

Figure 12-4: Volume Properties Form



12.3.5 Mirroring a Volume

A mirror is a copy of a volume. The mirror copy is not stored on the same disk as the original copy of the volume. Mirroring a volume assures you that the data in that volume will not be lost if one of your disk fails.

To mirror a volume, perform the following procedures:

1. In the ‘‘View of rootdg’’ window (or appropriate disk group view), select the volume you want to mirror.

2. Choose

   Basic-Ops → **Volume Operations** → **Add Mirror**

3. At this point, indicate whether a simple or striped plex is to be created by selecting either Simple or Striped, respectively.

An additional plex appears within the volume icon's borders.

12.3.6 Extending a Volume

If the volume is not large enough for the amount of data that needs to be stored in it, you need to extend the volume's length.

To extend a volume, follow these steps:

1. In the ''View of rootdg'' window (or appropriate disk group view), select the volume you want to extend.
2. Choose

   Basic-Ops → **Volume Operations** → **Resize**

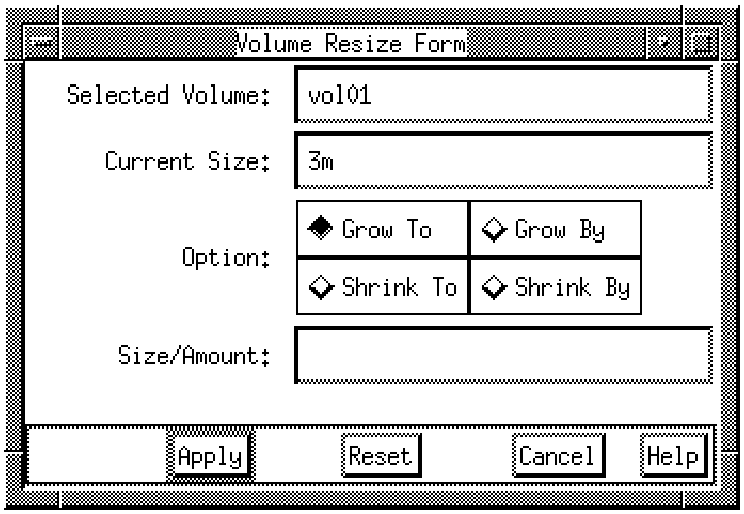
3. In the Volume Resize Form that appears (Figure 12-5), select either Grow To or Grow By in the Option field.
4. If you selected Grow To in the Option field, enter the new size of the volume in the Size/Amount field. If you selected Grow By, enter the amount by which you want the volume to grow.
5. Click the MB1 button on Apply to execute the extend command.

Notes

You cannot extend a striped volume.

To prevent accidental destruction of file systems, the extend operation will fail if a volume is open when you perform the operation. To override the failure, use the Execute menu to reissue the `volassist` command with an added `-f` force option.

Figure 12-5: Volume Resize Form



12.3.7 Shrinking a Volume

If you find that your volume is much larger than you really need it to be, you can shrink the volume's size. However, be aware that shrinking a volume containing data (perhaps in the form of a file system or database) can result in the loss of any data residing on the part of the volume that is removed.

To shrink a volume, complete the following operations:

1. In the ''View of rootdg'' window (or the appropriate disk group view), select the volume you want to shrink.
2. Choose

   Basic-Ops → **Volume Operations** → **Resize**

3. In the Volume Resize Form that appears, select either Shrink To or Shrink By in the Option field.
4. If you selected Shrink To in the Option field, enter the new size of the volume in the Size/Amount field. If you selected Shrink By, enter the amount by which you want the volume to shrink.
5. Click the MB1 button on Apply to execute the shrink command.

Notes

You cannot shrink a striped volume.

To prevent accidental destruction of file systems, the shrink operation will fail if a volume is open when you perform the operation. To override the failure, use the Execute menu to reissue the `volassist` command with an added `-f` force option.

12.3.8 Removing a Volume

Once a volume is no longer necessary (it is inactive and archived, for example), you can remove the volume and free up the disk space for other use. It is possible to remove a volume recursively, which automatically takes care of removing its associated plexes and freeing up its associated subdisks.

To remove a volume recursively, do the following:

1. In the ‘‘View of rootdg’’ window (or appropriate disk group view), select the volume you want to remove.
2. Choose

   Basic-Ops → **Volume Operations** → **Remove Volumes Recursively**

3. If the volume is enabled, a dialog box containing a warning message appears to inform you that removing an enabled volume may destroy valuable data. Selecting the Cancel button abandons the removal, while selecting OK activates the removal despite the warning.

   Click the MB1 button on the OK button to remove the volume and its components.

12.3.9 Backing Up a Volume

It is very important to make back up 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.

To back up a volume, perform the following steps:

1. In the ‘‘View of rootdg’’ window (or appropriate disk group view), select the volume you want to back up.
2. Choose

   Basic-Ops → **Volume Operations** → **Snapshot** → **Snapstart**

3. The new plex that appears in the selected volume is greyed out until it is completely updated. At this point, you may want to notify users of the upcoming snapshot and ask them to save files and temporarily reduce activity.

Note

> For UFS volumes, Digital recommends that you unmount the file system briefly to ensure the snapshot data on disk is consistent and complete.
>
> For volumes used in AdvFS domains, Digital recommends the use of an AdvFS backup method such as cloning.

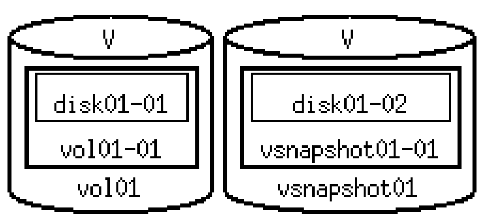
4. Choose

   Basic-Ops → **Volume Operations** → **Snapshot** → **Snapshot**

5. In the Snapshot Form that appears, either accept the default snapshot name that LSM supplies, or change the Snapshot Name field to the name of your choice.
6. Click the MB1 button on Apply to complete the backup snapshot. A new, snapshot volume appears (Figure 12-6). Normal usage of the original volume can now resume.
7. Backup the snapshot volume to tape.
8. Remove the snapshot volume (which now takes up unnecessary space) by selecting the snapshot volume and then selecting the following:

   Basic-Ops → **Volume Operations** → **Remove Volumes Recursively**

Figure 12-6: Volume and Snapshot



12.4 File System Operations

This section describes the file system management operations available with the Visual Administrator.

12.4.1 Creating a File System

With the Visual Administrator, a file system can be created on an underlying volume through a single operation. The layout of the volume on which the file system is created can be simple or striped.

12.4.1.1 Simple Volume

To create a file system on a simple concatenated volume, perform the following steps:

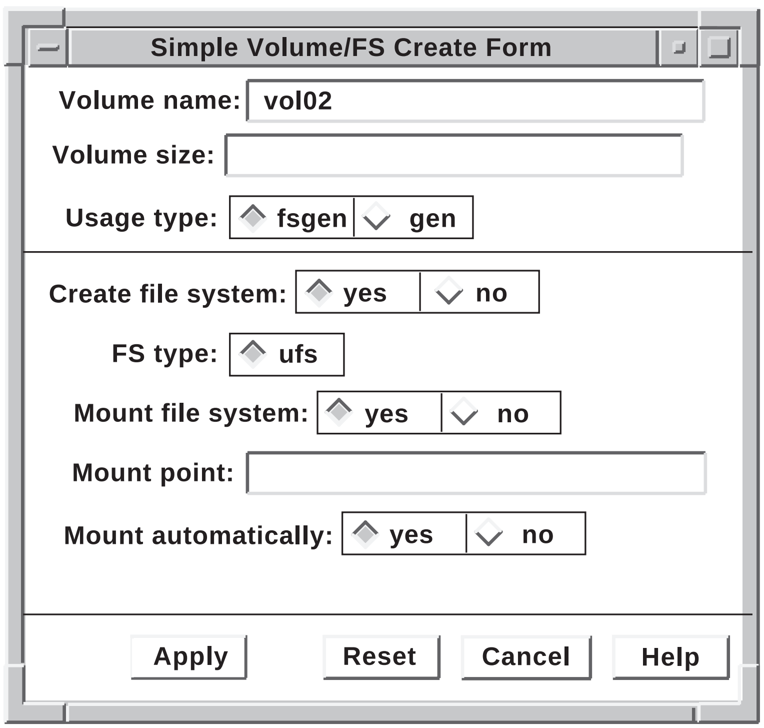
1. In the View of rootdg (or appropriate disk group view), select

   Basic-Ops → **UFS Operations** → **Create** → **Simple**

   This creates a new volume on which a new file system is created.
2. In the Simple Volume/FS Create Form (Figure 12-2), choose either a volume name for the new volume or allow LSM to choose a default name (which appears in the Volume name field).
3. Enter the desired volume size in the Volume size field. If no unit is specified, sectors are assumed.
4. Enter the Usage type . The default is `fsgen`.
5. Confirm that `Yes` is already selected in the Create file system field.
6. Choose a file system type. The default is `ufs`.
7. Choose `Yes` in the Mount file system field in order to mount the file system.
8. Enter the mount point for the new file system in the Mount point field.
9. Choose `Yes` in the Mount automatically field. This causes the file system to be added to the `/etc/fstab` file and be mounted every time the system comes up.
10. Click the MB1 button on Apply to create and mount the file system.

Figure 12-7 shows an example of this form.

Figure 12-7: Simple Volume/FS Create Form



The mount point appears below the volume icon containing the mounted file system.

12.4.1.2 Striped Volume

To create a file system on a striped volume, perform the following steps:

1. In the ‘‘View of rootdg’’ window (or appropriate disk group view), select:

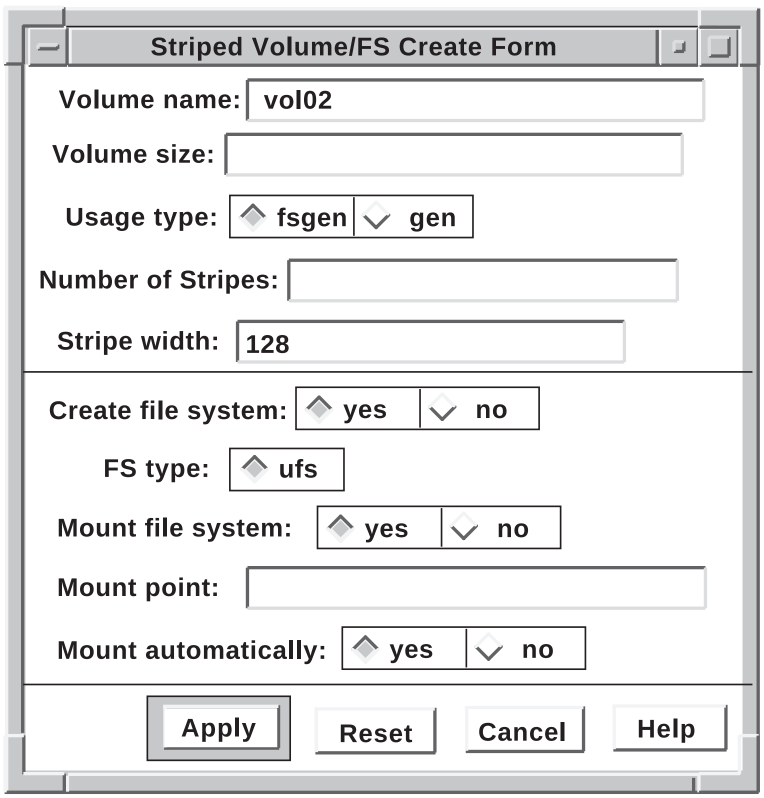
   Basic-Ops → **UFS Operations** → **Create** → **Striped**

2. In the Striped Volume/FS Create Form (Figure 12-8) either choose a volume name for the new file system or allow LSM to choose a default name (which appears in the Volume name field).
3. Enter the desired volume size in the Volume size field. If no unit is specified, sectors are assumed.

4. Enter the Usage type, the default is `fsgen`.
5. Enter the number of stripes to be created in the Number of Stripes field. This is equal to the number of disks on which the volume is to be created.
6. Enter the width of the stripes the volume will have in the Stripe width field. The default is 128 sectors.
7. Confirm that `Yes` is already selected in the Create file system field.
8. Choose a file system type. The default is `ufs`.
9. Choose `Yes` in the Mount file system field in order to mount the file system.
10. Enter the mount point for the new file system in the Mount point field.
11. Choose `Yes` in the Mount automatically field. This causes the file system to be mounted every time the system comes up.
12. Click the MB1 button on Apply to create and mount the file system.

Figure 12-8 shows an example of this form.

Figure 12-8: Striped Volume/FS Create Form



The mount point appears below the volume icon containing the mounted file system.

12.4.2 Mirroring a File System

A mirror is a copy of a volume. The mirror copy is not stored on the same disk as the original copy of the volume. Mirroring a volume containing a file system assures you that the data in that file system will not be lost if one of your disks fails.

To create a mirrored file system, perform the following steps:

1. Create a file system on a simple or striped volume, as described previously.
2. In the ''View of rootdg'' window (or the appropriate disk group view), select the volume containing the file system to be mirrored.

3. Choose

   Basic-Ops → **Volume Operations** → **Add Mirror**

4. Choose a simple or striped plex layout by selecting `Simple` or `Striped`, respectively.

12.4.3 Making a File System

With the Visual Administrator, making a file system differs from creating a file system in that a file system is made on a volume that already exists. File systems can be created and placed on existing volumes, one file system per volume.

To make a file system, perform the following steps:

1. In the ''View of rootdg'' window (or appropriate disk group view), select the volume icon on which to make the file system.
2. Choose

   Basic-Ops → **UFS Operations** → **Make File System**

3. In the Make File System Form, the name of the device on which the file system is to be made is displayed. This corresponds to the selected volume name and cannot be changed. The File system size field indicates the length of the file system to be made. This should correspond to the volume length, although it can be altered in special circumstances.
4. Choose a file system type. The default is `ufs`.
5. Choose `Yes` in the Mount file system field.
6. Enter the mount point for the new file system in the Mount point field.
7. Type `Yes` in the Mount automatically field. This causes the file system to be added to the `/etc/fstab` file and be mounted every time the system comes up.
8. Click the MB1 button on Apply to make the file system.

12.4.4 Mounting a File System

A file system may exist on a volume, without being mounted.

To mount a file system, do the following:

1. In the ''View of rootdg'' window (or appropriate disk group view), select the volume icon containing the valid, unmounted file system. (If a mount point is displayed below the volume icon, then that volume already contains a mounted file system.)

2. Choose

   Basic-Ops → UFS Operations → Mount

3. The Mount File System Form displays, containing the Device name field. This field displays the device on which to mount the file system. This corresponds to the volume you chose and you cannot change it.
4. Choose a file system type. The default is `ufs`.
5. Enter the mount point for the file system in the Mount point field.
6. Choose `Yes` in the Mount automatically field. This causes the file system to be mounted every time the system comes up.
7. Click the MB1 button on Apply to mount the file system.

When the file system is mounted, the mount point appears below the volume icon.

12.4.5 Unmounting a File System

A file system may be unmounted when it is no longer needed.

To unmount a file system, do the following:

1. In the ''View of rootdg'' window (or appropriate disk group view), select the volume whose file system you want to unmount.
2. Choose

   Basic-Ops → UFS Operations → Unmount

12.4.6 Displaying a Mounted File System

With the Visual Administrator, it is possible to view the properties of mounted file systems.

To display a file system's properties, do the following:

1. In the ''View of rootdg'' window (or appropriate disk group view), select the volume whose file system properties are to be displayed.
2. Choose

   Basic-Ops → UFS Operations → Display Properties

3. The File System Properties Form appears and displays detailed information about the file system mounted on the selected volume. You can select different mounted file systems from a menu box displayed in the upper left corner of this form.

12.5 Quitting dxlsm

To close a Visual Administrator window, select the following from the menu bar of that window:

File→Close

To end a `dxlsm` session completely, select the following from the menu bar area of any window.

File→Exit

A dialog box displays confirming that the `dxlsm` session is to be closed completely.

Part 5: Performance Tuning and Error Recovery

13 Advanced Volume Management Concepts

The two basic functions of the LSM software are to satisfy read and write requests, and to ensure that data is available. LSM automatically ensures that when you mirror disks, the corresponding plexes contain the same data. Under certain circumstances, LSM must perform a copy operation to ensure that corresponding plexes are mirrors of each other.

The state of LSM plexes and volumes can vary during the life of the LSM configuration. Changes in the composition of the volume are inevitable because:

- Disk drives occasionally need corrective maintenance
- New disks are added to replace other disks
- System failures occur, requiring copy operations to take place within the LSM volume

This chapter describes how LSM manages plexes and volumes, and provides information for administrators who want to understand LSM plex and volume states, usages, and policies.

13.1 Plex States

System administrators look at *plex states* to see whether or not plexes are complete and consistent copies of the volume contents. Although the LSM utilities automatically maintain a plex's state, it is possible for you to modify the state of a plex if necessary. For example, if a disk with a particular plex located on it begins to fail, you can temporarily disable the plex. See the `volume init` command on the `volume`(8) reference page and Section 7.6 for information about modifying plex states.

LSM utilities use plex states to:

- Indicate whether volume contents have been initialized to a known state
- Determine if a plex contains a valid copy of the volume contents
- Track whether a plex was in active use at the time of a system failure
- Monitor operations on plexes

Plex states are an important aspect of high data availability. The following subsections describe all of the different plex states, how they change to indicate abnormalities, and what LSM does to normalize the plex state again.

Plexes that are associated with a volume will be in one of the states shown in Table 13-1.

Table 13-1: LSM Plex States

Plex State	Description
ACTIVE	A plex can be in the ACTIVE state in two situations: 1) When the volume is started and the plex fully participates in normal volume I/O (meaning that the plex contents change as the contents of the volume change) and, 2) When the volume was stopped as a result of a system crash and the plex was ACTIVE at the moment of the crash. Because of the impossibility of making atomic changes to more than one plex, a system failure may leave plex contents in an inconsistent state. When a volume is started, LSM performs a recovery action to guarantee that the contents of the plexes that are marked as ACTIVE are made identical. **Note** On a system that is performing well, you should see most volume plexes in the ACTIVE plex state.
CLEAN	A plex is in a CLEAN state when it contains a consistent copy of the volume contents and a `volume stop` 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.
EMPTY	Volume creation sets all plexes associated with the volume to the EMPTY state to indicate to the usage type utilities (volume usage types are discussed later in this chapter) that the volume contents have not yet been initialized.
IOFAIL	On the detection of a failure of an ACTIVE plex, `vold` places that plex in the IOFAIL state so that it is disqualified from the recovery selection process at volume start time.
OFFLINE	The `volmend off` operation 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 until the plex is reattached with the `volplex att` operation. When this occurs, the plex is placed in the STALE state, which causes its contents to be recovered.

Table 13-1: (continued)

Plex State	Description
STALE	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 a `volume stop` operation sets the state of the plex to STALE. A `volume start` operation revives STALE plexes 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 `volplex det` operation.
TEMP	Setting a plex to the TEMP state facilitates 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 will set the plex state to TEMP at the start of such an operation and to a final state at the end of the operation. If the system goes down for any reason, a TEMP plex state indicates that the operation is incomplete; a `volume start` will disassociate plexes in the TEMP state.
TEMPRM	A TEMPRM plex state resembles a TEMP state except that at the completion of the operation, 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 goes down for any reason, the TEMPRM state indicates that the operation did not complete successfully. A subsequent `volume start` operation will disassociate and remove TEMPRM plexes.

13.1.1 Plex State Cycle

Plex states change as a normal part of disk operations. However, deviations in plex state indicate abnormalities that LSM must normalize. Table 13-2 describes possible failure scenarios and the actions LSM takes to fix deviations.

Table 13-2: How LSM Handles Changes in Plex States

Cycle	During Normal Operations	When a Crash Occurs
Startup	The `volume start` operation makes all CLEAN plexes ACTIVE. The plexes remain marked as ACTIVE unless a crash occurs.	If a crash occurs between startup and shutdown, the volume-starting operation does not find any CLEAN plexes, only ACTIVE plexes. The operation then establishes one plex as an up to date and suitable source for reviving the other plexes. LSM marks that source plex ACTIVE and marks all others as STALE. The volume-usage type determines which plex is selected as the source plex.[a] [b]
Shutdown	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 `volume start` makes them ACTIVE) indicates a normal shutdown and optimizes startup.	If an I/O error occurred and caused a plex to become disabled, the volume-stopping operation marks the plex in which the error occurred as STALE. Any STALE plexes require recovery. When the system restarts, a utility copies data from an ACTIVE to a STALE plex and makes the STALE plex ACTIVE.

Table Note:

a. Any plex can serve as the source for generic (`gen`) usage type volumes. The most up-to-date plex is selected for file system generic (`fsgen`) volumes. If the startup operation finds neither a CLEAN nor an ACTIVE plex, the system administrator must use `volmend` to select a plex to be set to CLEAN.

b. If the volume has the writeback-on-read flag set, all ACTIVE plexes are attached to the volume as ACTIVE plexes. A process is forked which reads the entire volume and the read data is written to the remaining plexes. Refer to Section 13.1.6 for more details.

13.1.2 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), and online (ENABLED) mode of operation. These modes are described in the following table.

Mode	Description
DISABLED	The plex may not be accessed.
DETACHED	A write to the volume is not reflected to the plex. A read request from the volume will never be satisfied from the plex device. Plex operations and `ioctl` functions are accepted.
ENABLED	A write request to the volume will be reflected to the plex, if the plex is set to ENABLED for write mode. A read request from the volume is satisfied from the plex if the plex is set to ENABLED.

13.1.3 Plex Layout Policy

Plexes are logical groupings of subdisks that create an area of disk space that is independent of any physical disk size. The subdisks that make up a plex can be filled with data in two ways, *concatenation* or *striping*:

- *Concatenation* places sequentially written data on subdisks in the order that the data was created. The first subdisk is filled, then the second, and so on.
- *Striping* alternates sections of plex data among multiple disks. To accomplish this, the subdisks forming a plex must all be of equal size and are divided into stripe-blocks of equal size (the stripe width). Data is placed on the subdisks by stripe; sequentially written data fills stripe block 0 on Subdisk 1 first, then stripe block 1 on Subdisk 2, and so on.

  For striping to be effective, subdisks must be spread across multiple disks, with one stripe per disk. Striping provides a performance advantage over concatenation because striping allows parallel I/O activity. You can use striping to distribute hot spots or areas of high I/O traffic across multiple devices.

See also Chapter 2 for more information about concatenation and striping.

13.1.4 Block-Change Logging

Block-change logging is a method used to dramatically reduce synchronization overhead of a mirrored volume during recovery in case of a system failure. Block-change logging keeps a log of the blocks that have changed due to I/O writes to a plex. If block-change logging is not enabled and a system failure occurs, LSM must restore all plexes to a consistent state by copying the full contents of an ACTIVE plex to the STALE plexes. This process can be lengthy and I/O intensive.

Block-change logging tracks writes by identifying and logging the block number that has changed, and stores this number in a *logging subdisk*. The block-change log maintains a record of all pending I/O records. Because the block-change log is only one block long, and because the log stores the I/O information until a process completes, the log may not be able to store all I/O changes as they occur. Once a process completes, it is flushed from the log and pending I/O information is then placed in the log.

Log records are written before the data is written. Thus, if the system experiences a crash, on system restart LSM searches the log and uses the log IDs to determine which plex contains the latest data written before the crash. In this way, plexes remain consistent, and except for possibly the last write before the crash, data is intact and up to date.

Block-change logging is enabled if two or more plexes in a mirrored volume have a logging subdisk associated with them. In addition, only the blocks recorded in the log of the ACTIVE plex need to be copied to restore the STALE plexes and maintain data integrity. For example:

```
# volmake sd rz1-01 len=1 rz1
# volsd aslog vol01-01 rz1-01
```

Block-change logging can add some overhead to your system, because LSM must perform an extra I/O for every write operation. If you want to disable block-change logging, use a command like the following:

voledit set logtype=none *volume_name*

13.1.5 Persistent State Logging

Persistent state logging ensures that only active plexes are used for recovery purposes and prevents failed plexes from being selected for recovery. Table 13-3 describes how persistent state logging solves problems due to plex failures and recovery.

Table 13-3: Recovering from Plex Failures

Problem	Resolution
When LSM error policies sometimes detach failing plexes, the state information about the failed plex is maintained on disk and is kept by the kernel as a dynamic record of the state of the configuration. However, in the event of a system failure the kernel state of the plex at the time of the failure is unknown. Thus, during recovery, LSM might select a plex that has been detached from the volume (due to an error) some time before the system failed, and the data contained in the selected failed plex could be significantly out of date.	When LSM detaches a failed plex, it immediately writes a record to the persistent state log. This way, even if a system failure occurs between the time that the plex is detached and the state change is logged, the detached plex is then disqualified from the recovery selection process. Persistent state logging can therefore guarantee that only active plexes are selected for recovery purposes.[a]
Without persistent state logging, a system crash causes all plexes to go through a recovery process, regardless of whether or not the plexes had been accessed.	Persistent state logging prevents unnecessary plex recoveries associated with started volumes that have never been accessed. Persistent state logging maintains a record of the first write to a volume and also of the last close of the volume so that no plex recovery is attempted following a crash.[b]

Table Notes:

a. A special plex state, IOFAIL, exists for failed plexes. As soon as the failure of an active plex is detected, *vold* places that plex in the IOFAIL state to ensure that it is disqualified from the selection process for any subsequent volume start operation.

b. With persistent state logging, a transaction completion record is logged whenever a transaction completes. In this way, a later recovery will be able to determine the state of pending transactions and perform the appropriate recovery action.

13.1.6 Plex Resynchronizing Policy

During a `volume start` operation for an LSM volume, it might be necessary to resynchronize plexes that have become out of date using the following steps:

1. A set of the volume's plexes are chosen as being up to date or most up to date and the volume is made available with the up-to-date plexes in the ACTIVE state.
2. The remaining plexes are then set to the STALE state and are made unavailable for reading. These plexes are brought up to date with the ACTIVE plexes through a VOL_COPY process that copies the contents of the ACTIVE plexes to the out-of-date plexes and changes their state to ACTIVE.
3. The volume is then available with all plexes active.

During the resynchronization process, however, it is possible that only one plex is active for a volume. For the duration of the resynchronization, the volume is therefore vulnerable to I/O failures because there is no reliable redundancy of the contents of the volume. This might leave the volume in an irrecoverable state if an error is encountered.

The writeback-on-read mode avoids such problems. When a read is received for a volume, the *`writeback-on-read`* flag causes the data read to be written back to all other plexes in the volume. The *`volume start`* command uses the writeback-on-read mode to start volumes with only ACTIVE plexes from which to recover.

The writeback-on-read model works as follows:

1. All plexes that are marked as ACTIVE are attached to the volume as ACTIVE upon volume startup, and the *`writeback-on-read`* flag is set for the volume.
2. The start operation then forks off a process which generates read I/O for the entire volume. The reads are serviced normally (for example, a plex from among the ACTIVE plexes is chosen via the volume read policy) and the read data is then written back to the remaining plexes.
3. When the read loop has finished, the *`writeback-on-read`* flag is unset and all the plexes of the volume are now consistent.

The success of writeback-on-read depends on the existence of Persistent State Logging, which guarantees that plexes marked as ACTIVE and not marked as DETACHED in the persistent state log area were all active at the time of the system crash. If this is the case, then the only areas of the ACTIVE plexes where data may disagree are those places that had active write I/O at the time of the crash.

Data in these areas is not guaranteed to be correct in terms of representing the data either before or after the write, but LSM guarantees that the data areas are consistent across all plexes. Writeback-on-read supports this consistency.

In addition, the failure of any plex during the synchronization process results in the normal error processing being performed without entering an irrecoverable state, because other plexes are available for use.

13.2 Volume States

Volume states indicate whether or not the volume is initialized, written to, and the accessibility of the volume.

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

Table 13-4 describes the volume states, some of which are similar to plex states.

Table 13-4: LSM Volume States

State	Description
ACTIVE	The volume has been started (kstate is currently ENABLED) or was in use (kstate 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.
CLEAN	The volume is not started (kstate is DISABLED) and its plexes are synchronized.
NEEDSYNC	The volume is not started (kstate is DISABLED) and its plexes are not synchronized. This can occur after a power failure or system failure.
EMPTY	The volume contents are not initialized. The kstate is always DISABLED when the volume is EMPTY.

Table 13-4: (continued)

State	Description
SYNC	The volume is either in read-writeback mode (kstate is currently ENABLED) or was in read-writeback mode when the machine was rebooted (kstate is DISABLED). If the volume is ENABLED, this means that the plexes are being resynchronized via the read-writeback recovery. If the volume is DISABLED, it means that the plexes were being resynchronized via read-writeback when the machine rebooted and therefore still needs to be synchronized.

The following subsections describe the different volume states, how they change to indicate abnormalities, and what LSM does to normalize the volume state.

13.2.1 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), and online (ENABLED) mode of operation. These modes are described in the following table:

Mode	Description
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.

13.2.2 Volume Usage Types

A *volume usage type* is a type label given to each volume under LSM control. Just as a file system type establishes and enforces policies for file operations, a volume usage type establishes and enforces policies for volume operations. The rules and capabilities differ for different usage types. The volume usage types affect such things as plex synchronization and error handling.

LSM provides the options described in the following table for volume usage types.

Option	Description
fsgen (file system generic)	The `fsgen` usage type assumes the volume is being used by a file system. This usage type assumes there is a way to synchronize file system data to a volume during `volplex` or snapshot procedures. It uses the file system time stamp to see which plex is most up to date. It determines the file system type and calls an appropriate procedure to do a synchronization just prior to the plex split off.
gen (generic)	The `gen` usage type makes no assumptions regarding the data content of the volume. This usage type does not handle synchronization. The `gen` option is useful for databases that reside directly on volumes.

Operations that are dependent on usage-type must determine the usage type of a volume before switching control to a utility customized for that usage type. For example, following a failure, the `gen` and `fsgen` usage types — using different algorithms — guarantee that all plexes of a volume are identical.

Note

Use the `fsgen` usage type when creating a volume if the volume is to be used by a file system. Otherwise, use the `gen` usage type.

13.2.3 Volume Read Policy

Starting a volume changes its state from DISABLED or DETACHED to ENABLED; stopping a volume changes its state from ENABLED or DETACHED to DISABLED.

Table 13-5 describes the three volume-read policies that can be selected.

Table 13-5: LSM Volume Read Policies

Policy	Description
Round	Prescribes round-robin reads of enabled plexes (this is the default read policy). If the read policy for a volume has been set to `round`, the read policy for that volume evenly distributes I/O read requests between all plexes of that volume. Reads are distributed by alternating read requests to each plex in a volume. This policy is preferred in cases where read access performance for all plexes is the same.
Prefer	Prescribes preferential reads from a specified plex. Setting the read policy to `prefer` designates a specific plex of a particular volume to be used for I/O read requests. This policy is preferred if read access performance of one plex is better than the other mirrors. For example, if one of the plexes is striped and the other plexes are concatenated.
Select	Selects a default read policy, based on the plex associations to the volume. If the volume contains a single, enabled, striped plex, the default is to prefer that plex. For any other set of plex associations, the default is to use a round-robin policy.

13.2.4 Managing Available Disk Space

Utilities such as the `volassist` utility obtain information on available disk space and use the information to calculate acceptable layouts for LSM objects. The concept of free space management is based on the idea that free space mapping can be derived by mapping out the existing allocations from the total space on the disk. The methods and specifications for dealing with the allocations and layouts may be provided at the command line; otherwise, they are obtained from defaults specified in a default file or internally.

13.3 Implementing LSM Configuration Changes

When a system administrator makes changes to a set of LSM objects, LSM groups the changes into a transaction. For any transaction, LSM ensures that either all related changes occur successfully or none of the changes are made. LSM makes all configuration changes appear to occur simultaneously and any intermediate stages of change are invisible. If a problem is encountered during the transaction, LSM does not allow any changes to occur and returns the configuration to its original state.

To achieve these atomic, all-or-nothing configuration changes, the LSM volume daemon (`vold`) envelops all configuration changes into a transaction by performing the following steps:

1. Locks all affected objects
2. Gets information about the locked records
3. Records prospective changes
4. Makes all the changes
5. Unlocks the changed objects

The result is that atomic transactions:

- Permit several system administrators to make concurrent changes to the configuration.
- Prevent inconsistent LSM configurations from occurring when there is a system failure.

If a system failure occurs during a transaction, restarting the system causes the `vold` utility to back out the partial changes. This prevents the disks maintained by LSM from becoming inconsistently configured.

Refer to the `vold`(8) reference page for additional information about the volume daemon.

Recovering from Errors 14

This chapter explains some of the procedures you can follow to recover from errors.

14.1 Protecting Your System

There are a several steps that you can take to prevent loss of data and to make it easier to recover your system, in case of failure:

- Do regular backups.

  Backups are necessary, in case 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. See Section 7.3.5 for information on how you can use the `volassist` command to reduce backup downtime.
- Mirror your root disk on a second disk, and, at the console prompt, set both a primary and an alternate boot device. See Chapter 5 for information on how to do this.
- Create and use volumes that have at least two mirrors (plexes).
- Put the mirrored plexes on different disks, and preferably on different controllers. By mirroring drives critical to booting, you ensure that no single disk failure will leave your system unusable.

  The `volassist` utility locates the plexes such that the loss of one disk will not result in a loss of data. Note that you can edit the file `/etc/default/volassist` to set the default number of plexes for newly created volumes to two.
- Use the `volsave` command to save copies of your LSM configuration files, in case you need to recreate the configuration.

14.2 Monitoring LSM Events

LSM provides the `volwatch`, `volnotify`, and `voltrace` commands to monitor LSM events and configuration changes.

The `volwatch` shell script is started automatically when you install LSM. This script sends mail to the `root` login when certain LSM configuration events occur, such as a plex detach caused by a disk failure.

The volwatch script sends mail to root by default. You can specify another login as the mail recipient.

If you need to restart volwatch, use the following command:

```
# volwatch root
```

The volnotify command is useful for monitoring disk and configuration changes and for creating customized scripts similar to /usr/sbin/volwatch.

The voltrace command provides a trace of physical or logical I/O events or error events.

For further information, refer to the volnotify(8), volwatch(8), and voltrace(8) reference pages.

14.3 Handling Common Problems

The following sections describe some of the more common problems that LSM users might encounter and suggests corrective actions.

14.3.1 An LSM Command Fails to Execute

When an LSM command fails to execute, LSM may display the following message:

```
Volume daemon is not accessible
```

This message often means that the volume daemon vold is not running.

To correct the problem, try to restart vold. Refer to Section 14.4 for detailed instructions.

14.3.2 The vold Daemon Fails to Restart

If the vold daemon fails to restart (either during system reboot or from the command line), the following message may be displayed:

```
lsm:vold: Error: enable failed: Error in disk group
configuration copies
No valid disk found containing disk group; transactions are
disabled.
```

This message could imply that the /etc/vol/volboot file has no valid disks that are in the rootdg diskgroup.

To correct the problem, update the /etc/vol/volboot file by adding disks that belong to the rootdg disk group and have a configuration copy.

Then, restart `vold`. For example:

```
# voldctl add disk rz8h
# voldctl add disk rz9
# vold -k
```

14.3.3 LSM Volume I/O or Mirroring Fails to Complete

If I/O to a LSM volume or mirroring of a LSM volume does not complete, check whether or not the LSM error daemon, `voliod`, is running on the system. Refer to Section 14.5 for details.

14.3.4 Creating a Volume or Adding a Disk Fails

When creating a new volume or adding a disk, the operation may fail with the following message:

```
No more space in disk group configuration
```

This often means that you are out of room in the disk group's configuration database. Refer to Section 6.3.8 and Section 6.3.9 for more information.

14.3.5 Mounting a File System or Opening an LSM Volume Fails

If a file system cannot be mounted or an open function on an LSM volume fails, check if `errno` is set to EBADF. This could mean that the LSM volume is not started.

Use the `volinfo` command to determine whether or not the volume is started. For example:

```
# volinfo -g rootdg
vol1            fsgen    Startable
vol-rz3h        fsgen    Started
vol2            fsgen    Started
swapvol1        gen      Started
rootvol         root     Started
swapvol         swap     Started
```

To start volume `vol1` you would enter the following command:

```
# volume -g rootdg start vol1
```

Refer to Section 7.6.4 and Section 14.9 for further information

14.4 Ensuring the Volume Configuration Daemon (vold) is Running

Before any LSM operations can be performed, the `vold` daemon must be running. Typically, the `vold` daemon is configured to start automatically during the reboot procedure. Perform the following steps to determine the state of the volume daemon:

1. Determine if the volume daemon is running and enabled by entering the `voldctl` mode command as follows:

   ```
   # voldctl mode
   ```

If...	Then...
The `vold` daemon is both running and enabled	The following message displays: **`mode:enabled`**
The `vold` daemon is running, but is not enabled	The following message displays: **`mode:disabled`**
The `vold` daemon is not running	The following message displays: **`mode:not-running`**

2. If necessary, enable the volume daemon by entering the `voldctl` enable command:

   ```
   # voldctl enable
   ```

3. If necessary, start the volume daemon by entering the `vold` command:

   ```
   # vold
   ```

For additional information about the `vold` daemon, refer to the `vold`(8) reference page.

14.5 Ensuring the Volume Extended I/O Daemon (voliod) is Running

Volume log I/O `voliod` kernel threads are started by the `vold` daemon (if block-change logging is enabled) and are killed by the kernel when these threads are no longer needed. Volume error kernel threads are started automatically by LSM startup procedures. Rebooting after your initial installation should start the `voliod` error daemon automatically.

Note

> Digital recommends that there be at least as many `voliod` error daemons as the number of processors on the system.

You can perform these steps to determine the state of the error daemon:

1. Verify that the error daemon is running and enabled by entering the following command:

   ```
   # voliod
   ```

If...	Then...
Any `voliod` processes are running	The following message displays: *n* `"volume I/O daemons running"` The *n* symbol in the previous example indicates the number of `voliod` daemons running.
There are no `voliod` daemons currently running	Start some daemons by entering the following command: `# voliod set 2`

2. If necessary, enable the volume error daemon by entering the following command:

   ```
   # voliod set 2
   ```

For more detailed information about the `voliod` daemon, refer to the `voliod`(8) reference page.

14.6 Problems Encapsulating the Root and Swap Partitions

The following sections describe two recovery procedures you can try if problems occur during the encapsulation procedure described in Section 5.2.

- If something goes wrong during conversion from the root partition to the LSM root volume, you use a process called *zapping* to undo kernel changes that were made as a result of encapsulating the root disk. Zapping is described in Section 14.6.1.
- If booting to multiuser mode is impossible after root encapsulation has succeeded, you can allow booting from the physical disk partition. See Section 14.6.2 for information about how to boot from the physical disk partition so that you can perform root maintenance.

14.6.1 Unencapsulating the Root Disk

If something goes wrong during the conversion from the root partition to the root LSM volume, the encapsulation procedure tries to back out all changes made, and restores the use of partitions for the root file system. Under some circumstances, you might need to manually undo the changes made as a result of encapsulating the root disk.

The following steps describe how to manually reset the changes made during root encapsulation:

1. Boot the system to single-user mode.
2. Enter the following command:

   ```
   # voldctl -z
   ```

3. Mount the root partition as follows:
   - If the root file system is UFS, mount the root partition with the following command:

     ```
     # mount -u /dev/rzXa /
     ```

   - If the root file system is AdvFS, mount the root partition with the following command:

     ```
     # mount -u /
     ```

4. Edit the `/etc/fstab` file as follows:
   - If the root file system is UFS, change the device-special file from `/dev/vol/rootdg/rootvol` to the `a` partition of the boot disk. Change the primary swap device from `/dev/vol/rootdg/swapvol` to the block-device file of the swap partition.

– If the root file system is AdvFS, enter the following commands:

```
# cd /etc/fdmns/root_domain
# rm rootvol
# ln -s /dev/rzxa rzxa
```

Change the primary swap device from `/dev/vol/rootdg/swapvol` to the block-device file of the swap partition.

5. Edit the `/etc/sysconfigtab` file and change the LSM entry from

   `lsm_rootdev_is_volume = 1`

   `lsm_swapdev_is_volume = 1`

   to

   `lsm_rootdev_is_volume = 0`

   `lsm_swapdev_is_volume = 0`

6. Change the `/sbin/swapdefault` file (if it exists) to be a link to the swap partition's device-special file. For example, if the disk `rz8b` is the swap partition, enter the following commands:

   ```
   # mv /sbin/swapdefault /sbin/swapdefault.swapvol
   ```

   ```
   # ln -s /dev/rz8b /sbin/swapdefault
   ```

7. Remove files that were related to the conversion:

   ```
   # rm -rf /etc/vol/reconfig.d/disk.d/*
   ```

   ```
   # rm -rf /etc/vol/reconfig.d/disks-cap-part
   ```

8. Reboot the system on the same boot disk. The system will reboot using disk partitions for root and swap.

14.6.2 Performing Root Maintenance

If you encounter problems in which booting to multiuser mode is impossible, you can use the following steps to allow booting from the physical disk partition, so that you can perform maintenance to fix the problem:

1. Use the step-by-step instructions for zapping in Section 14.6.1.
2. After the system has rebooted, use the `volmend` utility to set the good plex in your `rootvol` volume to ACTIVE. Refer to the `volmend`(8) reference page for information about fixing the volume.
3. After fixing the problem, undo the changes that you made in steps 4 through 6 in Section 14.6.1.
4. Reboot the system.

14.7 Recovering from Boot Disk Failure

When the boot disk is mirrored, failures occurring on the original boot disk are transparent to all users. However, during a failure, the system might do one or both of the following:

- Write a message to the console indicating there was an error reading or writing to the plex on the boot disk.
- Suffer from slow performance (depending on the problem encountered with the disk containing one of the plexes in the `root` or `swap` volumes).

To reboot the system before the original boot disk is repaired, you can boot from any disk that contains a valid `root` and `swap` volume plex. Chapter 5 shows how to set an alternate boot device from your system console.

If all copies of `rootvol` are corrupted, and you cannot boot the system, you must reinstall the system. Refer to Section 14.11 for details.

14.7.1 Re-adding and Replacing Boot Disks

Normally, replacing a failed disk is as simple as putting a new disk somewhere on the controller and running LSM replace disk commands. It's even possible to move the data areas from that disk to available space on other disks, or to use a "hot spare" disk already on the controller to replace the failure. For data that is not critical for booting the system, it doesn't matter where the data is located. All data that is not boot critical is only accessed by LSM after the system is fully operational. LSM can find this data for you. On the other hand, boot-critical data must be placed in specific areas on specific disks in order for the boot process to find it.

When a disk fails, there are two possible routes that can be taken to correct the action. If the errors are transient or correctable, then the same disk can be re-used. This is known as *re-adding* a disk. On the other hand, if the disk has truly failed, then it should be completely replaced.

14.7.1.1 Re-adding A Failed Boot Disk

Re-adding a disk is the same procedure as replacing a disk, except that the same physical disk is used. Usually, a disk that needs to be re-added has been *detached*, meaning that LSM has noticed that the disk has failed and has ceased to access it.

If the boot disk has a transient failure, its plexes can be recovered using the following steps. The `rootvol` and `swapvol` volumes can have two or three LSM disks per physical disk, depending on the layout of the original root disk.

1. Enter the `voldisk` command to list the LSM disks that are associated with the failed physical disk. For example:

   ```
   # voldisk list

   DEVICE       TYPE      DISK     GROUP     STATUS
   rz10         sliced     -         -       error
   rz10b        nopriv     -         -       error
   rz10f        nopriv     -         -       error
   rz21         sliced    rz21     rootdg    online
   rz21b        nopriv    rz21b    rootdg    online
   -              -       rz10     rootdg   removed was:rz10
   -              -       rz10b    rootdg   removed was:rz10b
   -              -       rz10f    rootdg   removed was:rz10f
   ```

   In this example, if `rz10` was the failed boot disk, then you can assume that `rz10`, `rz10b`, and `rz10f` are the LSM disks associated with the physical disk `rz10`.

2. Enter the following commands to add the LSM disks back to the `rootdg` disk group:

   ```
   # voldisk online rz10 rz10b rz10f

   # voldg -k adddisk rz10=rz10

   # voldg -k adddisk rz10b=rz10b

   # voldg -k adddisk rootrz10=rz10f
   ```

3. After the disks have been added to the `rootdg` disk group, enter the `volrecover` command to resynchronize the plexes in the `rootvol` and `swapvol` volumes. For example:

   ```
   # volrecover -sb rootvol swapvol
   ```

14.7.1.2 Replacing a Failed Boot Disk

If a boot disk that is under LSM control fails and you are replacing it with a new disk, perform the following steps:

1. Disassociate the plexes on the failed disk from `rootvol` and `swapvol`.
2. Remove the failed LSM disks from the disk group. Refer to `volplex`(8), `voldg`(8), and `voldisk`(8) for more information about how to accomplish this.
3. Mirror the `rootvol` and `swapvol` volumes onto the new disk, as described in Section 5.3.1. The replacement disk should have at least as much storage capacity as was in use on the old disk.

14.7.2 Stale or Unusable Plexes on Boot Disk

If a disk is unavailable when the system is running, any plexes of volumes that reside on that disk will become stale, meaning the data on that disk is out of date relative to the other plexes of the volume.

During the boot process, the system accesses only one copy of the root and swap volumes (the copies on the boot disk) until a complete configuration for those volumes can be obtained. If it turns out that the plex of one of these volumes that was used for booting is stale, the system must be rebooted from a backup boot disk that contains nonstale plexes. This problem can occur, for example, if the boot disk was replaced and restarted without adding the disk back into the LSM configuration. The system will boot normally, but the plexes that reside on the newly powered disk will be stale.

Another possible problem can occur if errors in the LSM headers on the boot disk prevents LSM from properly identifying the disk. In this case, LSM will be unable to know the name of that disk. This is a problem because plexes are associated with disk names, and therefore any plexes on that disk are unusable.

If either of these situations occurs, the LSM daemon `vold` will notice it when it is configuring system as part of the `init` processing of the boot sequence. It will output a message describing the error, describe what can be done about it, and halt the system. For example, if the plex `rootvol-01` of the root volume `rootvol` on disk `disk01` of the system was stale, `vold` would print the following message:

```
lsm:vold: Warning Plex rootvol-01 for root volume is stale or unusable.
lsm:vold: Error: System boot disk does not have a valid root plex
Please boot from one of the following disks:

        Disk: disk02                    Device: rz2

lsm:vold: Error: System startup failed
```

This informs the administrator that the disk `disk02` contains usable copies of the root and swap plexes and should be used for booting. This is the name of the system backup disk. When this message appears, the administrator should reboot the system from a backup boot disk.

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

If the plexes on the boot disk were unavailable, the administrator should get mail from the LSM `volwatch` utility describing the problem. Another way to discover the problem is by listing the disks with the `voldisk` utility. In the previous example, if the problem is a failure in the private area of `disk01` (such as due to media failures or accidentally overwriting the LSM private region on the disk), enter the following command:

```
# voldisk list
```

This command produces the following output:

```
DEVICE       TYPE      DISK     GROUP     STATUS
-            -         disk02   rootdg    failed was:  rz1
rz2          sliced    disk02   rootdg    online
```

14.7.3 Crash Dumps

If a system failure occurs, the system console writes a crash dump to the boot disk. However, if the original boot disk has had a problem such that the corresponding plex in the root or swap volumes has been disabled, then the crash dump is written to the first available plex in the swap volume. The system reports the name of the disk that has the crash dump by printing a message on the system console.

For example, the following messages are printed to the console along with other dump information:

```
WARNING: LSM: Original dump device not found
LSM attempting to dump to SCSI device unit number rz1
```

To obtain the crash dump when the system reboots, you must boot the system from the disk that contains the crash dump.

14.8 Recovering from Disk Problems

The following sections describe recovery procedures for problems related to LSM disks.

14.8.1 Detecting Failed Disks

If one plex of a volume encounters a disk I/O failure (for example, because the disk has an uncorrectable format error), one of the the following may happen:

- LSM may detach the plex.

  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, LSM may 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 disabled as well.

If a volume, a plex, or a disk is detached by failures, the `volwatch`(8) utility sends mail to `root` indicating the failed objects. For example, if a disk containing two mirrored volumes fails you might receive a mail message similar to the following:

```
To: root
Subject: Logical Storage Manager failures on mobius.lsm.com

Failures have been detected by LSM on host
mobius.lsm.com:

failed plexes:
 home-02
  src-02

No data appears to have been lost.  However, you should replace
the drives that have failed.
```

To determine which disks are causing the failures in this message, enter the following command:

```
# volstat -sff home-02 src-02
```

This produces output such as the following:

```
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` (the basename for the displayed subdisks).

Sometimes these errors are caused by cabling failures. You should look at the cables connecting your disks to your system. If there are any obvious problems, correct them and recover the plexes with the following command:

```
# volrecover -b home src
```

This command starts a 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, replace the disk.

If you do not see any obvious cabling failures, then the disk probably needs to be replaced.

If a disk fails completely, the mail message will list the disks that have failed, all plexes that use the disk, and all volumes defined on the disk that was disabled because the volumes were not mirrored. For example:

```
To: root
Subject: Logical Storage Manager failures on mobius.lsm.com

Failures have been detected by LSM on host
mobius.lsm.com:

failed disks:
 disk02

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

failed volumes:
 mkting

The contents of failed volumes may be corrupted, and should be
restored from any available backups.  To restart one of
these volumes so that you can restore it from backup, replace
disks as appropriate then use the command:

        volume -f start <volume-name>

You can then restore or recreate the volume.
```

This message indicates that `disk02` was detached by a failure; that plexes `home-02`, `src-02`, and `mkting-01` were also detached (probably because of the failure of the disk); and that the volume `mkting` was disabled.

Again, the problem may be a cabling error. If the problem is not a cabling error, then you must replace the disk.

14.8.2 Replacing a Failed Disk

Disks that have failed completely, and that have been detached by failure, can be replaced by running the `voldiskadm` menu utility and selecting item 5, `Replace a failed or removed disk,` from the main menu. If you have any disks that are initialized for LSM but have never been added to a disk group, you can select one of those disks as a replacement. Do not choose the old disk drive as a replacement even though it may appear in the selection list. If there are no suitable initialized disks, you can choose to initialize a new disk.

If a disk failure caused a volume to be disabled, then 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 `volinfo` command.

Any volumes that are listed as `Unstartable` must be restored from backup. For example, the `volinfo` command might display:

```
home            fsgen    Started
mkting          fsgen    Unstartable
src             fsgen    Started
```

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

```
# volume -obg -f start mkting
```

The `-obg` option causes any plexes to be recovered in a background task.

14.8.3 Replacing a Disk that is Beginning to Fail

Often a disk has recoverable (soft) errors before it fails completely. If a disk is getting an unusual number of soft errors, replace it. This involves two steps:

1. Detaching the disk from its disk group
2. Replacing the disk with a new one

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

When you select a disk to remove for replacement, all volumes that will be affected by the operation are displayed. For example, the following output might be displayed:

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

        lhome 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 do this? [y,n,q,?] (default: n)
```

If any volumes would be disabled, quit from `voldiskadm` and save the volume. Either back up the volume or move the volume off of the disk. To move the volume mktinq to a disk other than disk02, use the command:

```
# volassist move mkting disk02
```

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

After the disk has been removed for replacement, specify a replacement disk by selecting item 5, Replace a failed or removed disk, from the main menu in voldiskadm.

Refer to Section C.10 for examples of how to replace disks.

14.8.4 Modifying the Disk Label to Start at Block 1 Instead of Block 16

In LSM Version 1.0, disks added to LSM skip physical block 0 and start at block 1 because block 0 contains the disk label and is write-protected.

Starting with LSM Version 1.1, disks added to LSM start at physical block 16 for performance reasons with certain disks. To start a disk at physical block 1 instead of block 16, use the disklabel command to modify the partition start offset and length accordingly before adding the disk to LSM.

For example:

```
# disklabel -e /dev/rrz16c
# voldisk init rz16 type=sliced
```

Refer to the disklabel(8) reference page for details.

14.9 Recovering Volumes

The following sections describe recovery procedures for problems relating to LSM volumes.

14.9.1 Listing Unstartable Volumes

An unstartable volume is likely to be incorrectly configured or has other errors or conditions that prevent it from being started. To display unstartable volumes, use the volinfo command, which displays information on the accessibility and usability of one or more volumes:

```
# volinfo -g diskgroup [volname]
```

14.9.2 Recovering a Disabled Volume

If a system crash or an I/O error corrupts one or more plexes of a volume and no plex is CLEAN or ACTIVE, 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 a CLEAN state, use the following command:

```
# volmend fix clean plex_name
```

For example, the command line to place one plex labeled `vol01-02` in the CLEAN state looks like this:

```
# volmend fix clean vol01-02
```

Refer to the `volmend`(8) reference pages for more information.

14.10 Problems with volrestore

If you used the `volsave` command to save a copy of your configuration, you can use the `volrestore` command to restore the configuration. This section describes problems that may arise in restoring a configuration.

See Section 7.4 and Section 7.5 for information on `volsave` and `volrestore`. See Appendix C for examples of handling restore failures.

14.10.1 Conflicts While Restoring the Configuration

When `volrestore` executes, it can encounter conflicts in the LSM configuration, for example, if another volume uses the same plex name or subdisk name, or the same location on a disk. When `volrestore` finds a conflict, it displays error messages and the configuration of the volume, as found in the saved LSM description set. In addition, it removes all volumes created in that disk group during the restoration. The disk group that had the conflict remains imported, and `volrestore` continues to restore other disk groups.

If `volrestore` fails because of a conflict, you can use the `-b` option to do the ‘‘best possible’’ restoration in a disk group. You will then have to resolve the conflicts and restore the volumes in the affected disk group.

See Section C.26 for further information and examples.

14.10.2 Failures in Restoring the Configuration

The restoration of volumes fails if one or more disks associated with the volumes are unavailable, for example due to disk failure. This, in turn, can cause the restoration of a disk group to fail. You can use a command like the following to restore the LSM configuration of a disk group:

```
# volrestore -b -g diskgroup
```

The volumes associated with the failed disks can then be restored by editing the `volmake` description file to remove the plexes that use the failed disks. Note that editing the description file will affect the checksum of the files in the backup directory, so you will have to override the checksum validation by using the `-f` option.

See Section C.26 for further information and examples.

14.11 Reinstallation Recovery

Occasionally, your system may need to be reinstalled after some types of failures. Reinstallation is necessary if all copies of your root (boot) disk are damaged, or if certain critical files are lost due to file system damage. When a failure of either of these types occurs, you must reinstall the entire system.

If these types of failures occur, attempt to preserve as much of the original LSM 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.

The following sections describe the procedures used to reinstall LSM and preserve as much of the original configuration as possible after a failure.

14.11.1 General Recovery Information

A system reinstallation completely destroys the contents of any disks that are reinstalled. Any LSM related information, such as data in the LSM private areas on reinstalled disks (containing the disk identifier and copies of the LSM configuration), is removed during reinstallation. The removal of this information makes the disk unusable as an LSM disk.

If a disk was placed under LSM control (either during the LSM installation or by later encapsulation), that disk and any volumes on it are lost during reinstallation. If a disk was not under LSM control before the failure, no volumes are lost at reinstallation. You can replace any other disks by following the procedures in Section 9.2.6,

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, the reinstallation, or both; volumes on disks involved with the failure or reinstallation are lost during reinstallation. If backup copies of these volumes are available, you can restore them after reinstallation. The system root disk is always involved in reinstallation. Other disks may also be involved.

If the root disk was placed under LSM control by encapsulation, that disk and any volumes or volume plexes on it are lost during reinstallation. In addition, any other disks that are involved in the reinstallation (or that are removed and replaced), also lose any LSM data (including volumes and plexes).

If a disk (including the root disk) is not under LSM control prior to the failure, no volumes are lost at reinstallation. Although having the root disk under LSM control simplifies the recovery process after reinstallation, not having the root disk under LSM control increases the likelihood of a reinstallation being necessary. Having the root disk under LSM control, and creating plexes of the root disk contents, eliminates many of the problems that require system reinstallation.

14.11.2 Overview of Reinstallation and Reconfiguration Procedures

To reinstall the system and recover the LSM configuration you need to perform the following procedures:

1. Prepare the system for installation. This includes replacing any failed disks or other hardware, and detaching any disks not involved in the reinstallation.
2. Save the current copy of `/etc/vol/volboot`.
3. Install the operating system.
4. Recover the LSM configuration. Restore the saved copy of `/etc/vol/volboot`.
5. Cleanup the configuration. This includes restoring any information in volumes affected by the failure or reinstallation.

Each of these procedures is described in detail in the sections that follow.

14.11.3 Preparing the System for Reinstallation

To prevent the loss of data on disks not involved in the reinstallation, you should only involve the root disk in the reinstallation procedure. It is recommended that any other disks (that contain volumes) be disconnected from the system before you start the reinstallation procedure. Disconnecting the other disks ensures that they are unaffected by the reinstallation. For example, if the operating system was originally installed with a file system on the second drive, the file system may still be recoverable. Removing the second drive ensures that the file system remains intact.

14.11.4 Reinstalling the Operating System

Once any failed or failing disks have been replaced and disks uninvolved with the reinstallation have been detached, reinstall the operating system as described in the *Installation Guide*.

While the operating system installation progresses, make sure no disks other than the root disk are accessed in any way. If anything is written on a disk other than the root disk, the LSM configuration on that disk could be destroyed.

14.11.5 Recovering the LSM Configuration

Once the LSM subsets have been loaded, recover the LSM configuration by doing the following:

1. Shut down the system.
2. Physically reattach the disks that were removed from the system.
3. Reboot the system. When the system comes up, make sure that all disks are configured in the kernel and that special device files have been created for the disks.
4. Run the `volinstall` script to create LSM special device files and to add LSM entries to the `/etc/inittab` file:

   ```
   # volinstall
   ```

5. Bring the system to single-user mode by entering the following command:

   ```
   # shutdown now
   ```

6. You need to remove some files involved with installation that were created when you loaded LSM but are no longer needed. To do this, enter the following command:

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

7. Once these files are removed, you must start some LSM daemons. Start the daemons by entering the command:

   ```
   # /sbin/voliod set 2
   ```

8. Start the LSM configuration daemon, `vold`, by entering the command:

   ```
   # /sbin/vold -m disable
   ```

9. If a copy of `/etc/vol/volboot` exists on backup media, restore it. Go to the next step.

   If a saved copy of `/etc/vol/volboot` does not exist, initialize `/etc/vol/volboot` by entering:

   ```
   # voldctl init
   ```

   Add one or more disks that have configuration databases to the `/etc/vol/volboot` file. You must do this otherwise LSM cannot restart after a reboot.

   To reenable the previous LSM configuration, you need to determine the name of one of the disks that was in the `rootdg` disk group. If you do not know the name of one of the disks, you can scan the disk label on the disks available on the system for LSM disk label tags such as `LSMpubl` or `LSMsimp`. If you find the `LSMpubl` disk label tag on a disk, add the disk as an LSM sliced disk. If you find the `LSMsimp` disk label tag, add the partition as an LSM simple disk.

```
# voldctl add disk rz3
```

10. Enable `vold` by entering:

```
# voldctl enable
```

11. Start LSM volumes by entering:

```
# volrecover -sb
```

The configuration preserved on the disks not involved with the reinstallation has now been recovered. However, because the root disk has been reinstalled, it appears to LSM as a non-LSM disk. Therefore, the configuration of the preserved disks does not include the root disk as part of the LSM configuration.

Note

> If the root disk of your system and any other disk involved in the reinstallation were not under LSM control at the time of failure and reinstallation, then the reconfiguration is complete at this point. If any other disks containing volumes or volume plexes are to be replaced, follow the replacement procedures in Chapter 6. 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 volume or volume plexes 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 on that the volume is lost and must be restored from backup. In addition, the system's root file system and swap area are not located on volumes any longer. To correct these problems, follow the instructions in Section 14.11.6.

14.11.6 Configuration Cleanup

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

14.11.6.1 Rootability Cleanup

To clean up the LSM configuration, remove any volumes associated with rootability, and their associated disks. This must be done if the root disk was under LSM control prior to installation. The volumes to remove are:

- `rootvol`, which contains the root file system
- `swapvol`, which contains the swap area

Follow these steps:

1. To begin the cleanup, remove the root volume, stop the volume, and then use the `voledit` command, as follows:

   ```
   # volume stop rootvol
   # voledit -r rm rootvol
   ```

2. Repeat the command, using `swapvol` in place of `rootvol`, to remove the swap volume.
3. Remove the LSM disks used by `rootvol` and `swapvol`.

   For example, if disk `rz3` was associated with `rootvol` and disk `rz3b` was associated with `swapvol`, you would enter the following commands:

   ```
   # voldg rmdisk rz3 rz3b
   # voldisk rm rz3 rz3b
   ```

14.11.6.2 LSM Volumes for /usr and /var

If `/usr` and `/var` were on LSM volumes prior to the reinstallation, clean up the volumes using the `voledit` command similar to the previous example shown for `rootvol`. Remove the LSM disks associated with the volumes used for `/usr` and `/var`.

14.11.6.3 Volume Cleanup

After completing the rootability cleanup, you must determine which volumes need to be restored from backup. The volumes to be restored include any volumes that had all plexes residing on disks that were removed or reinstalled. These volumes are invalid and must be removed, recreated, and restored from backup. If only some plexes or a volume exist on reinitialized or removed disks, these plexes must be removed. The plexes can be readded later.

To restore the volumes, do the following:

1. Establish which LSM disks have been removed or reinstalled, by entering the command:

   ```
   # voldisk list
   ```

   LSM 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 `voldisk list` command produces an output similar to this:

```
DEVICE  TYPE      DISK       GROUP      STATUS
rz0     sliced    -          -          error
rz1     sliced    disk02     rootdg     online
rz2     sliced    disk03     rootdg     online
-       -         disk01     rootdg     failed was:  rz0
```

The previous display shows that the reinstalled root device, `rz0` is not recognized as an LSM disk and is marked with a status of `error`. `disk02` and `disk03` were not involved in the reinstallation and are recognized by LSM and associated with their devices (`rz1` and `rz2`). The former `disk01`, the LSM disk that had been associated with the replaced disk device, is no longer associated with the device (`rz0`).

If there had been other disks (with volumes or volume plexes on them) removed or replaced during reinstallation, these disks would also have a disk device in `error` state and an LSM disk listed as not associated with a device.

2. Once you know which disks have been removed or replaced, all the plexes on disks with a status of `failed` must be located. Enter the command:

   ```
   # volprint -sF "%vname" -e 'sd_disk = "<disk>""'
   ```

   In this command, the variable *<disk>* is the name of a disk with a `failed` status.

   Note

   Be sure to enclose the disk name in quotes in the command. Otherwise, the command will return an error message.

   The `volprint` command returns a list of volumes that have plexes on the failed disk. Repeat this command for every disk with a `failed` status.

3. Check the status of each volume. To print volume information, enter:

   ```
   # volprint -th <volume_name>
   ```

   In this command, *volume_name* is the name of the volume to be examined.

   The `volprint` command displays the status of the volume, its plexes, and the portions of disks that make up those plexes. For example, a volume named `fnah` with only one plex resides on the reinstalled disk named `disk01`. The `volprint -th` command, applied to the volume fnah, produces the following display:

```
V  NAME      USETYPE  KSTATE   STATE    LENGTH READPOL  PREFPLEX
PL NAME      VOLUME   KSTATE   STATE    LENGTH LAYOUT ST-WIDTH MODE
SD NAME      PLEX     PLOFFS   DISKOFFS LENGTH DISK-MEDIA   ACCESS

v  fnah      fsgen     DISABLED ACTIVE   24000  SELECT   -
pl fnah-01   fnah      DISABLED NODEVICE 24000  CONCAT   -
sd disk01-06 fnah-01  0        519940   24000  disk01   -
```

4. The only plex of the volume is shown in the line beginning with pl. The STATE field for the plex named fnah-01 is NODEVICE. The plex has space on a disk that has been replaced, removed, or reinstalled. Therefore, the plex is no longer valid and must be removed. Since fnah-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, you will need it for the backup procedure.

   To remove the volume, use the voledit command. To remove fnah, enter the command:

   ```
   # voledit -r rm fnah
   ```

   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 woof has one striped plex, striped across three disks, one of which is the reinstalled disk disk01. The output of the volprint -th command for woof returns:

```
V  NAME     USETYPE KSTATE   STATE    LENGTH  READPOL  PREFPLEX
PL NAME     VOLUME  KSTATE   STATE    LENGTH  LAYOUT   ST-WIDTH MODE
SD NAME     PLEX    PLOFFS   DISKOFFS LENGTH  DISK-MEDIA   ACCESS

v  woof      fsgen   DISABLED ACTIVE   4224    SELECT   -
pl woof-01   woof    DISABLED NODEVICE 4224    STRIPE   128      RW
sd disk02-02 woof-01 0        14336    1408    disk02         rz1
sd disk01-05 woof-01 1408     517632   1408    disk01       -
sd disk03-01 woof-01 2816     14336    1408    disk03         rz2
```

   The display shows three disks, across which the plex woof-01 is striped (the lines starting with sd represent the stripes). The second stripe area is located on LSM disk01. This disk is no longer valid, so the plex named woof-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 woof exists on the backup media, it can be restored later.

Note

Keep a record of the volume name and length of any volumes you intend to restore from backup.

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

A volume that has one plex on a failed disk may also have other plexes 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 `volprint -th` command for a volume with one plex on a failed disk (`disk01`) and another plex on a valid disk (`disk02`) would look like this:

```
V  NAME    USETYPE  KSTATE   STATE    LENGTH  READPOL  PREFPLEX
PL NAME    VOLUME   KSTATE   STATE    LENGTH  LAYOUT ST-WIDTH MODE
SD NAME    PLEX     PLOFFS   DISKOFFS LENGTH  DISK-MEDIA   ACCESS

v  foo        fsgen    DISABLED ACTIVE   10240   SELECT   -
pl foo-01     foo      DISABLED ACTIVE   10240   CONCAT   -   RW
sd disk02-01 foo-01   0        0        10240   disk02   rz1
pl foo-02     foo      DISABLED NODEVICE 10240   CONCAT       RW
sd disk01-04 foo-02   0        507394   10240   disk01   -
```

This volume has two plexes, `foo-01` and `foo-02`. The first plex, `foo-01`, does not use any space on the invalid disk, so it can still be used. The second plex, `foo-02`, uses space on the invalid disk, `disk01`, and has a state of `NODEVICE`. Mirror `foo-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 if you want to create another plex later.

To remove an invalid plex, the plex must be dissociated from the volume and then removed. This is done with the `volplex` command. To remove the plex `foo-02`, enter the following command:

```
# volplex -o rm dis foo-02
```

5. Once all the volumes have been cleaned up, you must clean up the disk configuration as described in the following section.

14.11.6.4 Disk Cleanup

Once all invalid volumes and volume 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 `voldisk list` command) must be removed from the configuration.

To remove the disk, use the `voldg` command. To remove the failed disk01, enter:

```
# voldg rmdisk disk01
```

If the voldg command returns an error message, some invalid volume plexes exist. Repeat the processes described in ‘‘Volume Cleanup’’ until all invalid volumes and volume plexes are removed.

14.11.6.5 Rootability Reconfiguration

Once all the invalid disks have been removed, the replacement or reinstalled disks can be added to LSM control. If the root disk was originally under LSM control (the root file system and the swap area were on volumes), or you now want to put the root disk under LSM control, add this disk first.

To add the root disk to LSM control, enter the following command:

```
# /usr/sbin/volencap <boot_disk>
```

For more information see Chapter 5.

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

14.11.6.6 Final Reconfiguration

Once the root disk is encapsulated, any other disks that were replaced should be added using voldiskadm. If the disks were reinstalled during the operating system reinstallation, they should be encapsulated; otherwise, simply add them. See Chapter 6.

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 on their contents restored from backup. The volume recreation can be done using either volassist or the Logical Storage Visual Administrator (dxlsm) interface.

To recreate the volumes fnah and woof using the volassist command, enter:

```
# volassist make fnah 24000
# volassist make woof 4224 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 plexes recreated following the instructions for mirroring a volume for the interface (volassist, voldiskadm, or dxlsm) you choose.

To replace the plex removed from the volume foo using volassist, enter:

```
# volassist mirror foo
```

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

LSM Performance Management 15

This chapter suggests performance priorities and guidelines for use with LSM. It also provides information about monitoring LSM and gathering performance data.

15.1 Performance Strategies

Achieving optimal performance by balancing input/output (I/O) load among several disks on a system without LSM may be limited because it is difficult to anticipate future disk usage patterns, and it is not always possible 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.

LSM provides flexibility in configuring storage to improve system performance. Table 15-1 describes two basic strategies available to optimize performance.

Table 15-1: Strategies for Improved Performance

Strategy	Result
Assign data to physical drives to evenly balance the I/O load among the available disk drives	Achieves a finer level of granularity in data placement because LSM provides a way for volumes to be split across multiple drives. After measuring actual data-access patterns, you 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.

Table 15-1: (continued)

Strategy	Result
Identify the most-frequently accessed data and increase access bandwidth to that data through the use of mirroring and striping	Achieves a significant improvement in performance when there are multiple I/O streams. If you can identify the most heavily-accessed file systems and databases, then you can realize significant performance benefits by striping the high traffic data across portions of multiple disks, and thereby increasing access bandwidth to this data. Mirroring heavily-accessed data not only protects the data from loss due to disk failure, but in many cases also improves I/O performance.

15.1.1 Improving Mirrored Disk Performance

The use of mirroring to store multiple copies of data on a system improves the chance of data recovery in the event of a system crash or disk failure, and in some cases can be used to improve system performance. However, mirroring degrades write performance slightly. On most systems, data access patterns conform to the 80/20 concept: Twenty percent of the data is accessed 80 percent of the time, and the other 80 percent of the data is accessed 20 percent of the time.

The following sections describe some guidelines for configuring mirrored disks, improving mirrored system performance, and using block-change logging to speed up the recovery of mirrored volumes.

15.1.1.1 Configuring Mirrored Disks for Performance

When properly applied, mirroring can provide continuous data availability by protecting against data loss due to physical media failure. Use the following guidelines when using mirroring:

- Never place subdisks from different mirrors of a mirrored volume on the same physical disk; this action compromises the availability benefits of mirroring and significantly impacts performance.
- To provide optimum performance improvements through the use of mirroring, at least 30 percent of the physical I/O operations should be reads; a higher percentage of read operations results in a higher benefit of performance. Mirroring may provide no performance increase or result in a decrease of performance in a write-intensive workload environment.

Note

> The Digital UNIX operating system implements a file system cache. Because read requests frequently can be satisfied from this cache, the read/write ratio for physical I/O's through the file system can be significantly more biased toward writing than the read/write ratio at the application level.

- Where feasible, use disks attached to different controllers when mirroring or striping. Although most disk controllers support overlapped seeks (allowing seeks to begin on two disks at once), do not configure two mirrors of the same volume on disks attached to a controller that does not support overlapped seeks. This is very important for older controllers or SCSI disks that do not do caching on the drive. It is less important for many newer SCSI disks and controllers.
- If one plex exhibits superior performance — either because the disk is being striped or concatenated across multiple disks, or because it is located on a much faster device — then the read policy can be set to the preferred read policy (described in Table 15-2) for the faster plex. By default, a volume with one striped plex should be configured with preferred read of the striped plex.

15.1.1.2 Using Mirroring to Improve System Performance

Mirroring can also improve system 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 70 percent writes), then mirroring can result in somewhat reduced performance.

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

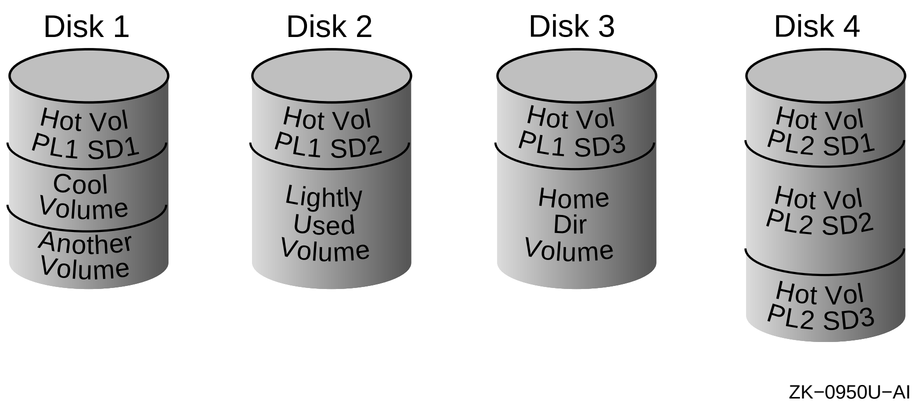
To provide optimal performance for different types of mirrored volumes, LSM supports the read policies shown in Table 15-2.

Table 15-2: LSM Read Policies

Policy	Description
Round-robin read	Satisfies read requests to the volume in a round-robin manner from all plexes in the volume
Preferred read	Satisfies read requests from one specific plex (presumably the plex with the highest performance)

For example, in the configuration shown in Figure 15-1, the read policy of the volumes labeled Hot Vol should be set to the preferred read policy from the striped mirror labeled PL1. In this way, reads going to PL1 distribute the load across a number of otherwise lightly used disk drives, as opposed to a single disk drive.

Figure 15-1: Improving System Performance Using Mirroring and Striping



To improve performance for read-intensive workloads, up to eight plexes can be attached to the same volume, although this scenario results in a decrease of effective disk space use. 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.

15.1.1.3 Improving Mirrored-Volume Recovery with Block-Change Logging

LSM *block-change logging* keeps track of the blocks that have changed as a result of writes to a mirror. Block-change logging does this by identifying the block number of changed blocks, and storing this number in a *log subdisk*.

Block-change logging can significantly speed up recovery of mirrored volumes following a system crash.

Note

> Using block-change logging can significantly decrease system performance in a write-intensive environment.

Logging subdisks are one-block long subdisks that are defined for and added to a mirror that is to become part of a volume that has block-change logging enabled. They are ignored as far as the usual mirror policies are concerned and are only used to hold the *block-change log*.

Follow these guidelines when using block-change logging:

- Make sure that the subdisk that will be used as the log subdisk does not contain necessary data.
- Ensure that the logging subdisks are one block in length.
- If possible, do not place the log subdisk on a heavily-used disk.
- If persistent (nonvolatile) RAM disks are available, use them for log subdisks.
- Make sure that all mirrors within the volume have a block-change log. If only one plex has a block-change log, logging will be disabled for the volume.

15.1.2 Improving Striped Disk Performance

Striping can improve serial access when I/O exactly fits across all subdisks in one stripe. Better throughput is achieved because parallel I/O streams can operate concurrently on separate devices.

The following sections describe how to use striping as a way of slicing data and storing it across multiple devices to improve access bandwidth for a mirror.

15.1.2.1 Configuring Striped Disks for Performance

Follow these guidelines when using striping:

- Calculate stripe sizes carefully. If it is not feasible to set the stripe width to the track size, use 64 kilobytes for the stripe width, which is the default.
- Avoid small stripe widths; small stripe widths can result in poor system performance *unless* its total width exactly matches the size of the I/O.
- Never put more than one subdisk of a striped mirror on the same physical disk.

- 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 (MTBF) 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 mirror of a mirrored volume is striped, be sure to set the policy of the volume to `preferred read` for the striped mirror. (The default read policy, `select`, does this automatically.)
- When striping is used with mirroring, never place subdisks from one mirror on the same physical disk as subdisks from the other mirror.
- If more than one mirror of a mirrored volume is striped, make sure the stripe width is the same for each striped mirror.
- 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.

The `volassist` command automatically adopts many of these rules when it allocates space for striped plexes in a volume.

15.1.2.2 Improving Access Bandwidth with Striped Plexes

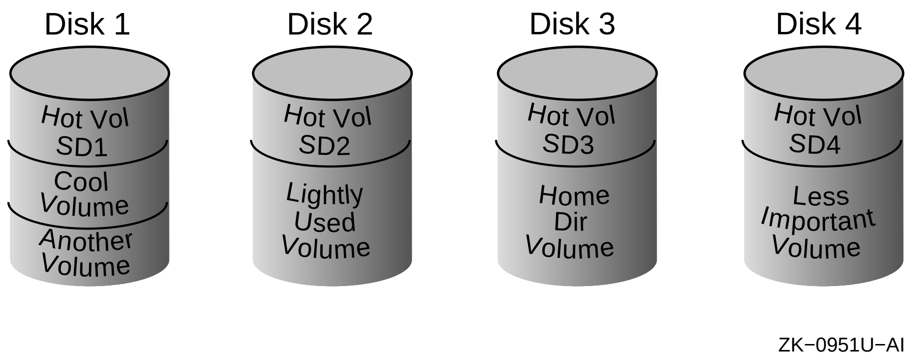
Striping can provide increased access bandwidth for a plex. Striped plexes exhibit improved access performance for both read and write operations. Where possible, disks attached to different controllers should be used to further increase parallelism.

One disadvantage of striping is that some configuration changes are harder to perform on striped plexes than on concatenated plexes. For example, it is not possible to move an individual subdisk of a striped plex, or to extend the size of a striped plex, except by creating a completely new plex and removing the old striped plex. This can be done with the `volassist move` command or the `volplex mv` command.

While these operations can be performed on concatenated plexes without copying through a plex, striping offers the advantage that load balancing can be achieved in a much simpler manner.

Figure 15-2 is an example of a single file system that has been identified as a data-access bottleneck. This file system was striped across four disks, leaving the remainder of those four disks free for use by less-heavily used file systems.

Figure 15-2: Use of Striping for Optimal Data Access



15.1.2.3 Striped Plex Configuration Changes

Simple performance experiments can be run to determine the appropriate configuration for a striped plex. The configuration changes can be done while the data is online. For example, the stripe width or nstripe of a striped plex can be modified to determine the optimal values. Similarly, data can be moved from a "hot" disk to a "cold" disk.

The following example gives the steps required to change the stripe width of a plex `P1` from 64 kilobytes (the default) to 32 kilobytes.

1. Create a 32 kilobyte stripe width plex `pl2`.

```
# volmake sd sd3 rz10,0,102400
# volmake sd sd4 rz11,0,102400
# volmake plex pl2 layout=stripe stwidth=32 sd=sd3,sd4
```

2. Use the `volplex` command to replace the original plex `pl1` with the new plex `pl2`.

```
# volplex mv pl1 pl2
```

   This command will take some time to complete.

15.1.2.4 Improving Performance Under AdvFS

When adding LSM volumes to AdvFS domains, Digital recommends the addition of multiple simple volumes rather than a single, large, striped or concatenated volume. This type of configuration enables AdvFS to take advantage of multiple storage containers in an AdvFS domain by sorting and balancing the I/O across all the storage containers in the domain.

15.2 Monitoring LSM Performance

The following sections suggest ways to prioritize your performance requirements, and how to obtain and use performance data and recorded statistics to help you gain the performance benefits provided by LSM.

Table 15-3 describes the two sets of performance priorities for a system administrator.

Table 15-3: LSM Performance Priorities

Priority	Description
Physical (hardware)	Addresses 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, it may be necessary to move subdisks around to balance the disks.
Logical (software)	Involves software operations and how they are managed. Based on monitoring, certain volumes may be *mirrored* (multiple plexes) or *striped* to improve their performance. Overall throughput may be sacrificed to improve the performance of critical volumes. Only you can decide what is important on a system and what tradeoffs make sense.

15.2.1 Statistics Recorded by LSM

LSM records the following three I/O statistics:

- A count of operations
- The number of blocks transferred (one operation could involve more than one block)
- The total active time

LSM records these statistics for logical I/Os for each volume. The statistics are recorded for the following types of operations: reads, writes, atomic copies, verified reads, verified writes, mirror reads, and mirror writes.

For example, one write to a two-mirror volume will result in the following statistics being updated:

- One operation for each plex
- One operation for each subdisk
- One operation for the volume

Similarly, one read that spans two subdisks results in the following statistics being updated: — one read for each subdisk, one for the mirror, and one for the volume.

LSM also maintains other statistical data. For example, read and write failures that appear for each mirror, and corrected read and write failures for each volume accompany the read and write failures that are recorded.

15.2.2 Gathering Performance Data

LSM provides two types of performance information — I/O statistics and I/O traces:

- I/O statistics are retrieved using the `volstat` utility
- I/O tracing can be retrieved using the `voltrace` utility

Each type of performance information can help in performance monitoring. The following sections briefly discuss these utilities.

15.2.2.1 Obtaining I/O Statistics

The `volstat` utility provides access to information for activity on volumes, plexes, subdisks, and disks under LSM control. The `volstat` utility reports statistics that reflect the activity levels of LSM objects since boot time. Statistics for a specific LSM 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 you do not specify a particular disk group on the `volstat` command line, statistics for the default disk group (`rootdg`) are displayed.

The amount of information displayed depends on what options are specified to `volstat`. For detailed information on available options, refer to the `volstat`(8) reference page.

The `volstat` utility is also capable of resetting the statistics information to zero. This can be done for all objects or for only those objects that are specified. Resetting just prior to a particular operation makes it possible to measure the impact of that particular operation afterwards.

The following example shows typical output from a `volstat` display:

```
                  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
```

15.2.2.2 Tracing I/O Operations

The `voltrace` command is used to trace operations on volumes. Through the `voltrace` utility, you can set I/O tracing masks against a group of volumes or to the system as a whole. You can then use the `voltrace` utility to display ongoing I/O operations relative to the masks.

The trace records for each physical I/O show a volume and buffer-pointer combination that enables you to track each operation even though the traces may be interspersed with other operations. Like the I/O statistics for a volume, the I/O trace statistics include records for each physical I/O done, and a logical record that summarizes all physical records. For additional information, refer to the `voltrace`(8) reference page.

15.2.3 Using Performance Data

Once performance data has been gathered, you can use the data to determine an optimum system configuration that makes the most efficient use of system resources. The following sections provide an overview of how you can use I/O statistics and I/O tracing.

15.2.3.1 Using I/O Statistics

Examination of the I/O statistics may suggest reconfiguration. There are two primary statistics to look at: volume I/O activity and disk I/O activity. The following steps describes how to record and examine I/O statistics:

1. Before obtaining statistics, consider clearing (resetting) all existing statistics. Clearing statistics eliminates any differences between volumes or disks that might appear due to volumes being created, and also removes statistics from booting (which are not normally of interest). Use the following command to clear all statistics:

   ```
   # volstat -r
   ```

2. After clearing the statistics, gather I/O statistics during a time span when typical system activity is occurring. You may also want to gather I/O statistics during the time that you run a specific application to obtain statistics specific to that application or workload.

   When monitoring a system that is used for multiple purposes, try not to exercise any application more than it would be exercised under typical circumstances. When monitoring a time-sharing system with many users, try to let the I/O statistics accumulate during typical usage for several hours during the day.

3. To display volume statistics, enter the `volstat` command without any

arguments. The output might appear as shown in the following example:

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

4. To display disk statistics, enter the `volstat -d` command. The resulting output might appear as shown in the following example:

```
                  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
```

5. Check the displays for volumes with an unusually large number of operations or excessive read or write times.

6. To move the volume `archive` onto the boot disk (`disk01` in the previous example), identify which disks it is on using the `volprint -tvh archive` command. The display from this command is similar to the following example:

```
V  NAME       USETYPE    KSTATE  STATE    LENGTH READPOL...
PL NAME       VOLUME     KSTATE  STATE    LENGTH LAYOUT...
SD NAME       PLEX       PLOFFS  DISKOFFS LENGTH DISK-MEDIA...

v  archive    fsgen      ENABLED ACTIVE   204800 SELECT   -
pl archive-01 archive    ENABLED ACTIVE   204800 CONCAT   -  RW
sd disk03-03  archive-01 0       409600   204800 disk03     rz2
```

Looking at the associated subdisks indicates that the `archive` volume is on disk `disk03`. To move the volume off `disk03` and onto `disk01`, use one of the following commands.

K-shell users, enter:

```
# volassist move archive !disk03 disk01
```

C-shell users, enter:

```
# volassist move archive \!disk03 disk01
```

These commands indicate that the volume should be reorganized so that no part is on `disk03`, and that any parts to be moved should be moved to `disk01`.

Note

The easiest way to move pieces of volumes between disks is to use the Logical Storage Manager Visual Administrator (`dxlsm`). If `dxlsm` is available on the system, you may prefer to use it instead of the command-line utilities.

7. If there are two busy volumes, try to move them so that each volume is on a different disk.
8. If there is one volume that is particularly busy (especially if it has unusually large average read or write times), consider striping the volume (or splitting the volume into multiple pieces, with each piece on a different disk). Converting a volume to use striping requires sufficient free space to store an extra copy of the volume.

   To convert to striping, create a striped mirror of the volume and then remove the old mirror. For example, to stripe the volume `archive` across disks `disk02` and `disk04`, enter the following commands:

   ```
   # volassist mirror archive layout=stripe disk02 disk04
   # volplex -o rm dis archive-01
   ```

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

If...	Then...
Some disks appear to be used excessively (or have particularly long read or write times)	Reconfigure some volumes.
There are two relatively busy volumes on a disk	Consider moving the volumes closer together to reduce seek times on the disk.
There are too many relatively busy volumes on one disk	Try to move the volumes to a disk that is less busy.

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

Note

File systems and databases typically shift their use of allocated space over time, so this position-specific information on a volume often is 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 disks that are not busy.

11. Examine the ratio of reads and writes to identify volumes that can be mirrored to improve their performance.

If...	Then...
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.
A particularly busy volume has a ratio of reads to writes as high as 5:1	It is likely that mirroring can dramatically improve performance of that volume.

Note

By using LSM mirroring, you can substantially reduce the risk that a single disk failure will result in the failure of a large number of volumes. This is because striping a volume 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 were on a separate disk, only one volume would have to be restored.

15.2.3.2 Using I/O Tracing

Whereas I/O statistics provide the data for basic performance analysis, I/O traces provide for more detailed analysis. With an I/O trace, the focus of the analysis is narrowed to obtain an event trace for a specific workload. For example, you can identify exactly where a hot spot is, how big it is, and which application is causing it.

By using data from I/O traces, you can simulate real workloads on disks and trace the results. By using these statistics, you can anticipate system limitations and plan for additional resources.

Appendixes: Visual Administrator Reference, Error Messages, and Examples

Visual Administrator Reference A

Various Visual Administrator operations and options can be accessed using the main menus located in the menu bars of the Visual Administrator root window, view windows, or both. Some of these menus lead to submenus; some lead to forms; and some simply execute operations directly.

A.1 Introduction and Setup

The following sections provide reference information for the introductory and setup menus.

A.1.1 Views

The LSM Visual Administrator root window contains a Views menu that is used to create and modify view windows. The following list describes the views available using the Views menu (located in the LSM Visual Administrator root window only).

- Create a View

  Views → Create a View

  `Description:` Creates a new user-specified view (commonly referred to as a *user-created view* or a *user view*). The user specifies a unique name for this new view. Once created, icons can be added to this new view window by copying them over from existing views.

  User-created views differ from default views (disk, volume, disk group, and world views) in that they contain copies of icons from the default views. Any operations performed from user-created views are automatically activated in the corresponding default views. Icons that appear in default views are not necessarily duplicated in user-created views.

- Rename a View

  Views → Rename a View

  `Description:` Renames an existing user-created view. The user specifies a new, unique name for the view.

  `Requirement:` Only a user-created view can be renamed.

- Remove Views

  Views → Remove Views

  `Description:` Permanently removes a user-created view. The removal of a user-created view results in the removal of its icons, but the objects represented by those icons are unaffected.

  `Requirement:` Only a user-created view can be removed.

A.1.2 Views Forms

The following list shows the forms accessed via the Views menu (located in the LSM Visual Administrator root window only).

- Create a View

  Views → Create a View

  `Description:` Creates a new view window. Once created, this view is represented by a new view button in the Visual Administrator root window. This user-created view is also retained across future Visual Administrator sessions. The following table describes the field for this form.

Field	Description
View Name:	The name of the view to be created. This name must be unique. The maximum length of the name is 14 characters. This field is required.

  `Requirement:` The name specified for the new view must be unique.

- Rename a View

  Views → Rename a View

  `Description:` This form is used to change the name of an existing user-created view. The following table describes the fields for this form.

Field	Description
Old View Name:	The name of the view to be renamed. This name must belong to a user-created view. This field is required.
New View Name:	The name to which the existing name is to be changed. This name must be unique. The maximum length of the name is 14 characters. This field is required.

`Requirements:` Only user-created views can be renamed, and the new view name must be unique.

Views → Remove a View

`Description:` This form is used to remove a user-created view. This form contains no fields. Instead, it lists all user-created views that currently exist.

The user selects one or more views for removal by highlighting the desired view names with the MB1 button. If a view is mistakenly selected, the MB1 button can be used again to clear that view name. Removal is activated by clicking MB1 on the Apply button once, then confirming the removal by selecting Apply again (this ensures that a view is not accidentally removed).

A.1.3 Options

User preferences for how the LSM Visual Administrator should operate are set via the Options menu.

Once set, most preferences for each user are saved to the `.dxlsm_pref`, file located in each user's home directory. The `.dxlsm_pref` file should not be edited by users. Once set, these preferences are maintained across future Visual Administrator session.

A.1.3.1 Options Menu

The following list shows the menu selections accessible from the Options menu:

- Show Command

  Options → Show Command

  `Description:` This menu selection is used to specify whether the Command Info Window is to be displayed before each command is executed or only upon command failure.

  From the Show Command menu, a submenu allows the user to indicate when to display the Command Info Window. The following table describes the available options.

Option	Description
Show on Error	Display the Command Info Window only when an LSM or system command has failed. This is the default behavior.
Show at Start	Display the Command Info Window whenever an LSM or system command is ready to be sent. This enables the user to view the actual commands being executed via the Visual Administrator.

- When Commands Are Ready

  Options → When Commands Are Ready

  `Description:` This menu selection is used to specify whether commands should be executed immediately or simply displayed for user review.

  From the When Commands Are Ready menu, a submenu allows the user to indicate what the Visual Administrator is to do when a command is issued:

  Execute Commands
  : Automatically execute commands as soon as they are issued. This is the default behavior.

  Show Commands Only
  : Display commands in the Command Info Window for user review rather than executing them. Upon approval, the user can execute the displayed command directly from the Command Info Window by highlighting that command and then using the Execute menu.

- Logging

  Options → Logging

  `Description:` This menu selection is used to start or stop logging of Visual Administrator commands. Logging records all commands sent to LSM or the system by the LSM Visual Administrator in a specified file.

  From the Logging menu, a submenu allows you to indicate whether logging should be activated or deactivated:

Option	Description
Start	Begin recording all commands to a log file. A log file is created if one does not already exist. The file to be used for logging must be specified in the resulting form. If the file exists, the user must have permission to write to that file. The log information will be appended to the end of the specified file.
Stop	Stop recording all commands to the log file. When logging is discontinued, the user is responsible for remembering the name of the log file that was used.

Note

Unlike other user preferences, the logging setting is not saved across Visual Administrator sessions.

`Form:` Log File Form (described in Section A.1.3.2).

- Popup the Command Window

  Options → Popup the Command Window

  `Description:` This menu selection is used to access and display the Command Info Window. This window displays current and previous commands, along with the status of each command. Once accessed in this way, the Command Info Window remains visible until it is closed via its File menu.

  See Section A.1.4 for further details.

- Format of Size

  Options → Format of Size

  `Description:` This menu selection is used to specify the units (megabytes, kilobytes, or sectors) to be used for size-related output. The unit of size is set to megabytes until the user resets it.

  In properties forms, length values are displayed as sectors (s), kilobytes (k), or megabytes (m). If the size cannot be cleanly converted into kilobytes or megabytes, it is displayed in sectors instead (even though another format of size preference may be set).

  The preferred format of size applies to output only and does not impact input in any way. Input typically defaults to sectors, unless megabytes or kilobytes are specified.

  From the Format of Size menu, a submenu allows the user to select the units of size:

Units	Description
Mbytes	Use megabytes when displaying size-related output.
Kbytes	Use kilobytes when displaying size-related output.
Sectors	Use sectors when displaying size-related output. This is the default.

A.1.3.2 Options Forms

The following forms are accessible via the Options menu:

- Start Logging

  Options → Logging → Start

  `Description:` This form is used to specify the file to be used for logging purposes. The following table describes the field for this form.

Field	Description
Log File:	The name of the file (and path name) to be used to store the command log. If no path is specified, the file is created in the directory from which the Visual Administrator session was started. The maximum length of the path specified here is 127 characters.

`Requirements:`

  – The user must have privileges appropriate to access and write to the named file (and any directories in its path).

  – If a path name is included, it must be valid.

A.1.4 Command Info Window

You access the Command Info Window via the Options menu. Once accessed, this window displays information on current and previous commands executed by the Visual Administrator.

In addition to viewing or previewing commands, you can use the Command Info Window to execute commands directly through its Execute menu. This is useful for reexecuting commands or for executing commands that were previously only shown in this window. The following list describes using the Execute and Execute With Force options:

- Execute

Execute → Execute

`Description:` This menu selection is used to execute the command highlighted in the Command History section of the window. The Visual Administrator sends the highlighted command to LSM or the system for execution.

`Requirement:` A single command in the Command History section of the window must be highlighted.

- Execute With Force

  Execute → Execute With Force Option

  `Description:` This menu selection is used to forcefully execute the LSM command highlighted in the Command History section of the window (using the `-f` option). This option effectively forces LSM to complete an operation that is considered unsafe and should therefore be used only when the user is certain that an operation should be performed in this way.

  Note

  This is a dangerous operation that can cause irreparable loss of data. Use this option only when absolutely necessary.

  `Requirements:`

  – The force option works for some LSM commands only (those that take the `-f` option). The `-f` option does not apply to file system operations offered through the Visual Administrator.

  – Use this option only when you are sure that an operation will succeed, even though LSM error checking prevents it.

A.1.5 Icons

You can access a set of icon-related options via the Icon menu located in views.

The following list shows the menu selections you can access via the Icon menu:

- Maximize Icons

  Icon → Maximize Icons

  `Description:` This operation maximizes the selected minimized icons, making it show all of its subicons.

  `Requirement:` At least one minimized icon must be selected.

- Minimize Icons

  Icon → Minimize Icons

  `Description:` This operation minimizes the selected icons, making it shrink down in size and hide all of its subicons.

  Minimized icons occupy less space and are displayed with their names in reverse type.

  `Requirement:` You must select at least one maximized icon.

- Maximize All Icons

  Icon → Maximize All Icons

  `Description:` This operation maximizes all icons in the current view window, making them show all of their subicons. No icons need to be selected.

- Create Icons

  Icon → Create Icons

  `Description:` This operation creates a copy of the icons selected from another view and places the new copy in the current user-created view. Icons that already exist in this user-created view will not be duplicated.

  `Requirement:` This option is only available in user-created views.

- Remove Icons

  Icon → Remove Icons

  `Description:` This operation removes the selected icons from the current user-created view.

  `Requirement:` This option is only available in user-created views.

A.1.6 Help

The LSM Visual Administrator provides an extensive Help facility, which is accessible from most windows or operations.

A.1.6.1 Help from Menus

Almost all Visual Administrator windows contain a Help menu that provides access to information relevant to the contents of that window. The Help menu contains a list of available Help options, each of which brings up a Help window containing information on the selected topic.

Most Visual Administrator menus contain a Help option (listed at the bottom of the menu). When the Help option is selected from a given menu, a Help window containing information relevant to the operations or options listed in the menu appears.

A.1.6.2 Help from Forms

Visual Administrator forms contain Help buttons that provide information relevant to the form and its fields. Help is accessed by clicking MB1 on a form’s Help button.

A.1.6.3 Navigating Help Windows

Regardless of where you try to access Help, LSM displays a Help window that includes text relevant to the topic.

From a given Help window, the user can access different Help topics in any of the following ways:

- Click on any Help topic listed in the current Help screen’s see ALSO section (located at the bottom of the Help window). The existing Help text is immediately replaced with that of the selected new topic.
- Click on Previous in the Help window’s menu bar. The existing Help text is immediately replaced with that of the Help topic that was previously being viewed.
- Click on Next in the Help window’s menu bar. The existing Help text is immediately replaced with that of the Help topic that was being viewed before skipping backwards to a previous topic.

A.1.6.4 Help Index

A Help index can be accessed via the Help menu located in the menu bar of the Help window itself. The Help index lists all available help topics for the Visual Administrator. Once identified, any of the listed Help topics can be accessed by clicking MB1 on the matching topic listed in the SEE ALSO section of the Help window (directly below the Help index listing).

A.1.7 Exiting the Visual Administrator and its Windows

Most Visual Administrator windows contain a File menu, which is used to exit that particular Visual Administrator window or to exit the session completely.

The following list describes the menu selections you can access via the File menu:

- Exit

  File → Exit

  `Description` This menu selection both closes the current window and exits the Visual Administrator completely. Although the session is closed down, user preferences set during the session are retained. Because the Visual Administrator operations are applied to the Visual Administrator configuration as they are issued, quitting a session has no effect on the configuration and does not undo any changes made through the Visual Administrator.

- Close File

  File → Close

  `Description:` This menu selection closes the current window only.

A.2 File System Operations

This section provides information on menus and forms relating to UFS operations. You access UFS operations via the Basic-Ops menu. This menu is located in view windows, such as View of rootdg. This menu provides access to UFS operations involving general file system maintenance, and is accessed as shown:

Basic-Ops → UFS Operations

You can access the following menu selections via the Basic-Ops menu.

- Create
- Make File System
- Mount
- Unmount
- Check File System (fsck)
- Display Properties
- Help

The Help selection accesses a Help window which displays information relevant to the available file system operations.

A.2.1 File System Menus

The following list describes the file system operations menu items:

- Create

  Basic-Ops → UFS Operations → Create

  `Description:` This operation creates a file system on an underlying volume. This is done by creating a volume on one or more disks and then creating the file system on that volume.

  You can select one or more disks on which to create the volume (providing that there is sufficient space on the disks). If you do not specify any disks, LSM automatically determines which disks to use based on available free space.

  From the Create menu, select the type of volume to be created from a submenu listing two of the basic types of volumes:

Type	Description
Simple	Creates a simple, concatenated volume whose subdisks are arranged both sequentially and contiguously within a plex.
Striped	Creates a volume with data spread fairly evenly across multiple disks by way of striping. *Stripes* are relatively small, equally-sized fragments that are allocated alternately to the subdisks of each plex.

  If a mirrored volume is desired, a simple or striped volume must be created and then mirrored using the Add Mirror option from the Volume Operations menu.

  `Requirements:`

  – Only disks in the same disk group can be selected.

  – Only LSM disks (disks under LSM control) can be selected.

  – If striping is to be in effect, at least two disks are required in order for the operation to succeed.

  `Forms:` Simple Volume/FS Create Form and Striped Volume/FS Create Form (described in Section A.2.2).

- Make File System

  Basic-Ops → UFS Operations → Make File System

  `Description:` This operation is used to make a file system on an existing volume. The user selects the volume on which to place the new file system, and specifies the mount point if the file system is to be mounted immediately.

  `Requirements:`

– A volume icon must be selected.

– The selected volume must be enabled.

– Only one mounted file system can exist on each volume.

`Form:` Make File System Form (described in Section A.2.2).

- Mount

  Basic-Ops → UFS Operations → Mount

  `Description:` This operation mounts the file system that resides on the selected volume. This operation assumes that the selected volume already contains a valid file system. The Visual Administrator has no way of knowing whether a valid, unmounted file system already exists on a given volume. You must make sure of the existence of an unmounted file system on a volume, as well as that file system's type.

  `Requirements:`

  – A volume icon must be selected.

  – A valid, unmounted file system must already exist on the selected volume.

  `Form:` Mount File System Form (described in Section A.2.2).

- Unmount

  Basic-Ops → UFS Operations → Unmount

  `Description:` This operation is used to unmount the file systems that resides on the selected volumes. The file system can be unmounted only if the mount point is not busy.

  `Requirements:`

  – At least one volume icon must be selected.

  – The selected volume must contain a mounted file system.

- Check File System

  Basic-Ops → UFS Operations → Check File System (fsck)

  `Description:` This operation checks the file systems on the selected volumes for consistency (using `fsck`). The file system to be checked must currently be unmounted.

  `Requirements:`

  – At least one volume icon must be selected.

  – The selected volumes must contain an unmounted file system.

  `Form:` File System Check Form (described in Section A.2.2).

- Display Properties

  Basic-Ops → UFS Operations → Display Properties

  `Description:` Display information for file systems mounted on the system. The user may select the file system for which information is to be displayed from a list of all mounted file systems. If a volume is selected, the properties for the file system that resides on that volume is displayed by default.

A.2.2 File System Forms

Some file system operations result in the appearance of forms that you must complete in order for that operation to proceed. Most forms provide a Help button that provides access to information relevant to the fields and other aspects of that particular form.

A.2.2.1 Basic-Ops Forms

The following list describes how to access forms via file system-related selections from the Basic-Ops menu:

- Simple Volume/FS Create Form

  Basic-Ops → UFS Operations → Create → Simple

  `Description:` This form creates a concatenated volume and then creates a file system on the new volume. The form is divided into two sections, one for volume creation and the other for file system creation. Most of the form fields are already set to the defaults for the creation of a new volume and file system. The following table describes the fields for this form.

  Most fields in this form are required; those that are optional are listed here. Fields in this form are read/write fields, unless listed as read only.

Field	Description
Volume name:	The name of the volume to be created. The name must be unique within this disk group. The maximum length of this field is 14 characters.
Volume size:	The desired volume size. The size should be entered as a number followed immediately by the letter `k`, `m`, or *s* to indicate kilobytes, megabytes, or sectors, respectively. If no unit is specified, the default is sectors. The volume size should be less than or equal to the available free space of the disks.

Field	Description
Usage Type:	The desired usage type. The `fsgen` file system is the generic usage type, which assumes that the volume is being used by a file system. The `gen` file system is the generic usage type, which makes no assumptions regarding the data content of the volume. The default is `fsgen`.
Create file system:	Indicates whether a file system is to be created. When this form is invoked from the UFS Operations menu, the default is to create a file system (Yes). All fields below this field are only accessible when Yes is specified here.
FS type:	UFS is the only currently supported file system type.
Mount file system:	Indicates whether the file system should be mounted after creation. If the answer is Yes (the default), a mount point must also be specified in the next field. All fields below this field are only accessible when Yes is specified here.
Mount point:	The desired mount point for the new file system. If the specified mount point does not already exist, the Visual Administrator automatically creates it. This field is required if the file system is to be mounted.
Mount automatically:	Indicates whether this file system should be mounted every time the system comes up (by placing an entry in `/etc/fstab`). The default is Yes.

- Striped Volume/FS Create Form

  Basic-Ops → UFS Operations → Create → Striped

  `Description:` This form creates a striped volume and creates a file system on the new volume. The form is divided into two sections, one for volume creation and the other for file system creation. Most of the form fields are already set to the defaults for the creation of a new volume. The following table describes the fields for this form.

  Most fields in this form are required; those that are optional are listed here. Fields in this form are read/write fields, unless listed as read only.

Field	Description
Volume name:	The name of the volume to be created. The name must be unique within this disk group. The maximum length of this field is 14 characters.

Field	Description
Volume size:	The desired volume size. The size should be entered as a number followed immediately by the letter `k`, `m`, or *s* to indicate kilobytes, megabytes, or sectors, respectively. If no unit is specified, the default is sectors. If the size is not wholly divisible by the stripe width, LSM adjusts the volume size up to the next even multiple in order to create the volume. For a striped volume, the volume size should be calculated as follows: *vol_size = stripe_width * number_of_stripes * n*, where *n* is a number greater than zero. The volume size should be less than or equal to the available free space of the disks.
Usage Type:	The desired usage type. The `fsgen` type is the file system generic usage type, which assumes that the volume is being used by a file system. The `gen` type is the generic usage type, which makes no assumptions regarding the data content of the volume. The default is `fsgen`.
Number of Stripes:	The number of stripes that the volume's plex is to have. This is effectively the number of disks on which the volume is to be created. If some number of disks have already been selected, that number of stripes appears in this field. This number corresponds to the number of disks across which data will be striped. If no number is specified, Logical Storage Manager selects an appropriate number (usually 2).
Stripe width:	The width of the stripes on the plex that this volume will have. The value specified may be optimized for the particular drive configuration, as best striping performance is achieved when the stripe width corresponds to the track width of the drive. The default value for this field is 128 sectors, chosen as a good stripe width for most systems.
Create file system:	Indicates whether a file system is to be created. When this form is invoked from the UFS Operations menu, the default is to create a file system (Yes). All fields below this field are only accessible when Yes is specified here.
FS type:	UFS is the only currently supported file system type.
Mount file system:	Indicates whether the file system should be mounted after creation. If the answer is Yes (the default), a mount point must also be specified in the next field. All fields below this field are only accessible when Yes is specified here.
Mount point:	The desired mount point for the new file system. If the specified mount point does not already exist, the Visual Administrator automatically creates it. This field is required if the file system is to be mounted.

Field	Description
Mount automatically:	Indicates whether this file system should be mounted every time the system comes up (by placing an entry in `/etc/fstab`). The default is Yes.

- Make File System Form

  Basic-Ops → UFS Operations → Make

  `Description:` This form is used to make a file system (using `newfs`) according to the user's specifications. The following table describes the fields for this form.

  Most fields in this form are required; those that are optional are listed here. Fields in this form are read/write fields, unless they are listed as read only.

Field	Description
Device name:	This field displays the block device on which to make the file system, which corresponds to the name of the selected volume. This field is read only and cannot be changed.
File system size:	The length of the file system to be made. If no units are specified, sectors are assumed. This length should typically correspond to the length of the volume on which the file system is to be made, although it can be altered for special circumstances.
FS Type:	UFS is the only currently supported file system type.
Mount file system:	Indicates whether the file system should be mounted after creation. If the answer is Yes (the default), a mount point must also be specified in the next field. All fields below this field are only accessible when Yes is specified here.
Mount point:	The desired mount point for the new file system. If the specified mount point does not already exist, the Visual Administrator automatically creates it. This field is required if the file system is to be mounted.
Mount automatically:	Indicates whether this file system should be mounted every time the system comes up (by placing an entry in `/etc/fstab`). Yes is the default.

- Mount File System Form

  Basic-Ops → UFS Operations → Mount

`Description:` This form is used to mount a file system that already exists on a selected volume. The following table describes the fields for this form.

Most fields in this form are required; those that are optional are listed here. Fields in this form are read/write fields, unless listed as read only.

Field	Description
Device name:	This field displays the block device on which to make the file system, which corresponds to the name of the selected volume. This field is read only and cannot be changed.
FS Type:	UFS is the only currently supported file system type.
Mount point:	The desired mount point for the file system. If the specified mount point does not already exist, the Visual Administrator automatically creates it. The Visual Administrator attempts to provide a default mount point, which it obtains by scanning `/etc/fstab`.
Mount automatically:	Indicates whether this file system should be mounted every time the system comes up (by placing an entry in `/etc/fstab`). No is the default.

- File System Check Form

  Basic-Ops → UFS Operations → Check File System (fsck)

  `Description:` This form is used to check a file system that exists on a volume but is not currently mounted. The following table describes the fields for this form.

  The fields in this form are required. Fields in this form are read/write fields, unless listed as read only.

Field	Description
Volume:	This fields displays the name of the volume containing the file system to be checked (with `fsck`). This field is read only and cannot be changed.
FS type:	This fields indicates the type of the file system to be checked.

A.2.2.2 File Systems Properties Form

The following discussion describes the properties form. This form reveals the properties of a particular file system:

- File System Properties Form

  Basic-Ops → UFS Operations → Display Properties

  `Description:` This form provides detailed information on the attributes of a particular file system. This properties form contains a list of mounted file systems, from which the user can select the file system whose properties are to be displayed. The following table describes the fields for this form.

  All fields in this form are read only and cannot be changed.

Field	Description
Mount Point:	The mount point of this file system.
Device:	The block device on which this file system resides.
Block Size:	The block size of this file system.
Default block size:	Fundamental file system block size.
Total disk space:	Number of megabytes of disk storage on this file system available on the disk.
Disk space available:	Number of megabytes of disk storage on this file system that is available for use.
% disk space available:	Percentage of the total disk storage space still available for use. This is the "free space available" divided by the "total disk space."
Total files:	The maximum number of files allowed on this file system.
Free files available:	The number of files that still may be created on this file system.
FS type:	The file system type (such as `ufs`).
Max file name length:	This is the maximum number of characters that a file name may be on this file system. This restriction is imposed by the file system.
FS attributes:	This field indicates attributes associated with this file system: `Read Only` indicates a file system that cannot be written to.

A.3 Volume Operations

The following sections provide information on menus and forms relating to volume operations.

A.3.1 Volume Menus

Both the Basic-Ops and Advanced-Ops menus provide access to volume-related menus. Most menus provide a Help selection, which contains information relevant to the items and operations listed in that particular menu.

A.3.1.1 Basic-Ops Menu

You access the Basic-Ops menu as shown:

Basic-Ops → Volume Operations

This menu provides access to volume operations involving general volume maintenance. These operations use the automated approach to volume management.

The Volume Operations menu provides the following selections:

- Create
- Remove Volumes Recursively
- Add Plex
- Remove Plex
- Resize
- Snapshot
- Help

The Help selection accesses a Help window, which displays information relevant to the basic volume operations.

The following list describes these menu selections:

- Create

  Basic-Ops → Volume Operations → Create

  `Description:` This operation creates a volume from one or more disks. The user may select one or more disks on which to create the volume (providing that there is sufficient space on the disks). If no disks are specified, LSM automatically determines which disks are to be used based on available free space.

  From the Create menu, you select the type of volume to be created from a submenu listing two of the basic types of volumes:

Type	**Description**
Simple	Creates a simple, concatenated volume whose subdisks are arranged both sequentially and contiguously within a plex.
Striped	Creates a volume with data spread fairly evenly across multiple disks by way of striping. *Stripes* are relatively small, equally-sized fragments that are allocated alternately to the subdisks of each plex.

If a mirrored volume is desired, a simple or striped volume must be created and then mirrored using the Add Mirror option.

`Requirements:`

- Only disks in the same disk group can be selected.
- Only LSM disks (disks under LSM control and assigned to a disk group) can be selected.
- If striping is to be in effect, at least two disks are required in order for the operation to succeed.

`Forms:` Simple Volume/FS Create Form and Striped Volume/FS Create Form (described in Section A.3.2).

- Remove Volumes Recursively

Basic-Ops → Volume Operations → Remove Volumes Recursively

`Description:` This operation removes the selected volumes and deallocates all of the disk space set aside for that volume. It automatically removes all underlying plexes and subdisks associated with the volume.

Note

This is a permanent operation and cannot be undone. If completed, it will be difficult or impossible to retrieve the data associated with that volume. For this reason, a confirmation window is presented if the selected volume is not ready for removal (i.e. started or enabled).

`Requirements:`

- At least one volume icon must be selected.
- The selected volumes cannot contain a mounted file system.

- Add Plex

Basic-Ops → Volume Operations → Add Mirror

`Description:` This operation adds a plex to the selected volume by associating a plex of the correct length to the volume. The plex effectively duplicates the information contained in the volume. Although a volume can have a single plex, at least two are required for true mirroring (redundancy of data) to be in effect.

From the Add Mirror menu, you select the type of plex to be added from a submenu listing two of the basic types of plexes:

Type	Description
Simple	Adds a simple, concatenated plex whose subdisks are arranged both sequentially and contiguously.
Striped	Adds a plex whose data is allocated evenly across each of its subdisks in an alternating fashion. This is accomplished with *stripes*, which are relatively small, equally-sized fragments that are allocated alternately to each subdisk.

Disks can be selected for this operation. However, the number of selected disks must be sufficient to accommodate the layout type of both the existing volume and the plex to be added. If no disks are selected, the free space for the plex is allocated by LSM.

`Requirements:`

– A volume icon must be selected.

– For a striped plex, at least two disks other than those already in use by the volume must be available.

- Remove Plex

  Basic-Ops → **Volume Operations** → **Remove Mirror**

  `Description:` This operation removes the selected plex, along with any associated subdisks.

  `Requirements:`

  – A plex icon must be selected.

  – The last valid plex in a started or enabled volume cannot be removed.

- `Resize`

  Basic-Ops → **Volume Operations** → **Resize**

  `Description:` This operation resizes the selected volume. The volume can be increased to, increased by, reduced to, or reduced by a given length. This involves adding or removing disk space to or from the plexes associated with the volume.

If new disk space is needed during the resize, it is allocated as necessary; if space becomes unused, it is added to the free space pool. A disk cannot be selected for this operation.

`Requirements:`

– A volume icon must be selected.

– A striped volume cannot be resized and therefore cannot be selected.

– A volume containing a mounted file system cannot be shrunk.

`Form:` Volume Resize Form (described in Section A.3.2).

- Snapshot

Basic-Ops → **Volume Operations** → **Snapshot**

`Description:` This operation backs up a volume by creating a snapshot image of that volume. This is a convenient way of performing backup with minimal interruption.

This operation invokes the LSM snapshot approach, in which the snapshot operation creates a new volume that is a snapshot of an existing volume. This is done by creating a plex of the existing volume (creating and associating a plex) using disk space from the pool of free disk space. The plex is brought up to date (this may take some time) and a separate (snapshot) volume is then created for that plex. The snapshot volume represents a consistent copy of the original volume at the time the snapshot was begun. The snapshot volume can be used to make a backup of the original volume without stopping it. After the backup is made, the snapshot volume can be removed without losing any data.

Note

For UFS volumes, Digital recommends that you unmount the file system briefly to ensure the snapshot data on disk is consistent and complete.

From the Snapshot menu, a submenu allows you to first create the snapshot plex and then the snapshot volume:

Option	Description
Snapstart	Start the snapshot procedure by creating a snapshot plex within the volume to be backed up. It takes a variable amount of time to update the new plex, during which time the snapshot plex icon is greyed out.

Option	Description
Snapshot	At a convenient time (preferably after warning users to reduce activity briefly), create another volume for the snapshot plex. This portion of the procedure should take only seconds to complete.

`Requirements:`

– A volume icon must be selected.

– There must be sufficient free disk space to accommodate the snapshot volume.

`Form:` Snapshot Form (described in Section A.3.2).

A.3.1.2 Advanced-Ops Menu

You access the Advanced-Ops menu selections as shown:

Advanced-Ops → Volume

This menu provides access to assorted volume operations. These volume operations use the manual approach to volume management. The Volume menu provides the following selections:

- Create
- Remove Volumes
- Initialize Volumes
- Start Volumes
- Stop Volumes
- Resynchronize Volumes
- Set to Maint State
- Recover Volumes
- Help

The Help selection accesses a Help window, which displays information relevant to the advanced volume operations.

The following list describes these menu selections:

- Create

  Advanced-Ops → Volume → Create

  `Description:` This operation creates a volume. The user may select one or more plexes to be associated with the new volume after creation.

`Form:` Volume Create Form (described in Section A.3.2).

- Remove Volumes

  Advanced-Ops → Volume → Remove Volumes

  `Description:` This operation removes the selected volumes. If the selected volume is started, it must be stopped before it can be removed.

  Note

  This is a permanent operation and cannot be undone. Any plexes associated with the volume will be disassociated and left behind.

  `Requirements:`

  – At least one volume icon must be selected.

  – The volume must be stopped before it can be removed.

- Initialize Volumes

  Advanced-Ops → Volume → Initialize Volumes

  `Description:` This operation initializes the selected volumes.

  From the Initialize volumes menu, you select the type of initialization from a submenu listing the following choices:

Option	Description
Active	This enables the selected volume and its associated plexes, and sets the state of all associated plexes to ACTIVE.
Enable	This enables the selected volume and its associated plexes, but leave the plex states as EMPTY.
Clean	This sets the state for all associated plexes of the selected volume to CLEAN. This can be applied only under limited circumstances.
Zero	This enables the selected volume and its associated plexes, then write zeroes over the entire volume. After the operation completes, all associated plexes are set to ACTIVE, assuming that there are no I/O errors.

  `Requirements:`

  – At least one volume icon must be selected.

  – The selected volume cannot have been previously initialized.

- – The selected volume should have at least one associated plex that is complete (or contiguous).

- Start Volumes

  Advanced-Ops → Volume → Start Volumes

  `Description:` This operation starts the selected volumes. A volume must be started before it can be accessed.

  From the Start volumes menu, a submenu allows you to indicate whether all volumes or just those selected should be started:

Option	Description
Start	Start the selected volume, which must be startable.
Start All	Start all volumes in this disk group that can be started.

  `Requirements:`

  – At least one volume icon must be selected for the Start operation. No volume icons need to be selected for the Start All operation.

  – A volume should be initialized before it can be started.

- Stop Volumes

  Advanced-Ops → Volume → Stop Volumes

  `Description:` This operation stops the selected volumes. A volume that is stopped is inaccessible.

  From the Stop volumes menu, a submenu allows you to indicate whether all volumes or just those selected should be stopped:

Option	Description
Stop	Stop the selected volume.
Stop All	Stop all volumes in this disk group.

  `Requirements:`

  – At least one volume icon must be selected for the Stop operation. No volume icons need to be selected for the Stop All operation.

  – A volume must be started before it can be stopped.

  – A volume that is in use or contains a mounted file system cannot be stopped.

- Resynchronize Volumes

  Advanced-Ops → Volume → Resynchronize Volumes

  `Description:` This operation brings all plexes within the selected volumes up to date. Any plexes that are inconsistent are resynchronized to contain consistent data.

  Depending on how current the plexes are, this operation may take some time.

  `Requirements:`

  – At least one volume icon must be selected.

  – The selected volumes must be started.

- Set to Maintenance State

  Advanced-Ops → Volume → Set to Maint State

  `Description:` This operation sets the state of the selected volumes to a maintenance state. Refer to the `volume`(8) reference page for information on the maintenance state.

  `Requirement:` At least one volume icon must be selected.

- Recover Volumes

  Advanced-Ops → Volume → Recover Volumes

  `Description:` This operation recovers the selected volumes.

  `Requirement:` At least one volume icon must be selected.

A.3.2 Volume Forms

Some volume operations result in the appearance of forms, which must be completed in order for that operation to proceed. Most forms provide a Help button, which contains information relevant to the fields and other aspects of that particular form.

A.3.2.1 Basic-Ops Forms

The following forms are accessed via volume-related selections from the Basic-Ops menu:

- Simple Volume/FS Create Form

  Basic-Ops → Volume Operations → Create → Simple

  `Description:` This form creates a concatenated volume and optionally creates a file system on the new volume. The form is divided into two sections, one for volume creation and the other for file system creation. Most of the form fields are already set to the defaults for the creation of a new volume; the file system fields are greyed out because the default is not to add a file system to the volume. The following tables describes the fields for this form.

Most fields in this form are required; those that are optional are listed here. All fields in this form are read/write fields.

Field	Description
Volume name:	The name of the volume to be created. The name must be unique within this disk group. The maximum length of this field is 14 characters.
Volume size:	The desired volume size. The size should be entered as a number followed immediately by the letter `k`, `m`, or *s* to indicate kilobytes, megabytes, or sectors, respectively. If no unit is specified, the default is sectors. The volume size should be less than or equal to the available free space of the disks.
Usage Type:	The desired usage type. The `fsgen` type is the file system generic usage type, which assumes that the volume is being used by a file system. The `gen` type is the generic usage type, which makes no assumptions regarding the data content of the volume. The default is `fsgen`.
Create file system:	Indicates whether a file system is to be created. When you invoke this form from the Volume Operations menu, the default is not to create a file system (No). All fields below this field are only accessible when Yes is specified here.

The following fields only apply if the Create file system: field is set to Yes. Otherwise, these fields are inaccessible.

Field	Description
FS type:	UFS is the only currently supported files system type.
Mount file system:	Indicates whether the file system should be mounted after creation. If the answer is Yes (the default), a mount point must also be specified in the next field. All fields below this field are only accessible when Yes is specified here.
Mount point:	The desired mount point for the new file system. If the specified mount point does not already exist, the Visual Administrator automatically creates it. This field is required if the file system is to be mounted.

Field	Description
Mount automatically:	Indicates whether this file system should be mounted every time the system comes up (by placing an entry in `/etc/fstab`). The default is Yes .

- Striped Volume/FS Create Form

  Basic-Ops → Volume Operations → Create → Striped

  `Description:` This form creates a concatenated volume and optionally creates a file system on the new volume. The form is divided into two sections, one for volume creation and the other for file system creation. Most of the form fields are already set to the defaults for the creation of a new volume; the file system fields are greyed out because the default is not to add a file system to the volume. The following table describes the fields for this form.

  Most fields in this form are required; those that are optional are listed here. All fields in this form are read/write fields.

Field	Description
Volume name:	The name of the volume to be created. The name must be unique within this disk group. The maximum length of this field is 14 characters.
Volume size:	The desired volume size. The size should be entered as a number followed immediately by the letter `k`, `m`, or *s* to indicate kilobytes, megabytes, or sectors, respectively. If no unit is specified, the default is sectors. If the size is not wholly divisible by the stripe width, LSM will adjust the volume size up to the next even multiple in order to create the volume. For a striped volume, the volume size should be calculated as follows: *vol_size* = *stripe_width* * *number_of_stripes* * *n*, where *n* is a number greater than zero. The volume size should be less than or equal to the available free space of the disks.
Usage Type:	The desired usage type. The `fsgen` type is the file system generic usage type, which assumes that the volume is being used by a file system. The `gen` type is the generic usage type, which makes no assumptions regarding the data content of the volume. The default is `fsgen`.

Field	Description
Number of Stripes:	The number of stripes that the volume's plex is to have. This is effectively the number of disks on which the volume is to be created. If some number of disks have already been selected, that number of stripes appears in this field. This number corresponds to the number of disks across which data will be striped. If no number is specified, LSM selects an appropriate number (usually 2).
Stripe width:	The width of the stripes on the plex that this volume will have. The value specified may be optimized for the particular drive configuration, as best striping performance is achieved when the stripe width corresponds to the track width of the drive. The default value for this field is 128 sectors, chosen as a good stripe width for most systems.
Create file system:	Indicates whether a file system is to be created. When you invoke this form from the Volume Operations menu, the default is not to create a file system (`No`). All fields below this field are only accessible when Yes is specified here.

The following fields only apply if you set the Create file system: field to Yes. Otherwise, these fields are inaccessible.

Field	Description
FS type:	UFS is the only currently supported file system type.
Mount file system:	Indicates whether the file system should be mounted after creation. If the answer is Yes (the default), you must also specify a mount point in the next field. All fields below this field are only accessible when Yes is specified here.
Mount point:	The desired mount point for the new file system. If the specified mount point does not already exist, the Visual Administrator automatically creates it. This field is required if the file system is to be mounted.
Mount automatically:	Indicates whether this file system should be mounted every time the system comes up (by placing an entry in `/etc/fstab`). The default is Yes.

- Volume Resize Form

  Basic-Ops → Volume Operations → Resize

  `Description:` This form either grows or shrinks a volume using the Logical Storage Manager free space management resources. If new disk space is needed, it will be allocated as necessary; if space becomes unused, it will be added to the free space pool. The following table describes the fields in this form.

Most fields in this form are required; those that are optional are listed here. Fields in this form are read/write fields, unless they are listed as read only.

Field	Description
Selected Volume:	This field displays the name of the volume to be resized. This field is read only and cannot be changed.
Current size:	This field displays the current size of the volume to be resized. This field is read only and cannot be changed.
Option:	The type of resize operation to be performed. This will determine whether the volume is grown or shrunk to a certain size, or grown or shrunk by a given amount. The default is Grow To.
Size/Amount:	Enter either the length to which or the amount by which the volume is to be resized. If Grow To or Shrink To is selected, this field should reflect the final size. If Grow By or Shrink By is selected, this field should reflect the amount by which the size should change. The new volume size should be less than or equal to the available free space of the disks.

- Snapshot Form

  Basic-Ops → Volume Operations → Snapshot

  `Description:` This form creates a snapshot of the selected volume for backup purposes. The following table describes the fields for this form.

  Fields in this form are required. Fields in this form are read/write fields, unless they are listed as read only.

Field	Description
Selected Volume:	This field displays the name of the volume to be used as the snapshot source. This field is read only and cannot be changed.
Snapshot name:	The name of the snapshot volume to be created as a backup. Although a default name appears in this field, a name that more closely resembles that of the selected volume should be used for easier association. The maximum length is 14 characters. The snapshot name must be unique.

`Requirement:` There must be sufficient free space to accommodate the snapshot volume.

A.3.2.2 Advanced-Ops Forms

The following forms are accessed via volume-related selections from the Advanced-Ops menu:

- Volume Create Form

  Advanced-Ops → Volume → Create

  `Description:` This form creates a volume according to the user's specifications. The following table describes the fields for this form.

  Most fields in this form are required; those that are optional are listed here. Fields in this form are read/write fields, unless listed as read only.

Field	Description
Volume name:	The name of the volume to be created. The name must be unique within this disk group. The maximum length of this field is 14 characters. The name specified for the volume must be unique within this disk group.
Usage Type:	The desired usage type. The `fsgen` type is the file system generic usage type, which assumes that the volume is being used by a file system. The `gen` type is the generic usage type, which makes no assumptions regarding the data content of the volume. The default is `fsgen`. This field is optional.
User:	The name of the user who will be the owner of this volume. This must be a valid user name on the system. The maximum length of this field is 14 characters.
Group:	The name of the group that will own this volume. This must be a valid group name on the system. The maximum length of this field is 14 characters.
Mode:	The permissions mode for the new volume. Only numbers of the correct format are valid in this field. The maximum length of this field is 4 characters.
Length:	The length of the volume. If no unit is specified, the default is sectors. Only positive numbers greater than zero are valid. This field is optional.
Plexes:	This field displays the number of plexes associated with the volume. If no plexes were selected prior to invoking this form, this field displays 0. This field is read only and cannot be changed.

Field	Description
Read Policy:	The read policy that the volume adopts when deciding which plex to write to. These policies are distinguished as follows: Round Robin — All plexes are read equally, in turn. Preferred Plex — A particular plex is specified as the plex to be read whenever possible. The preferred plex will not be read in situations such as when that plex is detached due to I/O failure. Based on plex layouts — All plexes are read equally and in turn, unless a striped plex is present, in which case the striped plex becomes the preferred plex. This option is the default and it typically gives the best read performance.
Preferred Plex:	The name of the preferred plex if the Preferred Plex read policy has been specified. The string in this field must be the name of a valid plex that is associated with this volume. This field is required if Preferred Plex is specified in the Read Policy: field.
Comment:	An appropriate comment for this volume. The maximum length of the comment is 40 characters. This field is optional.
Startup:	This field may contain an arbitrary string that is reserved for the user by usage-type utilities. The intention is that this field be used to store options that apply to the volume, such as for the start volumes operation. This is normally a comma-separated list of flag names and *option=value* pairs. This field is optional.
Logging:	Indicates whether logging is defined and supported on this volume. An undefined log type is included to support old versions of the Logical Storage Manager. The default is Don't Log.
Writeback:	Indicates whether the volume is to write back on read failure. If set to Yes, an attempt will be made to fix a read error from a participating plex. The default is No.
Putil0:	Permanent utility field 0. This is reserved for Logical Storage Manager use, but may be changed. The maximum length of all Putil fields is 14 characters. This field is optional.
Putil1:	Permanent utility field 1. This field is reserved, but may be changed. This field is optional.
Putil2:	Permanent utility field 2. This field is reserved, but may be changed. This field is optional.

A.3.3 Volume Properties Form

The following is the properties form that reveals the properties of a particular volume:

- Volume Properties Form

You can access this form by clicking the MB3 mouse button on the desired volume icon. (If volume icon is undergoing analysis, press Shift-MB3 instead.)

`Description:` This form provides detailed information on the attributes of a particular volume. The following table describes the fields in this form.

The fields in this form are read/write fields, unless listed as read only. Properties of the volume can be changed via this form by altering the current values in the appropriate read/write fields and then clicking on the Apply button.

Field	Description
Volume name:	The name of the volume. This name must be unique within this disk group. The maximum length of this field is 14 characters. This volume name can be changed by entering another name in this field.
Usage Type:	The volume usage type. The `fsgen` type is the file system generic usage type, which assumes that the volume is being used by a file system. The `gen` type is the generic usage type, which makes no assumptions regarding the data content of the volume.
Utility State:	The state that the volume is currently in. This should be either Started, Startable, or Unstartable. This field is read only and cannot be changed.
User:	The name of the user who owns this volume. This must be a valid user name. The maximum length of this field is 14 characters.
Group:	The name of the group that will own this volume. This must be a valid group name. The maximum length of this field is 14 characters.
Mode:	The permissions mode for the volume. Only numbers of the correct format are valid in this field. The maximum length of this field is 4 characters.
Length:	The length of the volume. If no unit is specified, the default is sectors. Only positive numbers greater than zero are valid.
Plexes:	This field displays the number of plexes associated with the volume. If no plexes were selected prior to invoking this form, this field displays 0. This field is read only and cannot be changed.

Field	Description
Read Policy:	The read policy that the volume adopts when deciding which plex to write to. These policies are distinguished as follows: Round Robin — All plexes are read equally, in turn. Preferred Plex — A particular plex is specified as the plex to be read whenever possible. The preferred plex will not be read in situations such as when that plex is detached due to I/O failure. Based on plex layouts — All plexes are read equally and in turn, unless a striped plex is present, in which case the striped plex becomes the preferred plex. This option is the default and it typically gives the best read performance.
Preferred Plex:	The name of the preferred plex if the Preferred Plex read policy has been specified. The string in this field must be the name of a valid plex that is associated with this volume. This field applies only if Preferred Plex is specified in the Read Policy: field.
Comment:	A comment relevant to this volume. The maximum length of the comment is 40 characters.
Startup:	This field may contain an arbitrary string that is reserved for the user by usage-type utilities. The intention is that this field be used to store options that apply to the volume, such as for the start volumes operation. This is normally a comma-separated list of flag names and *option=valuepairs.*
Logging:	Indicates whether logging is defined and supported on this volume. An undefined log type is included to support old versions of the Logical Storage Manager.
Writeback:	Indicates whether the volume is to write back on read failure. If set to Yes, an attempt will be made to fix a read error from a participating plex.
Putil0:	Permanent utility field 0. This is reserved for Logical Storage Manager use, but may be changed. The maximum length of all Putil fields is 14 characters.
Putil1:	Permanent utility field 1. This field is reserved, but may be changed.
Putil2:	Permanent utility field 2. This field is reserved, but may be changed.
Tutil0:	Temporary utility field 0. This is reserved for LSM use, but may be changed. The maximum length of all Tutil fields is 14 characters.
Tutil1:	Temporary utility field 1. This field is reserved, but may be changed.
Tutil2:	Temporary utility field 2. This field is reserved, but may be changed.

Field	Description
Kernel State:	The kernel state of this volume. These states are distinguished as follows: Enabled — The volume block device can be used. This is the default state. Detached — The volume block device cannot be used, but ioctls will still be accepted and the plex block devices will still accept reads and writes. Disabled — The volume or its plexes cannot be used for any operations.
Number of IO Failures:	The number of I/O operations that have failed on this volume. This field cannot be changed.

A.4 Plex Operations

The following sections provide information on menus and forms relating to plex operations.

A.4.1 Plex Menus

You access plex operations via the Advanced-Ops menu, as shown here:

Advanced-Ops → Plex

The Advanced-Ops menu provides access to the following plex-related menus:

- Create
- Remove Plexes
- Associate Plexes
- Disassociate Plexes
- Attach Plexes
- Detach Plexes
- Help

The Help selection accesses a Help window that displays information relevant to the plex operations.

The plex Advanced-Ops menus are described in the following list:

- Create

  Advanced-Ops → Plex → Create

`Description:` This operation creates a plex. You can select one or more subdisks to be associated with the new plex after creation.

`Form:` Plex Create Form (described in Section A.4.2).

- Remove Plexes

  Advanced-Ops → Plex → Remove plexes

  `Description:` This operation removes the selected plexes. This is a permanent operation and cannot be undone. Any subdisks associated with the plex will be disassociated and left behind.

  `Requirements:`

  – At least one plex icon must be selected.

  – If the selected plex is associated with a volume, it must be disassociated before it can be removed.

- Associate Plexes

  Advanced-Ops → Plex → Associate Plexes

  `Description:` This operation associates one or more selected plexes with the selected volume. If the volume is started, LSM begins to bring the plex up to date by copying all necessary data to the plex. This may take a fair amount of time.

  `Requirements:`

  – A volume icon and at least one plex icon must be selected.

  – Only nonassociated plexes can be associated.

- Disassociate Plexes

  Advanced-Ops → Plex → Disassociate Plexes

  `Description:` This operation disassociates one or more selected plexes from their parent volumes. This operation will fail if the plex cannot be disassociated. For example, the last plex in a started volume cannot be disassociated.

  `Requirements:`

  – At least one plex icon must be selected.

  – Only associated plexes can be disassociated.

  – Before the last plex in a volume can be disassociated, that volume must be stopped.

- Attach Plexes

  Advanced-Ops → Plex → Attach Plexes

`Description:` This operation attaches one or more selected plexes to their parent volumes. A plex must be detached but still associated with an enabled volume in order to be attached; the plex is actually being reattached with its parent volume.

`Requirements:`

– At least one plex icon must be selected.

– A plex must be detached before it can be attached.

– Only a plex associated with an enabled volume can be attached.

- Detach Plexes

  Advanced-Ops → **Plex** → **Detach Plexes**

  `Description:` This operation detaches one or more selected plexes from their parent volumes. A detached plex is inaccessible for reads and writes, but is still associated with the volume.

  `Requirements:`

  – At least one plex icon must be selected.

  – Only associated plexes can be detached.

  – This operation is not permitted when the specified plex is the last valid plex on the volume.

A.4.2 Plex Forms

Some plex operations result in the appearance of forms, which must be completed in order for that operation to proceed. Most forms provide a Help button, which contains information relevant to the fields and other aspects of that particular form.

The following forms are accessed via plex-related selections from the Advanced-Ops menu:

- Plex Create Form

  Advanced-Ops → **Plex** → **Create**

  `Description:` The following table describes the fields in this form.

  Most fields in this form are required; those that are optional are listed here. Fields in this form are read/write fields, unless listed as read only.

Field	Description
Plex name:	The name of the plex to be created. The name must be unique within this disk group. The maximum length of this field is 14 characters.
Plex state:	The plex utility state. This is reserved for use by usage types. This field is optional.
Volume:	The name of the volume that this plex should be associated with. The name must be a valid volume name in this disk group. The maximum length of this field is 14 characters. This field is optional.
Layout:	The desired layout for the plex. A concatenated plex is a plex with associated subdisks that are both sequentially and contiguously arranged. A striped plex is a plex that scatters data evenly across each of its associated subdisks. The default is Concatenated.
Stripe width:	The width of the stripes on the plex. The stripe width must be a number greater than 0. If no units are specified, sectors are assumed. The maximum length of this field is 14 characters. If Striped plex layout has been specified, this field is required. This field must be blank if Concatenated plex layout has been specified.
Subdisks:	The number of subdisks associated with the plex. This field is read only and cannot be changed.
Comment:	An appropriate comment for the plex. The maximum length of the comment is 40 characters. This field is optional.
Errors:	Indicates whether the plex should participate in LSM error policies. The default is Participate.
Putil0:	Permanent utility field 0. This is reserved for LSM use, but may be changed. The maximum length of all Putil fields is 14 characters. This field is optional.
Putil1:	Permanent utility field 1. This field is reserved, but may be changed. This field is optional.
Putil2:	Permanent utility field 2. This field is reserved, but may be changed. This field is optional.

A.4.3 Plex Properties Forms

The following list describes the properties form that reveals the properties of a particular plex:

- Plex Properties Form

  To access the plex properties form, click the MB3 mouse button on desired plex icon.

`Description:` This form provides detailed information on the attributes of a particular plex. The following table describes the fields in this form.

The fields in this form are read/write fields, unless listed as read only. Properties of the plex can be changed via this form by altering the current values in the appropriate read/write fields and then clicking on the Apply button.

Field	Description
Plex name:	The name of the plex. The name must be unique within this disk group. The maximum length of this field is 14 characters. The plex name can be changed by entering another name in this field.
Plex state:	The plex utility state. This is reserved for use by usage types. This field is read only and cannot be changed.
Volume:	The name of the volume that this plex should be associated with. This field is read only and cannot be changed.
Layout:	The layout of the plex: concatenated or striped. A concatenated plex is a plex with associated subdisks that are both sequentially and contiguously arranged. A striped plex is a plex that scatters data evenly across each of its associated subdisks. This field is read only and cannot be changed.
Stripe width:	The width of the stripes on the plex. If Striped plex layout has been specified, this field indicates the stripe width. This field should be blank if Concatenated plex layout has been specified. This field is read only and cannot be changed.
Subdisks:	The number of subdisks associated with the plex. This field is read only and cannot be changed.
Log Subdisk:	This field shows the name of the subdisk that is being used for logging on this plex. If there is no associated Block Change Logging subdisk (no logging in effect), this field is blank. This field is read only and cannot be changed.
Comment:	An appropriate comment for the plex. The maximum length of the comment is 40 characters.
Errors:	Indicates whether the plex participates in LSM error policies. This field is read only and cannot be changed.
Putil0:	Permanent utility field 0. This is reserved for use, but may be changed. The maximum length of all Putil fields is 14 characters.
Putil1:	Permanent utility field 1. This field is reserved, but may be changed.
Putil2:	Permanent utility field 2. This field is reserved, but may be changed.

Field	Description
Tutil0:	Temporary utility field 0. This is reserved for LSM use, but may be changed. The maximum length of all Tutil fields is 14 characters.
Tutil1:	Temporary utility field 1. This field is reserved, but may be changed.
Tutil2:	Temporary utility field 2. This field is reserved, but may be changed.
Kernel State:	The accessibility of the plex. This field is read only and cannot be changed.
Length:	The length of the plex. This field is read only and cannot be changed.
Number of I/O failures:	The number of I/O operations that have failed on this plex. This field is read only and cannot be changed.

A.5 Subdisk Operations

The following sections provide information on menus and forms relating to subdisk operations.

A.5.1 Subdisk Menus

You access the subdisk Advanced-Ops menu as shown here:

Advanced-Ops → **Subdisk**

This menu provides access to the following subdisk operations:

- Create
- Remove Subdisks
- Associate Subdisks
- Associate as Log Sd
- Disassociate Subdisks
- Join Subdisks
- Split the Subdisk
- Help

The Help selection accesses a Help window that displays information relevant to the subdisk operations.

The following list describes how to access the subdisk menus:

- Create

  Advanced-Ops → **Subdisk** → **Create**

  `Description:` This operation creates a subdisk on the selected LSM disk.

  `Requirement:` An LSM disk must be selected.

  `Form:` Subdisk Create Form (described in Section A.5.2).

- Remove Subdisks

  Advanced-Ops → **Subdisk** → **Remove Subdisks**

  `Description:` This operation removes the selected subdisks. This is a permanent operation and cannot be undone.

  `Requirements:`

  – At least one subdisk icon must be selected.

  – If the selected subdisk is associated with a plex, it must be disassociated before it can be removed. Only free subdisks can be removed.

  Associate Subdisks

  Advanced-Ops → **Subdisk** → **Associate Subdisks**

  `Description:` This operation associates one or more subdisks with the selected plex.

  `Requirements:`

  – A plex icon and at least one subdisk icon must be selected.

  – Only nonassociated (free) subdisks can be associated.

- Associate as Log Subdisk

  Advanced-Ops → **Subdisk** → **Associate as Log Sd**

  `Description:` This operation associates the selected subdisk as a log subdisk with the selected plex. Block Change Logging is in effect. The resulting log subdisk icon has double borders to distinguish it from normal subdisks.

  `Requirements:`

  – A plex icon and a subdisk icon must be selected.

  – Only nonassociated (free) subdisks can be associated.

  – The selected plex cannot already have a log subdisk.

- Disassociate Subdisks

  Advanced-Ops → Subdisk → Disassociate Subdisks

  `Description:` This operation disassociates one or more selected subdisks from their parent plexes. Both log subdisks and normal subdisks can be disassociated.

  `Requirements:`

  – At least one subdisk icon must be selected.

  – Only associated subdisks can be disassociated.

  – The last subdisk associated with a plex that is currently associated with a volume cannot be disassociated. The plex must be disassociated from its volume first.

- Join Subdisks

  Advanced-Ops → Subdisk → Join Subdisks

  `Description:` This operation joins the selected subdisks together to create a single subdisk. The resulting subdisk has the offset and name of the first subdisk (as arranged on the disk) and its length is the sum of the subdisk lengths.

  `Requirements:`

  – At least two subdisk icons must be selected.

  – The subdisks must be contiguous on the disk.

  – If the subdisks are associated, they must all be associated with the same plex and be contiguous on that plex.

  – Logging subdisks and subdisks associated with striped plexes cannot be joined.

- Split a Subdisk

  Advanced-Ops → Subdisk → Split the Subdisk

  `Description:` This operation splits the selected subdisk into either two or many parts. The resulting subdisks will occupy the same region on the disk that the previous subdisk occupied. If the subdisk is associated with a plex, the resulting subdisks will also be associated with that plex.

  From the Split the subdisk menu, a submenu allows the user to indicate whether the subdisk is to be split into two or several parts:

  Into 2 Subdisks
  : Split the selected subdisk into 2 subdisks.

  Into More Than 2 Subdisks
  : Split the selected subdisk into several subdisks.

`Requirements:`

- Only one subdisk icon can be selected.
- Logging subdisks and subdisks associated with striped plexes cannot be split.

`Forms:` Subdisk Split Into Two

Subdisk Split Into Many (described in Section A.5.2).

A.5.2 Subdisk Forms

Some subdisk operations result in the appearance of forms, which must be completed in order for that operation to proceed. Most forms provide a Help button, which contains information relevant to the fields and other aspects of that particular form.

The following forms are accessed via subdisk-related selections from the Advanced-Ops menu:

- Subdisk Create Form

  Advanced-Ops → Subdisk → Create

  `Description:` This form creates a subdisk according to the user's specifications. The following table describes the fields in this form.

  Most fields in this form are required; those that are optional are listed here. Fields in this form are read/write fields, unless listed as read only.

Field	Description
Disk name:	The name of the LSM disk on which the subdisk is to be created. This field is read only and cannot be changed.
Subdisk name:	The name of the subdisk to be created. The name must be unique within this disk group. The maximum length of this field is 14 characters.
Disk offset:	The length into the disk where this subdisk should be located. If no units are specified, sectors are assumed. This offset should not place this subdisk within the bounds of another subdisk on the disk or past the end of the disk. Only valid positive numbers are allowed in this field.
Subdisk length:	The length of the subdisk to be created. If no units are specified, sectors are assumed. The length should not place this subdisk within the bounds of another subdisk on the disk or past the end of the disk. Only valid positive numbers are allowed in this field.

Field	Description
Plex name:	The name of the plex with which the subdisk is to be associated. This must be a valid plex that already exists in this disk group. The maximum length of this field is 14 characters. This field is optional.
Plex offset:	The offset of this subdisk into its associated plex. Only valid positive numbers are allowed in this field. This field is required only if a plex has been specified for association. If the subdisk is not to be associated with a plex, this field must be left blank.
Comment:	An appropriate comment for the subdisk. The maximum length of the comment is 40 characters. This field is optional.
Putil0:	Permanent utility field 0. This is reserved for Logical Storage Manager use, but may be changed. The maximum length of all Putil fields is 14 characters. This field is optional.
Putil1:	Permanent utility field 1. This field is reserved, but may be changed. The maximum length of this field is 14 characters. This field is optional.
Putil2:	Permanent utility field 2. This field is reserved, but may be changed. The maximum length of this field is 14 characters. This field is optional.

- Subdisk Split Into Two

Advanced-Ops → Subdisk → Split the Subdisk –> Into 2 Subdisks

`Description:` This form is used to split the selected subdisk into exactly 2 subdisks. The first subdisk retains the name and size of the original one; the second subdisk adopts the name and size specified in this form. The following table describes the fields in this form.

Most fields in this form are required; those that are optional are listed here. Fields in this form are read/write fields, unless listed as read only.

Field	Description
Present size:	The size of the subdisk to be split. This field is read only and cannot be changed.
Name of new subdisk:	The name of the subdisk to be created from the original one. This must be a valid name and must be unique in this disk group.
Size of new subdisk:	The size of the subdisk to be created from the original one. This must be a valid number, greater than zero. The new subdisk size must be at least one sector less than the present subdisk size.

- Subdisk Split Into Many

  Advanced-Ops → Subdisk → Split the Subdisk –> Into More Than 2 Subdisks

  `Description:` This form is used to split the selected subdisk into several subdisks of equal sizes. The first subdisk retains the name and size of the original one; the additional subdisks are automatically named by LSM. The following table describes the fields in this form.

  Most fields in this form are required; those that are optional are listed here. Fields in this form are read/write fields, unless listed as read only.

Field	Description
Original subdisk:	The name of the selected subdisk. This field is read only and cannot be changed.
Present size:	The size of the subdisk to be split. The original subdisk must contain enough sectors to accommodate the desired total number of subdisks for the split. This field is read only and cannot be changed.
Number of new subdisks:	The total number of subdisks to be created by the split. There must be a sufficient number of sectors in the original subdisk to accommodate this number. This number should be at least 2.

  `Requirements:` The number of subdisks is limited by the amount of space left in the configuration database.

A.5.3 Subdisk Properties Forms

The following is the properties form that reveals the properties of a particular subdisk:

- Subdisk Properties Form

  To access the Subdisk Properties form, click the MB3 mouse button on desired subdisk icon.

  (If the subdisk is undergoing analysis, press Shift-MB3 instead.)
  `Description:` This form provides detailed information on the attributes of a particular subdisk. The following table describes the fields in this form.

  The fields in this form are read/write fields, unless listed as read only. Properties of the subdisk can be changed via this form by altering the current values in the appropriate read/write fields and then clicking on the Apply button.

Field	Description
Disk name:	The name of the disk where the subdisk resides. This field is read only and cannot be changed.
Subdisk name:	The name of the subdisk. The name must be unique within this disk group. The maximum length of this field is 14 characters. The subdisk name can be changed by entering another name in this field.
Disk offset:	The length into the disk where this subdisk is located, in sectors. This field is read only and cannot be changed.
Subdisk length:	The length of the subdisk. If no units are specified the number is assumed to be in sectors. This offset should not place this subdisk within the bounds of another subdisk on the disk or past the end of the disk. Only valid positive numbers are allowed in this field.
Plex name:	The name of the plex with which the subdisk is associated. This field is read only and cannot be changed.
Plex offset:	The offset of this subdisk into its associated plex. If the subdisk is not associated, this field contains a zero. This field is read only and cannot be changed.
Comment:	An appropriate comment for the subdisk. The maximum length of the comment is 40 characters.
Log Subdisk:	Indicates whether this subdisk is a Block Change Logging subdisk. This field is read only and cannot be changed.
Putil0:	Permanent utility field 0. This is reserved for LSM use, but may be changed. The maximum length of all Putil fields is 14 characters.
Putil1:	Permanent utility field 1. This field is reserved, but may be changed. The maximum length of this field is 14 characters.
Putil2:	Permanent utility field 2. This field is reserved, but may be changed. The maximum length of this field is 14 characters.
Tutil0:	Temporary utility field 0. This is reserved for LSM use, but may be changed. The maximum length of all Tutil fields is 14 characters.
Tutil1:	Temporary utility field 1. This field is reserved, but may be changed. The maximum length of this field is 14 characters.
Tutil2:	Temporary utility field 2. This field is reserved, but may be changed. The maximum length of this field is 14 characters.
Number of IO failures:	The number of I/O operations that have failed on this subdisk. This field is read only and cannot be changed.

A.6 Disk Operations

The following sections provide information on menus and forms relating to disk operations.

A.6.1 Disk Menus

Both the Basic-Ops and Advanced-Ops menus provide access to disk-related operations. Most menus provide a Help selection, which contains information relevant to the items and operations listed in that particular menu.

A.6.1.1 Basic-Ops Menu

You access the Basic-Ops menu as shown:

Basic-Ops → Disk Operations

This menu provides access to disk operations involving general disk maintenance. These operations use the automated approach to disk management.

The Disk Operations menu provides the following selections:

- Add Disks
- Evacuate Subdisks
- Replace Disks
- Remove Disks
- Help

The Help selection accesses a Help window, which displays information relevant to the basic disk operations.

The following list describes the menu selections you can access via the Basic-Ops menu:

- Add Disks

  Basic-Ops → Disk Operations → Add Disks

  `Description:` This operation adds a disk to the Logical Storage Manager, placing it under LSM control. This involves initializing, analyzing, and partitioning the raw disk; initializing the disk for LSM use; and adding the disk to a disk group (if requested).

  `Form:` Add Disks Form (described in the Disk Forms section).

- Evacuate Disks

  Basic-Ops → Disk Operations → Evacuate Disks

`Description:` This operation moves all subdisks from the selected disk to another disk in the same disk group.

`Requirements:` The disk from which subdisks are to be evacuated must be selected. Both disks must belong to the same disk group.

`Forms:` Evacuate Subdisks Form (described in the Disk Forms section).

- Replace Disks

  Basic-Ops → **Disk Operations** → **Replace Disks**

  `Description:` This operation replaces a disk. This is normally done when a failed disk needs to be replaced with a new one. This involves initializing and partitioning the raw disk; initializing the disk for LSM use; and replacing the old disk and associated disk media records with the new disk and its information.

  `Requirements:` A disk icon representing a failed disk must be selected.

  `Forms:` Replace Disks Form (described in the Disk Forms section).

- Remove Disks

  Basic-Ops → **Disk Operations** → **Remove Disks**

  `Description:` This operation removes a disk from a disk group and then removes the disk from LSM control.

  `Requirements:` A disk icon must be selected.

A.6.1.2 Advanced-Ops Menu

You access the Advanced-Ops menu selections as shown:

Advanced-Ops → **Disk**

This menu provides access to assorted disk operations using the manual approach to disk management.

The Disk menu provides the following selections:

- Initialize
- Define
- Remove
- Online
- Offline

- Help

The Help selection accesses a Help window, which displays information relevant to the advanced disk operations.

The following list describes the menu selections you can access via the Advanced-Ops menu:

- Initialize

  Advanced-Ops → Disk → Initialize

  `Description:` This operation identifies a disk to LSM and initializes the disk for LSM use. This involves installing a disk header and writing an empty configuration on the disk. A disk access record is created for the disk, unless such a record already exists.

  `Requirement:` The disk should not already be initialized.

  `Form:` Disk Init Form (described in the Disk Forms section).

- Define

  Advanced-Ops → Disk → Define

  `Description:` This operation defines a disk access record that enables LSM to scan the disk. This makes the disk accessible, but does not initialize the disk.

  `Form:` Define Disk Form (described in the Disk Forms section).

- Remove

  Advanced-Ops → Disk → Remove

  `Description:` This operation removes the LSM disk associated with the selected partitions from LSM control by removing the associated disk access records. If all partitions on a given disk are selected for removal at once, the disk is effectively removed from LSM control.

  `Requirements:`

  – At least one partition icon corresponding to a LSM disk must be selected.

  – The LSM disks corresponding to the selected partitions cannot belong to a disk group at the time of removal.

- Online

  Advanced-Ops → Disk → Online

  `Description:` This operation places the disk access record on a specified partition in an online state. During searches for disk IDs or members of a disk group, online disks are checked.

`Form:` Disk Online Form (described in the Disk Forms section).

- Offline

  Advanced-Ops → Disk → Offline

  `Description:` This operation places the disk access record on the selected partitions in an offline state. During searches for disk IDs or members of a disk group, offline disks are ignored.

  `Requirements:`

  – At least one partition icon must be selected.

  – The disks corresponding to the selected partitions must be initialized.

  – The selected partition icon cannot be in use (shaded and associated with a LSM disk).

A.6.2 Disk Forms

Some disk operations result in the appearance of forms. You must complete these forms in order for that operation to proceed. Most forms provide a Help button that provides access to information relevant to the fields and other aspects of that form.

A.6.3 Basic-Ops Forms

The following forms are accessed via disk-related selections from the Basic-Ops menu.

- Add Disks Form

  Basic-Ops → Disk Operations → Add Disks

  `Description:` This form is used to place a disk under Logical Storage Manager control. The following table describes the fields in this form.

  Most fields in this form are required; those that are optional are listed here. Fields in this form are read/write fields, unless listed as read only.

Field	**Description**
New disk name:	The name of the new *physical* disk in the form `rz`*`nn`*, for example, `rz10`. The name must be unique within this disk group. You can also place specific partitions on a disk under LSM control. For example, `rz3g` would put the `g` partition on `rz3` under LSM control.

Field	Description
Disk group:	The name of the disk group to which this disk is to be added. The named disk group must exist. If no name is provided, it will not be added to a disk group. This field is optional.

- Replace Disks Form

  Basic-Ops → Disk Operations → Replace Disks

  `Description:` This form is used to replace an existing LSM disk that has failed with another one. The following table describes the fields in this form.

  Most fields in this form are required; those that are optional are listed here. Fields in this form are read/write fields, unless listed as read only.

Field	Description
Old LSM disk name:	The name of the failed (collapsed or disconnected) LSM disk in this disk group. This field is read only and cannot be changed.
New physical disk name:	The name of the new *physical* disk that is to replace the existing one. The name should be in the form `rz`*nn*, for example, `rz10`. The new name must be unique in this disk group.

- Evacuate Subdisks Form

  Basic-Ops → Disk Operations → Evacuate Subdisks

  `Description:` This form is used to transfer subdisks from one LSM disk to another. The following table describes the fields in this form.

  Most fields in this form are required; those that are optional are listed here. Fields in this form are read/write fields, unless listed as read only.

Field	Description
Disk group name:	The name of the disk group to which both disks belong. Both disks must share the same disk group.
Evacuate From:	The name of the LSM disk from which the subdisks are to be evacuated.

Field	Description
To:	The name of the LSM disk to which the subdisks are to be moved. This field is optional. However, if no target disk is specified, the subdisks are evacuated to one or more random disks (depending on disk space availability).

A.6.3.1 Advanced-Ops Forms

The following forms are accessed via disk-related selections from the Advanced-Ops menu:

- Disk Init Form

  Advanced-Ops → Disk → Initialize

  `Description:` This form is used to initialize a disk for LSM use. The following table describes the fields in this form.

  Most fields in this form are required; those that are optional are listed here. Fields in this form are read/write fields, unless listed as read only.

Field	Description
Public Device	The pathname of the device node that represents a partition available for use. This name must be a valid entry in `/dev`. A name in the form `rz`*nn* is used to assign the full disk under LSM control. The disk `rz`*nn* would be added as a sliced LSM disk. Before a sliced disk can be defined, change the disk label to have LSM disk label tags. A name in the form `rz`*nnp* is used to assign partition *p* on disk `rz`*nn* under LSM control. The disk partition `rz`*nnp* would be added as a simple LSM disk.
Device Type	The desired disk type. The simple type (default) assumes that the public and private regions are stored on the same disk partition, with the public region following the private region. The sliced type assumes that the public and private regions are stored on different disk partitions. Before initializing the disk, change the disklabel to have LSM disklabel tags. The nopriv type has no private region and log and configuration copies cannot be written to the disk.
Public length (0 for whole device)	The length of the public section of the disk. If zero is provided as the length, the Logical Storage Manager computes a default value from available partition table information. This length must be valid and cannot exceed the length of the disk.

Field	Description
Private Length:	The length of the private region of the disk. When one is not specified, LSM chooses a default value. This length must be valid and cannot exceed the length of the disk. For a sliced disk, the length cannot exceed the size of the partition chosen for the private region. This field is optional.
Number of config copies:	The number of configuration copies to be stored in the private section of this disk. The default value is 2 copies.
Number of log copies:	The number of log copies to be stored in the private section of this disk. The default value is 2 copies.
Comment:	A comment appropriate for the LSM disk. The maximum length of the comment is 40 characters. This field is optional.

- Define Disk Form

  Advanced-Ops → Disk → Define

  `Description:` This form is used to define a disk. The following table describes the fields in this form.

  Most fields in this form are required; those that are optional are listed here. Fields in this form are read/write fields, unless listed as read only.

Field	Description
Public Device	The pathname of the device node that represents a partition available for use. This name must be a valid entry in `/dev`. A name in the form `rz`*`nn`* is used to assign the full disk under LSM control. The disk `rz`*`nn`* would be added as a sliced LSM disk. A name in the form `rz`*`nnp`* is used to assign partition *`p`* on disk `rz`*`nn`* under LSM control. The disk partition `rz`*`nnp`* would be added as a simple LSM disk.
Device Type	The desired disk type. The simple type (default) assumes that the public and private regions are stored on the same disk partition, with the public region following the private region. The sliced type assumes that the public and private regions are stored on different disk partitions. The nopriv type has no private region and log and configuration copies cannot be written to the disk.
Public Length (0 for whole disk):	The length of the public section of the disk. If zero is provided as the length, LSM computes a default value from available partition table information. This length must be valid and cannot exceed the length of the disk.

Field	Description
Offline:	Indicates whether to initially place the disk in the offline state. The default is No.
Comment:	A comment appropriate for this Logical Storage Manager disk. The maximum length of the comment is 40 characters. This field is optional.

- Disk Online Form

  Advanced-Ops → Disk → Online

  `Description:` This form is used to online a disk. The following table describes the fields in this form.

Field	Description
Device name:	The disk access name of the disk to be onlined. This must be a valid disk access name. This field is required.

- Free Space Form

  To access the free space form, click the MB3 mouse button on a gap between subdisk icons in a LSM disk icon.

  `Description:` This form provides information about a specific region of an LSM disk that contains free space.

  Free space results when subdisks are removed for some reason, making the space that they occupied available for use. Free space is visually represented as a gap or hole between subdisks that reside on a LSM disk icon. The following table describes the fields in the form.

Note

All fields in this form are read only and cannot be changed.

Field	Description
Device:	The name of the LSM disk where this free space resides.
Hole offset:	The offset into the LSM disk where this free space extent begins.
Hole size:	The size of this free space extent. The units used are specified by the user under the Options pull down menu.

A.6.4 Disk Properties Forms

Properties forms exist for LSM disks, physical disks, and partitions. The following list describes these forms:

- LSM Disk Properties Form

  To access the LSM disk properties form, click the MB3 mouse button on desired LSM disk icon. (If the LSM disk icon is undergoing analysis, press Shift-MB3 instead.)

  `Description:` This form provides detailed information on the attributes of a particular LSM disk that is under LSM control. The information displayed in this form actually corresponds to the disk media record associated with a disk. The following table describes the fields in this form.

  The fields in this form are read/write fields, unless listed as read only. Properties of the disk can be changed via this form by altering the current values in the appropriate read/write fields and then clicking on the Apply button.

Field	Description
LSM disk name:	The name of the LSM disk.
Disk Access:	The name of the disk access record that corresponds to this disk media record. This field is read only and cannot be changed.
Disk Type:	The type with which this disk media record was created. This field is read only and cannot be changed.
Public Region:	The name of the public region of this disk. This field is read only and cannot be changed.
Private Region:	The name of the private region of this disk. If there is no private region then this field will be blank. This field is read only and cannot be changed.
Public Region Offset:	The offset, in sectors, of the public region on the disk. This field is read only and cannot be changed.
Private Region Offset:	The offset, in sectors, of the private region on the disk. If there is no public region, then this field will display zero. This field is read only and cannot be changed.
Public Region Length:	The length, in sectors, of the public region on the disk. This field is read only and cannot be changed.
Private Region Length:	The length, in sectors, of the private region on the disk. If there is no private region, this field will display zero. This field is read only and cannot be changed.

Field	Description
Disk Attributes:	The attributes of this LSM disk. This field is read only and cannot be changed.
Comment:	The user-specified comment for this LSM disk. The maximum length of the comment is 40 characters.
Putil0:	Permanent utility field 0. This is reserved for LSM use, but may be changed. The maximum length of all Putil fields is 14 characters.
Putil1:	Permanent utility field 1. This field is reserved, but may be changed.
Putil2:	Permanent utility field 2. This field is reserved, but may be changed.
Tutil0:	Temporary utility field 0. This field is reserved, but may be changed. The maximum length of all Tutil fields is 14 characters.
Tutil1:	Temporary utility field 1. This field is reserved, but may be changed. The maximum length of this field is 14 characters.
Tutil2:	Temporary utility field 2. This field is reserved, but may be changed. The maximum length of this field is 14 characters.
Maximum Free Space:	The maximum amount of free space available on this LSM disk. This does not take disk extents into account. This number assumes every free sector on the LSM disk is usable. This field is read only and cannot be changed.

- Physical Disk Properties Form

  To access the physical disk properties form, click the MB3 mouse button on desired physical disk icon.

  `Description:` This form provides detailed information on the attributes of a particular physical disk. The following table describes the fields in this form.

  All fields in this form are read only and cannot be changed.

Field	Description
Device:	The raw device node for this physical disk.
Device Type:	A brief description the device type. Possible device types include SCSI hard drive and Floppy.
Cylinders:	The number of cylinders on this disk.
Tracks:	The number of tracks per cylinder.
Sectors:	The number of sectors per track.

Field	Description
Sector Size:	The size, in bytes, of each sector on this disk.
Total Size:	The total size of the disk, in sectors.

- Partition Properties Form

  To access the partition properties form, click the MB3 mouse button on desired partition icon.

  `Description:` This form provides detailed information on the attributes of a particular partition. The following table describes the fields in this form.

  All fields in this form are read only and cannot be changed.

Field	Description
Device:	The device node that the LSM Visual Administrator uses to communicate with this disk.
Start Sector:	The sector on the physical disk where this partition begins.
Size:	The length of this partition.
Type:	The identification tag associated with this partition.
Disk Media:	The disk media record that corresponds to this partition. If this field is empty, the partition has not been initialized with a disk media record.

A.7 Disk Group Operations

The following sections provide information on menus and forms relating to disk group operations.

Note

With the Visual Administrator, partition icons represent partitions containing disk access records.

A.7.1 Disk Group Menus

You access disk group operations via the Advanced-Ops menu, as shown here:

Advanced-Ops → Disk Group

The Advanced-Ops menu provides access to the following disk-related menus.

- Initialize
- Import Disk Groups
- Deport Disk Groups
- Add Disk
- Remove Disks
- Disconnect Disks
- Reconnect Disks
- Help

The Help selection accesses a Help window that displays information relevant to the disk group operations.

The following list describes the disk group menu options:

- Initialize

  Advanced-Ops → Disk Group → Initialize

  `Description:` This operation defines a new disk group with a name specified by the user. The new disk group contains one or more LSM disks corresponding to the partitions selected by the user.

  `Requirements:` At least one partition icon must be selected.

  `Form:` Initialize Disk Group Form (described in the Disk Group Forms section).

- Import Disk Group

  Advanced-Ops → Disk Group → Import Disk Group

  `Description:` This operation imports a disk group to make that disk group available on the local machine. If the name of a deported disk group is known, this operation can be used to make that disk group accessible again.

  `Form:` Import Disk Group Form (described in the Disk Group Forms section).

- Deport Disk Group

  Advanced-Ops → Disk Group → Deport Disk Group

  `Description:` This operation disables access to a disk group. A deported disk group is no longer accessible and its view window disappears. Once deported, a disk group can be reimported.

  `Requirements:` A disk group cannot be deported if any volumes in that disk group are currently open.

`Form:` Deport Disk Group Form (described in the Disk Group Forms section).

- Add Disk

  Advanced-Ops → Disk Group → Add Disk

  `Description:` This operation adds a LSM disk corresponding to the selected partition icon to a disk group. This involves creating a disk media record for the disk to be added. Partitions representing disks that already belong to disk groups cannot be added to disk groups.

  `Requirements:`

  – One partition icon must be selected.

  – The selected partition cannot already belong to a disk group.

  – Only one disk can be added to a disk group at a time.

  `Form:` Add Disk Form (described in the Disk Group Forms section).

- Remove Disks

  Advanced-Ops → Disk Group → Remove Disks

  `Description:` This operation removes the selected LSM disks from a disk group. Disks are removed from the disk group in which they reside. Any subdisks that exist on the selected disks must be removed before the disk can be removed.

  `Requirements:`

  – At least one LSM disk icon must be selected.

  – Only disks associated with the specified disk group can be removed.

  – Disks containing any subdisks cannot be removed.

  – Only disks in the same disk group can be selected for removal in a single operation.

  – The last disk in a disk group cannot be removed. The disk group itself must be deported in order for its last disk to be removed.

- Disconnect Disks

  Advanced-Ops → Disk Group → Disconnect Disks

  `Description:` This operation disables the selected LSM disk, making it unavailable for use within its disk group. This involves disassociating the disk media record from its disk access record.

  `Requirements:`

- At least one LSM disk icon must be selected.
- The LSM disk icons must contain a disk media record at the time of selection.

- Reconnect Disks

  Advanced-Ops → Disk Group → Reconnect Disks

  `Description:` This operation enables a LSM disk that has previously been disconnected. This involves connecting the selected LSM disk's disk media record with the selected disk access record. Although the LSM disk must be disconnected, it does not necessarily have to be reconnected to its former partition (disk access record).

  `Requirements:`

  - One LSM disk icon and one partition icon must be selected.
  - Neither the LSM disk icon nor the partition icon can already be connected.

A.7.2 Disk Group Forms

Some disk group operations result in the appearance of forms, which must be completed in order for that operation to proceed. Most forms provide a Help button, which contains information relevant to the fields and other aspects of that particular form.

The following forms are accessed via disk group-related selections from the Advanced-Ops menu:

- Initialize Disk Group Form

  Advanced-Ops → Disk Group → Initialize

  `Description:` This form is used to define a new disk group consisting of selected disks.
  The following table describes the fields in this form.

Field	Description
Disk group:	The name of the new disk group. This must be a valid and unique name. This field is required. This is a read/write field.

- Import Disk Group Form

  Advanced-Ops → Disk Group → Import Disk Group

  `Description:` This form is used to make the specified disk group available to the system. The following table describes the fields in this form.

Field	Description
Disk group:	The name of the disk group to be imported and made available to the system. This must be a valid and unique disk group name. This field is required. This is a read/write field.

- Deport Disk Group Form

  Advanced-Ops → Disk Group → Deport Disk Group

  `Description:` This form is used to make the specified disk group inaccessible to the system. The following table describes the fields in this form.

Field	Description
Disk group:	The name of the disk group to be deported and made inaccessible to the system. This must be a valid disk group.

  `Requirements:` The root disk group (rootdg) cannot be deported.

- Add Disk Form

  Advanced-Ops → Disk Group → Add Disk

  `Description:` This form is used to add a LSM disk to a disk group. The following table describes the fields in this form.

  Most fields in this form are required; those that are optional are listed here. Fields in this form are read/write fields, unless listed as read only.

Field	Description
Disk group:	The name of the disk group to which the LSM disk is to be added. This must be a valid disk group. This field is required.
Disk media name:	The name of the LSM disk to be created. The disk media name must be unique. By default, a unique name is generated. If this field is left blank, then the disk access name is used.

A.8 Projection Analysis

The following sections provide information on menus and forms relating to projection and analysis. In addition, tables are provided to summarize various aspects of projection and analysis behavior. You can access these operations as follows:

- Projection

  Projection operations are accessed via the Projection menu. This menu is located in view windows such as View of rootdg. The Projection menu start or stop projection, as well as to highlight any free subdisk icons.

  Projection can also be started or stopped by pressing Shift-MB2 with the pointer positioned on the desired icon.

- Analysis

  Analysis operations are accessed via the Analyze menu. This menu is located in view windows such as View of rootdg. The Analyze menu can be used to start or stop analysis, as well as to set analysis-related preferences.

A.8.1 Projection

Icon projection provides the user with visual information about the relationships between icons. When projection is started for an icon, all other icons (representing LSM objects) associated with that particular one are highlighted, no matter which views they occupy. Icons can be placed under projection either individually or in multiples. Projection highlighting can accumulate on a given icon when that icon is is undergoing projection from more than one source.

A.8.2 Projection Menus

The following list describes the menus, submenus, and menu selections you can access via the Projection menu:

- Icon Projection

  Projection → Icon Projection

  `Description:` This menu provides access to projection options used to start or stop projection for icons.

- Start

  Projection → Icon Projection → Start

  `Description:` This option starts projection for the selected icons. When projection is started, all icons related to the selected icons are highlighted. Highlighting occurs for related icons in any view windows. If the selected icon has no associated objects, the Visual Administrator issues a warning to this effect.

`Requirements:`

– At least one icon must be selected.

– Physical disk and partition icons cannot be selected for projection.

– The selected icons must be associated with at least one other icon in order for projection to take effect.

- Stop

  Projection → **Icon Projection** → **Stop**

  `Description:` This options stops projection for the selected icons. When projection is stopped, all icons related to the selected icons lose their projection highlighting.

  `Requirement:` At least one icon must be selected. If the selected icon is not undergoing projection, the Visual Administrator ignores the stop request.

- Stop All

  Projection → **Icon Projection** → **Stop All**

  `Description:` This options stops projection for all icons that are currently undergoing selection.

- Show Free Subdisks

  Projection → **Show Free Subdisks**

  `Description:` This menu selection determines whether free subdisks should be highlighted or not. When Show Free Subdisks is turned on, the Visual Administrator highlights all unassociated subdisks (representing unallocated disk space). Once turned on, any future free subdisks are automatically highlighted. Free subdisk icons can be used by designating them to objects, but the LSM Visual Administrator interface cannot automatically use free subdisks as free space. Free subdisk projection is either started or stopped across all Visual Administrator views. The start or stop preference is also retained for a particular user in future sessions.

  From the Show Free Subdisks menu, a submenu allows you to indicate whether or not to highlight free subdisks:

Option	Description
Start	Start highlighting free subdisks immediately and continue to do so until instructed to stop.

Option	Description
Stop	Stop highlighting free subdisks.

A.8.3 Projection Relationships

Table A-1 summarizes the projection relationships that are highlighted for particular icon types. If no icons of the correct type are associated with the selected icon, then nothing is highlighted.

Table A-1: Projection Table

Icon Selected	Icons Highlighted
Volume	All subdisks associated with any plex associated with the volume
Plex	All subdisks associated with the plex
Subdisk	Associated plex and volume, and all other subdisks associated with the plex
LSM Disk	All plexes associated with the subdisks that reside on the disk

A.9 Analysis

Analysis is the LSM Visual Administrator's way of displaying statistics on the performance of various LSM objects.

Statistics are displayed both visually (via color or pattern) and numerically (via pop-up statistics forms).

A.9.1 Analysis Menus

The following menu selections are accessed via the Analyze menu:

- Start

  Analyze → Start

  `Description:` This menu selection begins analysis of the selected icons. These icons are added to the list of objects being analyzed. Only volume and LSM disk icons can be analyzed. Once analysis is activated, the selected icons begin to display information about their performance characteristics.

`Requirement:` At least one volume or LSM disk icon must be selected.

- Stop

  Analyze → Stop

  `Description:` This menu selection terminates analysis of the selected icons. These icons are removed from the list of objects being analyzed. When analysis stops, the selected icons return to their preanalysis states. When analysis is stopped for one icon, other icons undergoing analysis are not affected.

  `Requirements:`

  – At least one volume or LSM disk icon must be selected.

  – The selected icons must be undergoing analysis.

- Stop All

  Analyze → Stop All

  `Description:` This menu selection automatically terminates analysis of all icons in all views. All icons return to their preanalysis states.

  `Requirements:` Analysis must be in effect.

- `Parameters`

  Analyze → Parameters

  `Description:` This menu selection accesses the Analysis Parameters form, which is used to set user preferences for how analysis is to be conducted.

  `Form:` Analysis Parameters Form (described in the Analysis Forms section).

A.9.2 Analysis Forms

The following forms are accessed via the Analyze menu:

- Analysis Parameters Form

  Analyze → Parameters

  `Description:` This form is used to set user preferences for conducting analysis. The following table describes the fields in this form.

  Most fields in this form are required; those that are optional are listed here. Fields in this form are read/write fields, unless listed as read only.

  Field **Description**

Field	Description
Sample Rate:	Determines the time interval between data samples. This field is divided into two sections: the slider bar is used to select the interval (1-60) and the menu to the right is used to select units of time (seconds or minutes). The default is 5 seconds. A shorter interval means the data will be updated more often, but is also a higher load on the system.
Volume Parameters:	Specifies the high and low values that decide the coloring (or pattern) of the volume icons.
Disk Parameters:	Specifies the high and low values that decide the coloring (or pattern) of the LSM disk icons.
Subdisk Parameters:	Specifies the high and low values that decide the coloring (or pattern) of the subdisk icons.
Log File:	The name of the file to be used for the statistics log. If the file does not already exist, it will be created. The filename is taken to be relative unless a pathname is given. To stop logging to the file, the filename text in this field must be erased. This field is optional. This log file is a binary file. In order to view the log file, `/usr/bin/lsmlog2text` *filename* must be run on this file to process it for viewing.

`Requirements:`

- For each set of high/low parameters, the high parameter must be greater than the low parameter.
- The user must have access to the specified log file.

• Analysis Statistics Form

To access the analysis statistics form, click the MB3 mouse button on desired icon that is being analyzed. `Description:` This form displays analysis statistics relevant to the selected volume or LSM disk icon. This form applies only to volume or disk icons that are undergoing analysis. The following table describes the fields in this form.

Note

All fields in this form are read only and cannot be changed.

Field	Description
Reads:	The number of times the object was read from during the last interval.

Field	Description
Writes:	The number of times the object was written to during the last interval.
Total R/W:	The total number of reads and writes during the last interval.
Blocks Read:	The number of disk blocks read from the object during the last interval.
Blocks Written:	The number of disk blocks written to the object during the last interval.
Total Blocks:	The total number of blocks read from or written to the object during the last interval.
Avg Read Time:	The average time, in milliseconds, that it took for a read operation to complete. This is equal to the number of number of reads during the last interval divided by the total time spent on reads.
Avg Write Time:	The average time, in milliseconds, that it took for a write operation to complete. This is equal to the number of writes during the last interval divided by the total time spent on writes.
Interval:	The actual time, in seconds, since the last data was sampled. This may vary slightly from the specified interval time due to uncontrollable variances from system to system.

`Requirements:` The icon selected by clicking MB3 must be undergoing analysis.

A.9.3 Analysis Table

Table A-2 summarizes the default colors and patterns associated with the various levels of analysis. These defaults can be changed using the dxlsm-related X resources. See `dxlsm`(8) for more information.

Table A-2: Analysis Table

Analysis Level	Color	Bitmap Pattern
low	green	cross_weave
medium	yellow	root_weave
high	red	wide_weave

Logical Storage Manager Error Messages B

The Logical Storage Manager is fault-tolerant and resolves most problems without system administrator intervention. If the volume daemon (`vold`) recognizes what actions are being taken, it can roll a transaction forward or back. When `vold` is unable to recognize and fix system problems, you need to handle the task of problem solving.

The following sections cover the majority of informational, failure, and error messages displayed by `vold` and the kernel driver. These sections include some errors that are infrequently encountered and difficult to troubleshoot. Clarifications are included to elaborate on the situation or problem that may have generated a particular message. Wherever possible, a recovery procedure (user action) is provided to locate and correct potential problems.

Should it be necessary to contact your customer support organization, these messages are numbered for ease of reference.

B.1 Volume Daemon Error Messages

The following list contains the error messages associated with the volume daemon.

1. `Message:`

   -r must be followed by 'reset'

   `Clarification:` This message is caused by a usage error.

   `User Action:` Correct the usage and try again.

2. `Message:`

   -x *argument***: prefix too long**

   `Clarification:` The stub-mode device path prefix name supplied exceeded the maximum of 32 characters.

   `User Action:` Select an alternate path for device files and retry the command.

3. `Message:`

   -x *string***: invalid debug string**

`Clarification:` An unknown argument string was given to the `-x` option to `vold`.

`User Action:` Select a valid string from the reference page for `vold` and try again.

4. `Message:`

   Usage: vold [-dkf] [-r reset] [-m mode] [-x level]
   For detailed help use: vold help

   `Clarification:` `vold` was invoked with an invalid set of arguments.

   `User Action:` Correct the usage and try again or type `vold help` for more help. This is the full usage message from entering `vold help:`

```
Usage:  vold [-dkf] [-r reset] [-m mode] [-x level]
Recognized options:
 -d       set initial mode to disabled for transactions
 -k       kill the existing configuration daemon process
 -f       operate in foreground; default is background
 -r reset reset kernel state; requires 'reset' option argument
 -m mode  set vold's operating mode
          modes: disable, enable, bootload, bootstart
 -x level set debugging level to <debug>, 0 turns off debugging
 -R file  set filename for client request rendezvous
 -D file  set filename for client diag request rendezvous
```

5. `Message:`

   lsm:vold: Error: volume *volume_name***: Logging daemon killed by\
   signal** *signal_number* **[core dumped]**

   `Clarification:` Someone killed the logging daemon.

   `User Action:` If required, restart the daemon with a call to `voliod logio`.

6. `Message:`

   lsm:vold: Error: /dev/volevent: *error_message*
   lsm:vold: Error: cannot open /dev/volconfig: *error_message*
   lsm:vold: Error: Cannot kill existing daemon, pid=*process_id*

   `Clarification:` An attempt to kill an existing `vold` process with a SIGKILL signal has failed. This might be due to the process being in an unkillable kernel state perhaps because of a hung I/O or a missing I/O interrupt. There may be disk driver error messages in the `/dev/osm` buffer.

   `User Action:` Try typing `cat /dev/osm` to see if any other messages have been output to the console device. If possible, use `crash` to determine the state of the process. If the process is asleep waiting for an I/O completion, then any disk driver error messages that have occurred might point to the solution. Failing this, a reboot is recommended.

7. `Message:`

   lsm:vold: Error: /dev/voliod: VOL_LOGIOD_CHECK failed

   `Clarification:` Some inconsistency between `vold` and the kernel has caused an ioctl to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

   `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reinstall of the LSM package. If this fails, contact Customer Support.

8. `Message:`

   lsm:vold: Error: /dev/voliod: VOL_LOGIOD_KILL failed

   `Clarification:` Some inconsistency between `vold` and the kernel has caused an ioctl to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

   `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration of LSM. If this fails, contact Customer Support.

9. `Message:`

   lsm:vold: Error: All transactions are disabled

   `Clarification:` This message may appear with the message `Disk group disabled by errors` if the disk group to be disabled is the root disk group. The continued use of the system could be dangerous since any configuration changes required (including error handling cases) could cause the loss of ability to perform I/O to a volume. Because this includes the root volume, this situation could, if uncorrected, cause the system to hang.

   `User Action:` This is a fatal error. All copies of the bootable root disk have failed. Recovery from this situation will require booting from floppy or from a disk unconnected with LSM. It may then be necessary to remove the LSM rootable disk configuration by using the `volunroot` command. See the LSM installation instructions for details. Once this has been achieved, the root disk group can be reinitialized to reestablish the database and log areas.

10. `Message:`

    lsm:vold: Error: Cannot get all disk groups from the kernel

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an `ioctl` to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

    `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reinstall of the LSM package. If this fails, contact Customer Support.

11. `Message:`

    lsm:vold: Error: Cannot get all disks from the kernel

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an `ioctl` to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

    `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by an LSM reconfiguration. If this fails, contact Customer Support.

12. `Message:`

    lsm:vold: Error: Cannot get kernel transaction state

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an ioctl to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

    `User Action:` Try stopping and restarting the `vold` daemon. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration LSM. If this fails, contact Customer Support.

13. `Message:`

    lsm:vold: Error: Cannot get private storage from kernel

    `Clarification:`

    `User Action:`

14. `Message:`

    lsm:vold: Error: Cannot get private storage from kernel

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an `ioctl` to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

    `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration LSM. If this fails, contact Customer Support.

15. `Message:`

    lsm:vold: Error: Cannot get private storage size from kernel

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an `ioctl` to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

    `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration LSM. If this fails, contact Customer Support.

16. `Message:`

    lsm:vold: Error: Cannot get record *name* **from the kernel:** *error_message*

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an `ioctl` to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

    `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration LSM. If this fails, contact Customer Support.

17. `Message:`

    lsm:vold: Error: Cannot not make directory *directory_path*

    `Clarification:` When trying to create the specified directory, `vold` got a failure.

    `User Action:` Try creating the directory manually and then issue the command `voldctl enable`.

18. `Message:`

    lsm:vold: Error: Cannot recover operation in progress
    Failed to get group *group_name* **from the kernel**

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an `ioctl` to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

    `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration of LSM. If this fails, contact Customer Support.

19. `Message:`

    lsm:vold: Error: Cannot start *usage_type* **volume, no valid complete plexes**

    `Clarification:` No usable plexes remain for either the root or swap volume. This error is fatal and will result in the message `System startup failed` also appearing and the system being shutdown.

    `User Action:` This is generally an unrecoverable error and will likely require a reload of the system from backups.

20. `Message:`

    lsm:vold: Error: Cannot start *usage_type* **volume, no valid plexes**

    `Clarification` No usable plexes remain for either the root or swap volume. This error is fatal and will result in the message `System startup failed` also appearing and the system being shutdown.

    `User Action:` This is generally an unrecoverable error and will likely require a reload of the system from backups.

21. `Message`

    lsm:vold: Error: Cannot start *usage_type* **volume, volume state is invalid**

    `Clarification:` The volume is not in a state that can be recovered from. This might be because of corruption of the databases or because of an invalid use of the `vold` interfaces without the use of the utilities.

    `User Action:` This is generally an unrecoverable error and will require reloading of the system from backups.

22. `Message:`

    lsm:vold: Error: Cannot store private storage into the kernel

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an `ioctl` to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

    `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration of LSM. If this fails, contact Customer Support.

23. `Message:`

    lsm:vold: Error: Differing version of vold installed

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an `ioctl` to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

`User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration of LSM. If this fails, contact Customer Support.

24. `Message:`

    lsm:vold: Error: Disk *disk_name*, **group** *group_name*, **device** *device_name*: **\ not updated with new host ID**
    Error: *error_message*

    `Clarification:` If the host ID for a system is changed using the `voldctl init` command then all disks in all imported disk groups will need to have the host ID changed to the new ID. If the host ID for a disk cannot be changed, then this message will be displayed. Other problems might also exist for this disk.

    `User Action:` The contents of the disk should be evacuated elsewhere and the disk should be reinitialized.

25. `Message:`

    lsm:vold: Error: Disk group *group_name*, **Disk** *disk_name*: **Cannot auto-import group:** *error_message*

    `Clarification:` The disk group *group_name*, could not be reimported after a system restart. The reason is given as part of the error message. Other error messages may appear which provide more information on what went wrong. Any volumes in the disk group will be unavailable until the error condition is fixed and the disk group is reimported.

    `User Action:` Clear the error condition, if possible, and then import the disk group by hand with `voldg import`. After importing, you should restart all volumes with `voldg -g` *groupname* `-sb`.

26. `Message:`

    lsm:vold: Error: Disk group *group_name*, **Disk** *disk_name*
    : Group name collides with\ record in rootdg

    `Clarification:` The disk group name *group_name*, for the disk group being imported from the named disk, collides with a configuration record in the `rootdg` disk group. Disk groups must have names that do not match any records in the root disk group.

    `User Action:` If you want to import the disk group, you will have to rename the conflicting record in `rootdg` to some other name.

27. Message:

lsm:vold: Error: Disk group *group_name***: Cannot recover temp database error_message**

Clarification: The temp database stored in the root file system could not be opened or read. Other messages will detail the error. This may happen because of an I/O error or a problem in the file system.

User Action: The system should be rebooted and the operation retried.

28. Message:

lsm:vold: Error: Disk group *group_name***: Disabled by errors**

Clarification: This message can appear if the last configuration database or last kernel log area for a disk group became disabled. This could have been due to an I/O error or some other condition. Other messages preceding this one are likely to highlight the root cause.

User Action: Any remaining active volumes should be backed up. The disk group will have to be reinitialized and the disks added again to the group to recover.

29. Message:

lsm:vold: Error: Disk group *group_name***: Errors in some configuration copies:**

Clarification: One or more on-disk database copies were found to contain errors. As a result, the disk group could not be imported. This is probably due to a disk I/O error, or to blocks of a configuration copy being overwritten within invalid contents. Check for messages from the disk driver.
Errors pertaining to specific configuration copies are listed on successive lines. These lines can be in either of the following forms:

File *filename***:** *error_message***: Block** *number***:** *error_message*
Disk *diskname***, copy** *copy_number***:** *error_message***: Block** *number***:** *error_message*

Lines beginning with `File` indicate an error in the special configuration copy file used for storing nonpersistent disk group information. Lines beginning with `Disk` indicate failure of a persistent configuration copy stored on a disk. The copy number indicates which of the disk's configuration copies contains the error.

User Action: If one or more disks for the disk group are currently inaccessible (such as due to a cabling error), make the disks accessible and try to import the disk group again with `voldg import`. Otherwise, the disk group is probably no longer usable and will have to be recreated. All volume configuration information for the disk group is lost.

30. `Message:`

    lsm:vold: Error: Disk group *group_name*: **Reimport of disk group failed:** *error_message*

    `Clarification:` The reload of a disk group into the kernel failed. This could be because the log size for the kernel may not be set or because of some other error in the import procedure. Other messages should indicate the true cause of the failure.

    `User Action:` The operation should be retried unless some other error message leads to a suggested course of action. If this fails, the system should be rebooted.

31. `Message:`

    lsm:vold: Error: Disk group *group_name*: **update failed:** *error_message*

    `Clarification:` This message occurs because a database update failed completely. No complete copy of the database could be written for the disk group. The disk group will be disabled and further access for configuration changes will be disallowed. If this error occurs for the root disk group, it will probably be necessary to reinstall the system.

    `User Action:` Any volumes still active in the disk group should be backed up. The disk group will then have to be reinitialized and the disks added again to it.

32. `Message:`

    lsm:vold: Error: Exec of /sbin/voliod failed

    `Clarification:` An `exec` of `/sbin/voliod` failed.

    `User Action:` Check the existence and permissions of the `/sbin/voliod` command. Try executing the command manually to ensure that it can be run.

33. `Message:`

    lsm:vold: Error: Failed to store commit status list into kernel: *error_message*

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an `ioctl` to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

`User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration of LSM. If this fails, contact Customer Support.

34. `Message:`

    lsm:vold: Error: Fork of logio daemon failed

    `Clarification:` The creation of a process that could then be used as a logging daemon failed.

    `User Action:` Check for messages explaining the reason that a `fork`(2) call failed. Retry the operation.

35. `Message:`

    lsm:vold: Error: GET_VOLINFO ioctl failed: *error_message*
    lsm:vold: Error: Version number of kernel does not match vold

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an `ioctl` to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

    `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration of LSM. If this fails, contact Customer Support.

36. `Message:`

    lsm:vold: Error: Get of current rootdg failed

    `Clarification:` An attempt to retrieve the rootdg from the kernel failed. This might be because of a kernel `vold` inconsistency or could also be because of a version difference between `vold` and the kernel.

    `User Action:` Check that the correct version of `vold` and the kernel are installed. Other messages might suggest other problems in a prior attempt at loading a configuration and possible courses of action. If this fails, contact Customer Support.

37. `Message:`

    lsm:vold: Error: No convergence between root disk group and disk list Disks in one version of rootdg:
    disk_name **type=***disk_type* **info=***disk_info*
    Disks in alternate version of rootdg:
    disk_name **type=***disk_type* **info=***disk_info*

    `Clarification:` This message can appear when `vold` is not running in autoconfigure mode (see the `vold`(8) reference page) and when, after several retries, it cannot resolve the set of disks belonging to the root disk group. The algorithm for non-autoconfigure disks is to scan disks listed in the `/etc/vol/volboot` file and then examine the disks to find a database copy for the rootdg disk group. The database copy is then read to find the list of disk access records for disks contained in the group. These disks are then examined to ensure that they contain the same database copy. As such, this algorithm expects to gain convergence on the set of disks and the database copies contained on them. If a loop is entered and convergence cannot be reached, then this message will appear and the root disk group importation will fail.

`User Action:` Reorganizing the physical locations of the devices attached to the system may break the deadlock. If this fails, contact Customer Support.

38. `Message:`

    lsm:vold: Error: Open of directory *directory_path* **failed**

    `Clarification:` When `vold` was trying to create node files for the volumes, it was unable to open the directory in which the nodes were to be created.

    `User Action:` Check for other errors that suggest why the directory might be missing or if the permissions might be incorrect. Fix the condition to allow `vold` to open or create the directory, then issue the command `voldctl enable`.

39. `Message:`

    lsm:vold: Error: Read of directory *directory_path* **failed**

    `Clarification:` The node directory could not be read when `vold` was trying to scan for volume nodes.

    `User Action:` Check for other messages that might suggest why the directory is inaccessible. Try reading the directory manually if the directory is corrupted, then try removing and recreating it and then restarting `vold`.

40. `Message:`

    lsm:vold: Error: Unexpected configuration tid for group *group_name* **found in kernel**

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an `ioctl` to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

    `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration of LSM. If this fails, contact Customer Support.

41. `Message:`

    lsm:vold: Error: Unexpected error during *usage_type* **volume reconfiguration:** *error_message*

    `Clarification:` A record lock for the volume could not be acquired as part of the initial volume setup for either a root or swap volume. This is most likely to occur under low memory conditions.

    `User Action:` Other messages may suggest an alternate course of action. Otherwise, this is generally an unrecoverable error and will require either the boot of an alternate root device or reloading of the system from backups.

42. `Message:`

    lsm:vold: Error: Unexpected error fetching disk for *usage_type* **volume:** *error_message*

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an `ioctl` to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

    `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration of LSM. If this fails, contact Customer Support.

43. `Message:`

    lsm:vold: Error: Unexpected values stored in the kernel

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an `ioctl` to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

    `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration of LSM. If this fails, contact Customer Support.

44. `Message:`

    lsm:vold: Error: VOL_RESET_KERNEL failed: a volume or plex device is open

    or

    lsm:vold: Error: VOL_RESET_KERNEL failed: *error_message*

    `Clarification:` An attempt at resetting the kernel state with a `vold -r reset` command failed because all the LSM objects in the kernel were not closed. If any volumes are in use, then the reset cannot be performed. This may also happen if a reset was requested on a system with root volumes. Root volumes are, by definition, never closed and so a reset cannot be performed.

`User Action:` If a reset is really desired, then checking the state of the volumes and any mounted file systems should result in information about who might have them open. Unmounting all volumes and killing any processes accessing the volumes should allow the reset to occur.

45. `Message:`

    lsm:vold: Error: *mode***: Unrecognized operating mode**

    `Clarification:` An unknown mode string was entered following a `-m` option.

    `User Action:` Select a valid mode from the `vold`(8) reference page and try again.

46. `Message:`

    lsm:vold: Error: cannot open /dev/voliod: *error_message*

    `Clarification:` The open of the `/dev/voliod` file can only fail if the device node is missing or has an incorrect major or minor number.

    `User Action:` Check the existence and values of the file and make sure that LSM was correctly installed.

47. `Message:`

    lsm:vold: Error: cannot open *argument***:** *error_message*

    `Clarification:` The tracefile specified on the command line could not be opened in append mode. The error message supplied should explain the reason.

    `User Action:` Select an alternate tracefile name that can be created or appended to.

48. `Message:`

    lsm:vold: Error: cannot open *volconfig_device***: Device is already open**

    or

    lsm:vold: Error: cannot open *volconfig_device***:** *error_message*

    `Clarification` The exclusive open device (`/dev/volconfig`) is already open. Only one `vold` process can be active on the system at one time. Subsequent attempts at starting `vold` or opening the device will result in this message.

`User Action:` Check for other running `vold` processes. The `voldctl mode` will report if `vold` is currently active.

49. `Message:`

    lsm:vold: Error: enable failed: *error_message*

    `Clarification:` This message may occur during an initial startup of `vold`. If changing to enabled mode when this error occurs, failures could be due to problems with the creation of the portal or with connection to the kernel. If changing from an enabled state to a disabled state, then problems could occur with removing the disk groups from the kernel because of such things as volumes in use.

    `User Action:` Evaluate other error messages occurring with this one to determine the root cause of the problem. Make changes suggested by the other errors and then retry the command.

50. `Message:`

    lsm:vold: Error: failed to create daemon: fork failed: *error_message*

    `Clarification:` The call to `fork`(2) to generate a background `vold` process failed.

    `User Action:` Check for messages explaining the reason that a `fork`(2) call failed. Retry the operation.

51. `Message:`

    lsm:vold: Error: volume *volume_name***: Wait for logging daemon failed**

    `Clarification:` The wait called to wait for the existence of the daemon process did not execute correctly. This can only happen if the `ioctl` does not correctly match the command required, perhaps because of a mismatch between the `voliod` command and the kernel versions or perhaps because of an incorrect minor number for the `/dev/voliod` device.

    `User Action:` Check the existence and permissions of the `/dev/voliod` device.

52. `Message:`

    lsm:vold: FATAL Error: Disk group rootdg: Inconsistency -- Not loaded into kernel

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an ioctl to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

`User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration of LSM. If this fails, contact Customer Support.

53. `Message:`

    lsm:vold: FATAL Error: Group *group_name***: Cannot update kernel**

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an ioctl to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

    `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration of LSM. If this fails, contact Customer Support.

54. `Message:`

    lsm:vold: FATAL Error: Interprocess communication failure: *error_message*

    `Clarification:` The portal to client utilities has returned a failure. This is a fatal error since without a portal to clients, `vold` cannot do anything useful.

    `User Action:` Check for other errors suggesting the reason for portal failure. Restart `vold`. If problems persist, reboot the system.

55. `Message:`

    lsm:vold: FATAL Error: Invalid status stored in kernel

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an ioctl to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

    `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration of LSM. If this fails, contact Customer Support.

56. `Message:`

    lsm:vold: Warning: Cannot create device *path***:** *error_message*

    `Clarification:` The `mknod`(2) call made by `vold` to create a device node failed. The reason for the error should be displayed.

    `User Action:` Fix the reason indicated for node creation failure and then issue the command `voldctl enable`.

57. `Message:`

    lsm:vold: Warning: Cannot exec /sbin/rm to remove *directory_path***:**

error_message

`Clarification:` An `exec` of `/sbin/rm` failed.

`User Action:` Ignore the error. It is not serious if the directory could not be removed.

58. `Message:`

    lsm:vold: Warning: Cannot fork to remove directory *directory_path***:** *error_message*

    `Clarification:` The call to `fork`(1) to generate a process that could then `exec rm`(2) failed.

    `User Action:` Ignore the error. It is not serious if the directory could not be removed.

59. `Message:`

    lsm:vold: Warning: Disk *device_name* **in kernel not a recognized type**

    `Clarification:` The disk type of a disk in the kernel does not match any known disk type. This can only occur if `vold` and the kernel are in an inconsistent state.

    `User Action:` Try stopping and restarting `vold`. If this fails then reconfigure LSM. If this fails, contact Customer Support.

60. `Message:`

    lsm:vold: Warning: Disk *disk_name* **names group** *group_name***, but group ID differs**

    `Clarification:` As part of a disk group import, a disk was discovered that had a mismatched disk group name and disk group ID. This disk will not have been imported. This can only happen if two disk groups of the same name exist that have different disk group ID values. In that case, one group will be imported along with all its disks and the other group will not. This message will appear for disks in the unselected group.

    `User Action:` If it turns out that the disk should be imported into the group, then this will have to be done by adding the disk to the group at a later stage. It will not happen automatically as part of the import. All configuration information for the disk will also be lost.

61. `Message:`

    lsm:vold: Warning: Disk group *group_name* **is disabled, disks not updated with\ new host ID**

`Clarification:` If the host ID for a system is changed using the `voldctl init` command then all disks in all imported disk groups will need to have the host ID changed to the new ID. If a disk group is found in the imported but disabled state, then the host ID will not be changed.

`User Action:` The host ID will need to be cleared using the `voldisk clearimport` command for each disk, and then the disk group should be reimported.

62. `Message:`

lsm:vold: Warning: Disk group *group_name***: Disk group log may be too small**
Log size should be at least *number* **blocks**

`Clarification:` The log areas for the disk group have become too small for the size of configuration currently in the group. This should usually never happen without first displaying a message about the database area size. This message only occurs during disk group import; it occurs if the disk was inaccessible while new database objects were added to the configuration, and the disk was then made accessible and the system restarted.

`User Action:` If this situation does occur, then the disks in the group will have to be explicitly reinitialized with larger log areas. See the reference page for `voldisk`(8). To reinitialize all the disks, they must be detached from the group with which they are associated and then reinitialized and readded. The group should then be deported and reimported for the changes to the log areas for the group to take effect.

63. `Message:`

lsm:vold: Warning: Disk group *group_name***: Errors in some configuration copies:**

`Clarification:` One or more on-disk database copies were found to contain errors. As a result, the disk group could not be imported. This is most likely to be due to a disk I/O error, or to blocks of a configuration copy being overwritten within invalid contents. Check for messages from the disk driver. Providing that other copies of the database can be successfully read, the system will continue and the disk group import or initial `vold` enable operation should succeed. If the database copy can subsequently be written to, then this message will not recur.

Errors pertaining to specific configuration copies are listed on successive lines. These lines can be in either of the following forms:

File *filename***:** *error_message***: Block** *number***:** *error_message*
Disk *diskname***, copy** *copy_number***:** *error_message***: Block** *number***:** *error_message*

Lines beginning with `File` indicate an error in the special configuration copy file used for storing nonpersistent disk group information. Lines beginning with `Disk` indicate failure of a persistent configuration copy stored on a disk. The copy number indicates which of the disk's configuration copies contains the error.

`User Action:` This message is likely to occur once due to an I/O failure and then not reoccur. If it does reoccur, then it may be necessary to remove the disk and reinitialize it to clear the condition. If all configuration copies for a disk group become unusable, then the disk group itself becomes unusable and must be recreated. If the `rootdg` disk group becomes unusable, LSM may need to be reconfigured. In this case, if root file system is on a volume, then the operating system itself may need to be reinstalled.

64. `Message:`

    lsm:vold: Warning: Error in volboot file: *error_message*
    Entry: disk *disk_name disk_type disk_info*

    `Clarification:` This message occurs when an entry in the `volboot` file does not contain the correct information to define a valid disk access record.

    `User Action:` Remove the entry using the `voldctl rmdisk` command and add it again using `voldctl adddisk`.

65. `Message:`

    lsm:vold: Warning: Failed to update voldinfo area in kernel: *error_message*

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an ioctl to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

    `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration of LSM. If this fails, contact Customer Support.

66. `Message:`

    lsm:vold: Warning: Field too long in volboot file:
    Entry: disk *disk_name disk_type disk_info*

    `Clarification:` The `volboot` file is maintained by `vold` and `voldctl` and should never normally exhibit this problem. This problem might indicate some corruption of the `volboot` file or could also be the result of manual editing of the file.

    `User Action:` The offending entry could try to be removed by use of the `voldctl rmdisk` command. If this fails, `volboot` may have to be reinitialized using a `voldctl init` command.

67. `Message:`

    lsm:vold: Warning: Get of record *record_name* **from kernel failed:** *error_message*

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an ioctl to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

    `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration of LSM. If this fails, contact Customer Support.

68. `Message:`

    lsm:vold: Warning: Plex *plex_name* **for** *usage_type* **volume is stale or unusable**

    `Clarification:` This message is output to alert the user to the failure of one or more plexes of either the root or swap volume. The system may be able to continue depending on the existence of other usable plexes for the volume.

    `User Action:` The failed plex should be repaired by either reattaching the plex to the volume once the system is booted, or by evacuating and replacing the disk on which the failed plex resides if it is thought that the disk is going bad.

69. `Message:`

    lsm:vold: Warning: cannot remove group *group_id* **from kernel:** *error_message*

    `Clarification:` Some inconsistency between `vold` and the kernel has caused an `ioctl` to fail. This could be caused by the use of older versions of `vold` or the kernel, or it could be due to a bug in LSM.

    `User Action:` Try stopping and restarting `vold`. If this fails then a reboot of the system should be attempted, possibly followed by a reconfiguration of the LSM package. If this fails, contact Customer Support.

70. `Message:`

    lsm:vold: Warning: response to client *client number* **failed:** *error_message*

    `Clarification:` The portal to client utilities has returned a failure.

This is a fatal error since without a portal to clients, `vold` cannot do anything useful. This could be caused by a STREAMS error or some other communications problem with the client.

`User Action:` Check for other errors suggesting the reason for portal failure. Restart `vold`. If problems persist, reboot the system.

B.2 Kernel Error Messages

The following are the kernel level error messages.

1. `Message:`

   NOTICE: *message* **on volume device** *hex_device_number* **(***volume_name***) in diskgroup** *group_name*

   `Clarification:` This is caused by a driver above the LSM level calling the LSM `volprint`() function. This usually happens when a driver detects some error condition in LSM and want to display the error.

   `User Action:` No action necessary, unless specified in a supplied string.

2. `Message:`

   NOTICE: io/vol.c(volerror): Correctable *type* **error on volume** *volume_name***,e plex** *plex_name***, block** *block_number*

   `Clarification:` A correctable I/O error was detected and corrected. A correctable I/O error is one where a read error from an underlying device driver could be corrected by reading the data from an alternate mirror copy and then writing it back to the failed mirror.

   `User Action:` If the I/O could have been completed by reading from an alternate mirror but the writeback to the failed mirror still failed, the mirror will be detached. This failure will cause the exception handling code to be entered, which will result in the volume's error recovery policy being followed. This usually results in either a mirror or the volume becoming detached. The user must intervene to reattach the mirror (`volplex att`), to bring back the failed mirror copy. If the volume was detached, then the data contained on it is unrecoverable and will have to be restored from backups.

3. `Message:`

   NOTICE: io/vol.c(volerror): Uncorrectable *type* **error on volume** *volume_name***, \ plex** *plex_name***, block** *block_number*

   `Clarification:` Following an I/O error from one mirror, an attempt to reread the data from an alternate mirror failed. This could be because no other mirrors exist or could be because the other mirrors also had I/O failures.

`User Action:` This failure will cause the exception handling code to be entered, which will result in the volume's error recovery policy being followed. This can have effects ranging from detaching a mirror to disabling the volume. The user must intervene to reattach the mirror (`volplex att`), to bring back the failed mirror copy. If the volume was detached, then the data contained on it is unrecoverable and will have to be restored from backups.

4. `Message:`

   NOTICE: lsm: Can't close disk *disk_name* **in group** *disk_group***.**
   If it is removable media (like a floppy), it may have been removed.
   Otherwise, there may be problems with the drive.
   Kernel error code *error_number/error_number*

   `Clarification:` This is unlikely to happen; closes cannot fail.

   `User Action:` None.

5. `Message:`

   NOTICE: lsm: Can't open device *disk_name***, device busy or inaccessible.**

   `Clarification:` The named disk cannot be accessed.

   `User Action:` Turn on the drive.

6. `Message:`

   WARNING: io/vol.c(volexcept): No volume error daemon - Cannot Log plex detach,\detaching volume

   `Clarification:` No `voliod` process was running and able to log a detach record for a mirror that is being detached due to an I/O error. This is a fatal error that causes future access to the volume to be rejected, because any system failure coming after additional I/O would not be able to detect the failure of the mirror and mirror inconsistencies might then occur.

   `User Action:` Although it is too late to rescue this volume, at least one `voliod` process should be started as soon as possible (using `voliod set 2`). The failed volume will have to be stopped and restarted, then reloaded from backups. Mirrors will have become inconsistent and so any attempt at using the data on the volume could prove disastrous.

7. `Message:`

   WARNING: volklog_dgfree: Can't clear group commit log record for group *disk_group*

   `Clarification:` This can occur if a log flush to disk could not be performed because no valid log copies remained. This is likely to compromise the ability of the LSM to recover from any further I/O errors.

   `User Action:` Disks should be added to the system such that new viable logging areas can be generated. Alternatively, failed disks should be removed and replaced with working devices.

8. `Message:`

   WARNING: volklog_dgfree: Can't free kernel logging area for vol_reset_kernel of group *disk_group*

   `Clarification:` A free of the logs for a disk group failed because either no valid log areas remained for flushing or some log records remained in the log before the clear operation was requested.

   `User Action:` No user action can be taken here; this is a LSM internal error. Contact Customer Support.

LSM Command Examples C

This appendix contains sample LSM commands and system output that show how to perform common LSM administrative tasks.

C.1 Setting Up LSM for the First Time

The following examples show the commands you use to set up LSM for the first time. Refer to Chapter 3 and the `volsetup`(8) and `volinstall`(8) reference pages for further information about setting up LSM.

The following command lists the disks configured on system:

```
# file /dev/rr*c
/dev/rrz10c: character special (8/18434) SCSI #1 RZ26 ...
/dev/rrz12c: character special (8/20482) SCSI #1 RZ26 ...
   .
   .
   .
```

The following command identifies disk partitions already in use:

```
# mount; swapon -s
/dev/rz3a on / type ufs (rw)
/proc on /proc type procfs (rw)
/dev/rz3g on /usr type ufs (rw)
/dev/rz3h on /usr/users type ufs (rw)
Swap partition /dev/rz3b (default swap):
   .
   .
   .
```

The following example confirms disk file system types (fstypes) fields that are unused:

```
# disklabel rz10; disklabel rz12
     size  offset  fstype [fsize bsize   cpg]
a: 131072       0  unused   1024 8192  # (Cyl.    0 - 164*)
b: 262144  131072  unused   1024 8192  # (Cyl.  164*- 492*)
c:2050860       0  unused   1024 8192  # (Cyl.    0 - 2569)
d: 552548  393216  unused   1024 8192  # (Cyl.  492*- 1185*)
e: 552548  945764  unused   1024 8192  # (Cyl. 1185*- 1877*)
f: 552548 1498312  unused   1024 8192  # (Cyl. 1877*- 2569*)
g: 819200  393216  unused   1024 8192  # (Cyl.  492*- 1519*)
h: 838444 1212416  unused   1024 8192  # (Cyl. 1519*- 2569*)
```

The following commands initialize the disklabel tag for any new disks:

```
# disklabel -rw /dev/rrz10c rz26
# disklabel -rw rz12 rz26
```

The following commands set up LSM for the first time:

```
# volsetup rz10 rz12 rz32 rz34
Approximate maximum number of physical disks .. by LSM ? [10] 50
     :
Initialization of vold and the rootdg disk group was successful.
```

At this point, the disks `rz10`, `rz12`, `rz32`, and `rz34` have been added to LSM and LSM has been initialized and is ready to use.

The following commands display various configuration information. The first command checks all LSM disk status:

```
# voldisk list
DEVICE  TYPE    DISK   GROUP    STATUS
rz10   sliced  rz10   rootdg   online
rz12   sliced  rz12   rootdg   online
rz32   sliced  rz32   rootdg   online
rz34   sliced  rz34   rootdg   online
```

The following command checks the status of the `rootdg` diskgroup:

```
# volprint -ht
DG NAME  GROUP-ID
DM NAME  DEVICE   TYPE    PRIVLEN  PUBLEN  PUBPATH
V  NAME  USETYPE  KSTATE  STATE    LENGTH  READPOL  ...
PL NAME  VOLUME   KSTATE  STATE    LENGTH  LAYOUT   ...
SD NAME  PLEX     PLOFFS  DISKOFFS LENGTH  DISK-NAME...

dgrootdg       784429068.1025.rio.zk3.dec.com

dm rz10   rz10    sliced   1024   2049820  /dev/rrz10g
dm rz12   rz12    sliced   1024   2049820  /dev/rrz12g
dm rz32   rz32    sliced   1024   2049820  /dev/rrz32g
dm rz34   rz34    sliced   1024   2049820  /dev/rrz34g
```

The following command checks the status of `/etc/vol/volboot`:

```
# voldctl list
Volboot file
version: 3/1
seqno:   0.6
hostid:  rio.zk3.dec.com
entries:
  disk rz10 type=sliced
  disk rz12 type=sliced
  disk rz32 type=sliced
  disk rz34 type=sliced
```

The following command checks the status of `vold`:

```
# voldctl mode
mode: enabled
```

C.2 Adding New Disks to LSM (Without Existing Data)

The following examples show how to add additional disks to LSM. Refer to Chapter 3, `voldiskadd`(8), `voldisksetup`(8), `voldisk`(8), `voldg`(8), and `voldctl`(8) for further information on adding disks under LSM.

Most of the following examples assume no data was already existing on the disk. See Section C.4 for encapsulation examples that add disks already containing data. If only certain partitions on the disk are free, those partitions can be added to LSM, as shown in the following examples.

The following example initializes and adds an entire disk to the `rootdg` disk group:

```
# voldiskadd rz17
Which disk group [<group>,none,list,q,?] (default: rootdg) <return>
     :
Enter disk name [<name>,q,?] (default: disk01) <return>
     :
Continue with operation? [y,n,q,?] (default: y) <return>
     :
Add rz17 to /etc/vol/volboot file ? [y,n,q,?] (default: y) <return>
```

If the entire disk is not free, but certain partitions are free, then you can add the free partitions to LSM. The following example shows the commands to initialize and add free partitions to `rootdg`.

```
# voldiskadd rz18h


Which disk group [<group>,none,list,q,?] (default: rootdg) <return>
     :
Enter disk name [<name>,q,?] (default: disk02) rz18h
     :
Continue with operation? [y,n,q,?] (default: y) <return>
     :
Add rz18h to /etc/vol/volboot file ? [y,n,q,?] (default: y) n
```

The following examples initialize the disk with the `voldisksetup` command:

```
# voldisksetup -i rz19 privlen=1024 nconfig=1 nlogs=1
```

The following command adds rz19 as disk02 to `rootdg`:

```
# voldg adddisk disk02=rz19
```

The following command initializes `rz18g`:

```
# voldisksetup -i rz18g privlen=1024 nconfig=1 nlogs=1
```

The following command adds `rz18g` to the `rootdg` disk group with disk media name `rz18g`:

```
# voldg adddisk rz18g=rz18g
```

Note that four to eight copies of a disk group's configuration on different disks or controllers is adequate. If you want to add more disks to a disk group, add them without a configuration database and kernel log.

The following example sequence adds more disks:

```
# voldisksetup -i rz21 privlen=1024 nconfig=0 nlogs=0
# voldisksetup -i rz36 privlen=1024 nconfig=0 nlogs=0
# voldg adddisk rz21
# voldg adddisk rz36
```

Note

> To add a disk with no configuration database to a disk group, set the `nconfig` attribute to 0 when initializing the disk with `voldisksetup`, as shown in the previous example. Do not initialize a new disk as a nopriv disk – this disk type is appropriate only for encapsulation of existing data.

C.3 Adding /usr (UFS) Under LSM

The following example shows how to add a partition that contains existing data (for example, the `/usr` file system) under LSM using the encapsulation utility. Refer to Chapter 4 and the `volencap`(8) reference page for further information.

The following command determines which partition has `/usr`:

```
# mount
/dev/rz3a on / type ufs (rw)
/proc on /proc type procfs (rw)
/dev/rz3g on /usr type ufs (rw)
```

The following command moves `/usr` under LSM:

```
# volencap rz3g
  The rz3g disk has been configured for encapsulation.
```

The following example shuts down the system in order for the change to take effect. After the system is rebooted, `/usr` will be under LSM.

```
# shutdown -r now
```

The following commands shows how the changes are made:

```
# cat /etc/fstab | grep vol
/dev/vol/rootdg/vol-rz3g        /usr    ufs rw 1 2
# mount
/dev/rz3a on / type ufs (rw)
/proc on /proc type procfs (rw)
/dev/vol/rootdg/vol-rz3g on /usr type ufs (rw)
# volprint -htv
V  NAME       USETYPE     KSTATE   STATE    LENGTH READPOL  ...
PL NAME       VOLUME      KSTATE   STATE    LENGTH LAYOUT   ...
SD NAME       PLEX        PLOFFS   DISKOFFS LENGTH DISK-NAME...

v  vol-rz3g    fsgen       ENABLED ACTIVE   819200 SELECT   ...
pl vol-rz3g-01 vol-rz3g    ENABLED ACTIVE   819200 CONCAT   ...
sd rz3g-01     vol-rz3g-01 0       0        819200 rz3g     ...
#
```

C.4 Adding Existing Data Under LSM

The following example shows an alternative way to add a partition containing data (for example, a UFS file system) under LSM by using the LSM commands. Refer to Chapter 3 and the `voldisk`(8), `voldg`(8), `volmake`(8), and `volume`(8) for more information.

The following command checks the disk label for UFS:

```
# disklabel rz8
     size   offset  fstype [fsize bsize cpg]
a:  131072       0  unused  1024  8192      # (Cyl.    0 - 164*)
b:  262144  131072  unused  1024  8192      # (Cyl.  164*- 492*)
c: 2050860       0  unused  1024  8192      # (Cyl.    0 - 2569)
d:  552548  393216  4.2BSD  1024  8192  16  # (Cyl.  492*- 1185*)
e:  552548  945764  unused  1024  8192      # (Cyl. 1185*- 1877*)
f:  552548 1498312  unused  1024  8192      # (Cyl. 1877*- 2569*)
g:  819200  393216  unused  1024  8192      # (Cyl.  492*- 1519*)
h:  838444 1212416  unused  1024  8192      # (Cyl. 1519*- 2569*)
```

The following command indicates that rz8d is not mounted:

```
# mount
/dev/rz3a on / type ufs (rw)
/proc on /proc type procfs (rw)
/dev/vol/rootdg/vol-rz3g on /usr type ufs (rw)
```

The following command adds rz8d with no private region:

```
# voldisk init rz8d type=nopriv
```

The following command adds rz8d to the `rootdg` disk group:

```
# voldg adddisk rz8d
```

You can use the `volmake help` command whenever you need help remembering the options:

```
# volmake help
```

The following command creates a subdisk:

```
# volmake sd ov-sd rz8d,0,552548s
```

The following command creates the plex and associates it to the subdisk:

```
# volmake plex ov-pl sd=ov-sd
```

The following command creates volume and attaches plexes:

```
# volmake vol ov-vol plex=ov-pl usetype=fsgen
```

The following command starts the volume:

```
# volume start ov-vol
```

The following command checks the results:

```
# volprint -ht ov-vol
DG NAME     GROUP-ID
DM NAME     DEVICE       TYPE     PRIVLEN  PUBLEN   PUBPATH
V  NAME     USETYPE      KSTATE   STATE    LENGTH   READPOL  ...
PL NAME     VOLUME       KSTATE   STATE    LENGTH   LAYOUT   ...
SD NAME     PLEX         PLOFFS   DISKOFFS LENGTH   DISK-NAME...

v  ov-vol   fsgen        ENABLED  ACTIVE   552548   ROUND    ...
pl ov-pl    ov-vol       ENABLED  ACTIVE   552548   CONCAT   ...
sdov-sd    ov-pl        0        0        552548   rz8d      ...
```

The following command mounts the volume:

```
# mount /dev/vol/ov-vol /usr/OV
```

The following command can be used to edit the `/etc/fstab` file and add an entry to automatically mount the volume:

```
# vi /etc/fstab
```

C.5 Creating LSM Volumes

The following examples show some different ways to create LSM volumes. Refer to Chapter 6 and the `voldisk`(8), `voldg`(8), `volmake`(8), and `volume`(8) reference pages for more information.

The following examples show how to use the LSM commands for a top-down approach to creating volumes. The first command shows how to create a volume of 100 Mb on disk01:

```
# volassist make myvol1 100m disk01
```

The following command creates a volume anywhere but on disk01:

```
# volassist make myvol2 100m !disk01
```

The following command creates a volume anywhere:

```
# volassist make myvol3 1g
```

The following command displays the results of the configuration changes:

```
# volprint -htv myvol1 myvol2 myvol3
V  NAME        USETYPE    KSTATE  STATE    LENGTH  READPOL ...
PL NAME        VOLUME     KSTATE  STATE    LENGTH  LAYOUT  ...
SD NAME        PLEX       PLOFFS  DISKOFFS LENGTH  DISK-NAME...

v  myvol1      fsgen      ENABLED ACTIVE   204800  SELECT  -
pl myvol1-01   myvol1     ENABLED ACTIVE   204800  CONCAT  -
sd disk01-01   myvol1-01  0       0        204800  disk01

v  myvol2      fsgen      ENABLED ACTIVE   204800  SELECT  -
pl myvol2-01   myvol2     ENABLED ACTIVE   204800  CONCAT  -
sd disk02-01   myvol2-01  0       0        204800  disk02

v  myvol3      fsgen      ENABLED ACTIVE   2097152 SELECT  -
pl myvol3-01   myvol3     ENABLED ACTIVE   2097152 CONCAT  -
sd disk01-02   myvol3-01  0       204800   1845020 disk01
sd disk02-02   myvol3-01  1845020 204800   252132  disk02
```

The following command puts a UFS on a volume called myvol1:

```
# newfs /dev/rvo1/rootdg/myvol1 rz26l
```

The following command mounts the volume:

```
# mount /dev/vol/rootdg/myvol1 /mnt8
```

The following shows how to use LSM commands for a bottom up approach to creating volumes. This method provides more control when setting up an LSM environment.

The following command looks for free space in the `rootdg` disk group:

```
# voldg -g rootdg free
```

```
GROUP    DISK    DEVICE   TAG   OFFSET  LENGTH   FLAGS
rootdg   rz10    rz10     rz10  0       2049820  -
rootdg   rz12    rz12     rz12  0       2049820  -
rootdg   disk02  rz19     rz19  456932  1592888  -
rootdg   rz21    rz21     rz21  0       2049820  -
rootdg   rz32    rz32     rz32  0       2049820  -
rootdg   rz34    rz34     rz34  0       2049820  -
rootdg   rz36    rz36     rz36  0       2049820  -
```

The following command creates a subdisk:

```
# volmake sd v4-sd disk02,456932,100m
```

The following command creates a plex:

```
# volmake plex v4-pl sd=v4-sd
```

The following command creates the volume and attaches plexes:

```
# volmake vol v4 plex=v4-pl usetype=fsgen
```

The following command starts the volume:

```
# volume start v4
```

The following command checks the results:

```
# volprint -ht v4
DG NAME    GROUP-ID
DM NAME    DEVICE  TYPE    PRIVLEN  PUBLEN  PUBPATH
V  NAME    USETYPE KSTATE  STATE    LENGTH  READPOL  ...
PL NAME    VOLUME  KSTATE  STATE    LENGTH  LAYOUT   ...
SD NAME    PLEX    PLOFFS  DISKOFFS LENGTH  DISK-NAME...

v  v4      fsgen   ENABLED ACTIVE   204800  ROUND    -
pl v4-pl   v4      ENABLED ACTIVE   204800  CONCAT   -
sd v4-sd   v4-pl   0       456932   204800  disk02
#
```

C.6 Mirroring LSM Volumes

The following examples show some different ways to mirror LSM volumes. Refer to Chapter 7, Chapter 15, and the `volassist`(8), `volmake`(8), `volsd`(8), `volplex`(8), and `volume`(8) reference pages for more information.

The commands in this section show how to use LSM commands for a top-down approach to mirroring LSM volumes. Note that this can be done while the volume is in use:

```
# mount | grep mnt8
/dev/vol/myvol1 on /mnt8 type ufs (rw)
```

The following command mirrors myvol1 using any available disk:

```
# volassist mirror myvol1
```

The following command mirrors the myvol2 volume on the rz12 disk:

```
# volassist mirror myvol2 rz12
```

The following command creates a 50Mb mirrored volume:

```
# volassist -U fsgen make v2_mirr 50m nmirror=2
# volprint -h myvol1 myvol2 v2_mirr
TYPE NAME         ASSOC        KSTATE     LENGTH COMMENT
vol  myvol1       fsgen        ENABLED    204800
plex myvol1-01    myvol1       ENABLED    204800
sd   disk01-01    myvol1-01    -          204800
plex myvol1-02    myvol1       ENABLED    204800
sd   disk02-04    myvol1-02    -          204800

vol  myvol2       fsgen        ENABLED    204800
plex myvol2-01    myvol2       ENABLED    204800
sd   disk02-01    myvol2-01    -          204800
plex myvol2-02    myvol2       ENABLED    204800
sd   rz12-02      myvol2-02    -          204800

vol  v2_mirr      fsgen        ENABLED    102400
plex v2_mirr-01   v2_mirr      ENABLED    102400
sd   disk02-05    v2_mirr-01   -          102400
plex v2_mirr-02   v2_mirr      ENABLED    102400
sd   rz10-02      v2_mirr-02   -          102400
#
```

The following command creates a new AdvFS domain:

```
# mkfdmn /dev/vol/rootdg/myvol1 dom1
```

The following command creates a new fileset:

```
# mkfset dom1 fset1
```

The following command mounts the fileset:

```
# mount -t advfs dom1#fset1 /mnt9
```

On systems that have the AdvFS Advanced Utilities package installed, the following command adds a second volume to the AdvFS domain:

```
# addvol /dev/vol/rootdg/myvol2 dom1
```

The following series of commands demonstrate how to use LSM commands for a bottom-up approach to creating a new, mirrored volume:

The following command creates a subdisk:

```
# volmake sd sd1 rz32,0,30m
```

The following command creates a plex and associates a subdisk with the plex:

```
# volmake plex pl1 sd=sd1
```

The following command creates a volume:

```
# volmake -U fsgen vol v_mir2 plex=pl1
```

The following command starts the volume:

```
# volume start v_mir2
```

The following command creates the second subdisk (sd2):

```
# volmake sd sd2 rz34,0,30m
```

The following command creates the second plex (pl2):

```
# volmake plex pl2
```

The following command associates pl2 with sd2:

```
# volsd assoc pl2 sd2
```

The following command attaches the plex pl2 with volume v_mir2:

```
# volplex att v_mir2 pl2
```

The following command displays the results of these commands:

```
# volprint -ht v_mir2
DG NAME    GROUP-ID
DM NAME    DEVICE   TYPE     PRIVLEN  PUBLEN  PUBPATH
V  NAME    USETYPE  KSTATE   STATE    LENGTH  READPOL  ...
PL NAME    VOLUME   KSTATE   STATE    LENGTH  LAYOUT   ...
SD NAME    PLEX     PLOFFS   DISKOFFS LENGTH  DISK-NAME...

v  v_mir2  fsgen    ENABLED  ACTIVE   61440   ROUND    -
pl pl1     v_mir2   ENABLED  ACTIVE   61440   CONCAT   -
sd sd1     pl1      0        0        61440    rz32
pl pl2     v_mir2   ENABLED  ACTIVE   61440   CONCAT   -
sd sd2     pl2      0        0        61440   rz34
#
```

C.7 Creating Striped LSM Volumes

The following examples show different ways to stripe data with an LSM volume. Refer to Chapter 7 and the `volassist`(8), `volmake`(8), `volsd`(8), `volplex`(8), and `volume`(8) reference pages for more information.

The following LSM commands demonstrate a top-down approach to striping data across LSM disks.

```
# volassist make v1_stripe 64m usetype=fsgen layout=stripe \
nstripe=4 stwidth=8k
# volprint -ht v1_stripe

DG NAME     GROUP-ID
DM NAME     DEVICE  TYPE   PRIVLEN  PUBLEN PUBPATH
V  NAME     USETYPE KSTATE STATE    LENGTH READPOL ...
PL NAME     VOLUME  KSTATE STATE    LENGTH LAYOUT  ...
SD NAME     PLEX    PLOFFS DISKOFFS LENGTH DISK-NAME...

v  v1_stripe   fsgen       ENABLED  ACTIVE 131072  SELECT...
pl v1_stripe-01 v1_stripe  ENABLED  ACTIVE 131072  STRIPE 16
sd disk02-03  v1_stripe-01 0        661732 32768   disk02
sd rz10-01    v1_stripe-01 32768    0      32768   rz10
sd rz12-01    v1_stripe-01 65536    0      32768   rz12
sd rz32-01    v1_stripe-01 98304    61440  32768   rz32
#
```

The following LSM commands demonstrate a bottom-up approach to creating a new, mirrored volume. The following command creates a subdisk:

```
# volmake sd s1-sd rz21,0,500m
```

The following command creates the second subdisk:

```
# volmake sd s2-sd rz36,0,500m
```

The following command creates a plex:

```
# volmake plex s-pl sd=s1-sd,s2-sd layout=stripe stwidth=16k
```

The following command creates the volume:

```
# volmake -U gen vol my_fast_one plex=s-pl
```

The following command starts the volume:

```
# volume start my_fast_one
```

The following command displays the results of the configuration changes:

```
# volprint -ht my_fast_one

DG NAME         GROUP-ID DM NAME
DEVICE       TYPE     PRIVLEN  PUBLEN   PUBPATH V  NAME...
KSTATE   STATE    LENGTH   READPOL  PREFPLEX PL NAME   ...
KSTATE   STATE    LENGTH   LAYOUT   ST-WIDTH MODE SD NAME...
PLOFFS   DISKOFFS LENGTH   DISK-NAME    DEVICE

v  my_fast_one  fsgen        ENABLED  ACTIVE   2048000  ROUND...
s-pl         my_fast_one  ENABLED  ACTIVE   2048000  STRIPE  ...
sd s1-sd        s-pl         0        0        1024000  rz21 ...
sd s2-sd        s-pl         1024000  0        1024000  rz36 ...
```

The following commands create a UFS file system on the volume and mount it:

```
# newfs /dev/vol/my_fast_one rz26l
# mount /dev/vol/my_fast_one /fast1
# df /fast1
Filesystem         512-blocks     Used       Avail Capacity...
/dev/vol/my_fast_one  1980986        2     1782884     0%  ...
#
```

C.8 Setting Attributes for a Volume

Use the LSM `voledit` command to set or change the owner, group, or mode of the special device file for a volume. Do not use standard UNIX commands such as `chown`, `chgrp`, or `chmod` to set or change LSM special device file attributes .

For example, the following `voledit` command changes the user and group to `dba` and the mode to 0600 for the volume `vol_db` in disk group `dbgrp`:

```
# voledit -g dbgrp set user=dba group=dba mode=0600 vol_db
```

Refer to the `voledit`(8) reference page for further information.

C.9 Moving Subdisks From a Disk

When a disk is getting large number of soft errors, you should move all subdisks on that disk to other disks. The free disk space in the disk group must be larger than that of the disk being evacuated.

Enter the following command to determine the free space in the disk group.

```
# voldg free
GROUP      DISK     DEVICE   TAG    OFFSET
LENGTH  FLAGS rootdg   rz10  rz10    rz10  0
32768   - rootdg    rz10     rz10    rz10  135168
1914652 - rootdg    rz12     rz12    rz12  0
32768   - rootdg    rz12     rz12    rz12  270336
1779484 - rootdg    disk02   rz19    rz19  661732
32768   - rootdg    disk02   rz19    rz19  1034468
1015352 - rootdg    rz21     rz21    rz21  1179648
870172  - rootdg    rz32     rz32    rz32  61440
1988380 - rootdg    rz34     rz34    rz34  61440
1988380 - rootdg    rz36     rz36    rz36  1159168
890652  -
```

Use the `volevac` command to move all volumes from a particular disk to another available disk. The following example shows that the LSM disk `rz32` is used by volume `v_mir2`:

```
# volprint -htv v_mir2

V  NAME   USETYPE  KSTATE   STATE    LENGTH  READPOL  ...
PL NAME   VOLUME   KSTATE   STATE    LENGTH  LAYOUT   ...
SD NAME   PLEX     PLOFFS   DISKOFFS LENGTH  DISK-NAME...

v  v_mir2 fsgen    ENABLED  ACTIVE   61440   ROUND    -
pl pl1    v_mir2   ENABLED  ACTIVE   61440   CONCAT   -
sd sd1    pl1      0        0        61440   rz32
pl pl2    v_mir2   ENABLED  ACTIVE   61440   CONCAT   -
sd sd2    pl2      0        0        61440   rz34
```

To move data from `rz32`, enter the following command:

```
# volevac rz32
```

This command can take a long time to finish.

The following command displays the results of the changes:

```
# volprint -htv v_mir2

V  NAME    USETYPE  KSTATE   STATE    LENGTH  READPOL  ...
PL NAME    VOLUME   KSTATE   STATE    LENGTH  LAYOUT   ...
SD NAME    PLEX     PLOFFS   DISKOFFS LENGTH  DISK-NAME...

v  v_mir2  fsgen    ENABLED  ACTIVE   61440   ROUND    -
pl pl1     v_mir2   ENABLED  ACTIVE   61440   CONCAT   -
sd rz21-02 pl1      0        1056768  61440   rz21
pl pl2     v_mir2   ENABLED  ACTIVE   61440   CONCAT   -
sd sd2     pl2      0        0        61440   rz34
```

C.10 Replacing a Disk

If a disk that was in use by LSM fails to restart or has other hardware problems, you can replace the disk with a new disk. The following sections describe the procedure you use to replace a disk. The procedure varies depending on whether the replacement disk has the same or a different physical unit number than the failed disk.

C.10.1 Replacing a Disk That Has a Different Unit Number

The examples in this section describe how to replace a disk that has a different unit number from that of the failed disk.

The example that follows is for disk `rz19`, which has hardware problems and needs to be replaced. Disk `rz19` has been added to the `rootdg` disk group as `disk02`. Follow these steps:

1. Remove the LSM disk, `disk02`, from the `rootdg` disk group. To do this, use the `-k` flag with the `voldg` command to keep the disk media records associated with `disk02`. The subdisk records associated with `disk02` will continue to point to the disk media record. For example:

   ```
   # voldg -g rootdg -k rmdisk disk02
   ```

2. If you have a spare disk available that is at least as large as the `rz19` disk (for example, `rz20`), you can use the spare disk to replace the failed disk. Initialize the new `rz20` disk using similar parameters to those used for `rz19`. For example:

   ```
   # voldisksetup -i rz20 privlen=1024 nconfig=1 nlogs=1
   ```

3. Add the new `rz20` disk to the `rootdg` disk group. Use the `-k` flag to associate a new disk with the existing disk media records in the disk group.

   ```
   # voldg -g rootdg -k adddisk disk02=rz20
   ```

4. If the failed disk `rz19` has any volumes with only one plex, you must restore data to the volumes from backup media. If the volumes using `rz19` are mirrored, resynchronize the volumes by issuing the following command:

   ```
   # volrecover -sb disk02
   ```

C.10.2 Replacing with a Same Unit Number

If a spare disk is available but has to replace the failed disk at the same unit number, follow these steps:

1. Remove the failed disk `rz19` from LSM:

   ```
   # voldg -g rootdg -k rmdisk disk02
   ```

2. Replace the new disk at the same unit number:

   ```
   # voldisk rm rz19
   ```

3. Initialize the new disk. Make sure that a disk label has been initialize on the new disk.

   ```
   # voldisksetup -i rz19 privlen=1024 nconfig=1 nlogs=1
   ```

4. Add the new `rz19` disk to the `rootdg` disk group. Use the `-k` flag to associate a new disk with the existing disk media records in the disk group:

   ```
   # voldg -g rootdg -k adddisk disk02=rz19
   ```

5. Recover the data on any mirrored volume using `disk02`:

   ```
   # volrecover -sb disk02
   ```

6. If the failed disk `rz19` was used by an unmirrored volume, restore the data from backup.

C.11 Removing Volumes

To remove LSM volumes, first ensure that they are not in use. If the LSM volume is in use, the `voledit` command will fail. Refer to `voledit`(8) for more details.

To remove the volume called `v_mir2` from the `rootdg` disk group, enter the following command:

```
# voledit -g rootdg -rf rm v_mir2
```

C.12 Removing a Disk from LSM

To remove a disk from LSM, first ensure that the disk is not in use by any subdisk. If the disk is in use, refer to Section C.9 for information about how to evacuate a disk.

Follow these steps to remove a disk from LSM:

1. Get the list of all disks added to LSM by entering the following

command:

```
# voldisk list
DEVICE       TYPE      DISK         GROUP        STATUS
rz10         sliced    rz10         rootdg       online
rz12         sliced    rz12         rootdg       online
rz17         sliced    disk01       rootdg       online
rz18g        simple    rz18g        rootdg       online
rz18h        simple    rz18h        rootdg       online
rz19         sliced    disk02       rootdg       online
rz21         sliced    rz21         rootdg       online
rz32         sliced    disk03       rootdg       online
rz34         sliced    rz34         rootdg       online
rz36         sliced    rz36         rootdg       online
rz8d         nopriv    rz8d         rootdg       online
```

2. Remove the LSM disk `disk03` from the `rootdg` disk group. For example:

```
# voldg -g rootdg rmdisk disk03
```

3. Use the `voldisk` command to note the change in status for `rz32`. For example:

```
# voldisk list
DEVICE TYPE      DISK         GROUP        STATUS
rz10   sliced    rz10         rootdg       online
rz12   sliced    rz12         rootdg       online
rz17   sliced    disk01       rootdg       online
rz18g  simple    rz18g        rootdg       online
rz18h  simple    rz18h        rootdg       online
rz19   sliced    disk02       rootdg       online
rz21   sliced    rz21         rootdg       online
rz32   sliced    -              -          online
rz34   sliced    rz34         rootdg       online
rz36   sliced    rz36         rootdg       online
rz8d   nopriv    rz8d         rootdg       online
```

4. Remove disk `rz32` from LSM:

```
# voldisk rm rz32
```

5. Get the list of disks known to LSM by entering the following command. Note that `rz32` is not among the disks listed.

```
# voldisk list
DEVICE       TYPE       DISK         GROUP        STATUS

rz10         sliced     rz10         rootdg       online
rz12         sliced     rz12         rootdg       online
rz17         sliced     disk01       rootdg       online
rz18g        simple     rz18g        rootdg       online
rz18h        simple     rz18h        rootdg       online
rz19         sliced     disk02       rootdg       online
rz21         sliced     rz21         rootdg       online
rz34         sliced     rz34         rootdg       online
```

```
rz36          sliced     rz36          rootdg        online
rz8d          nopriv     rz8d          rootdg        online
```

To remove the last disk of a disk group other than `rootdg`, you must deport the disk group. For example:

```
# voldg deport dg1
```

After deporting the disk group, You can use the `voldisk rm` command to remove the last disk in that disk group from LSM.

To remove the last disk in the `rootdg` disk group, you must shut down the LSM configuration daemon, `vold`. To stop `vold`, enter the following command:

```
# voldctl stop
```

After shutting down `vold`, the last disk is no longer in use by LSM.

C.13 Moving LSM Disks Between Disk Groups

You can use the `volmake` description file to save and re-create the volume configuration when disks are moved from one diskgroup to another.

For example, the following steps show how to move disks `rz16` and `rz17` (with disk names `disk01` and `disk02`, respectively) from disk group `dg1` to disk group `dg2.` The disks must have the same disk names in disk group `dg2` as they had in disk group `dg1`.

1. Before doing the LSM reconfiguration, ensure that the data on the affected volumes is properly backed up.
2. Determine which volumes will be affected when the disks are moved from disk group `dg1`.

   ```
   # volprint -g dg1 -vn -e "aslist.aslist.sd_da_name==\"rz16\""
   vol1
   vol2
   # volprint -g dg1 -vn -e "aslist.aslist.sd_da_name==\"rz17\""
   vol3
   vol4
   ```

3. Create a `volmake` description file for the affected volumes.

   ```
   # volprint -g dg1 -mh vol1 vol2 vol3 vol4 > tmp1.df
   ```

   This command creates a description file for volumes `vol1`, `vol2`, `vol3`, and `vol4` in disk group `dg1.`
4. If all the disks are being moved from disk group `dg2`, deport the disk

group dg1.

```
# voldg deport dg1
```

If only some of the disks are being moved, remove the volumes from disk group dg1, and then remove the disks from disk group dg1.

```
# voledit -g dg1 -rf rm vo1l vol2 vol3 vol4
# voldg -g dg1 rmdisk disk01 disk02
```

5. Initialize a new disk group, dg2. For example, the following command initializes a new disk group dg2 with disk rz16. The disk is named disk01, since this was its name in disk group dg1.

   ```
   # voldg init dg2 disk01=rz16
   ```

6. Add disk rz17 to the disk group dg2.

   ```
   # voldg -g dg2 adddisk dg2 disk02=rz17
   ```

7. Re-create the LSM configuration for volumes vol1, vol2, vol3, and vol4.

   ```
   # volmake -g dg2 -d tmp1.df
   ```

8. Set the plex states appropriately using the volmend command. Then start the volumes in disk group dg2.

   ```
   # volume -g dg2 start vol1 vol2 vol3 vol4
   ```

   Note

   If a plex was in the STALE state make sure that the plex state is set to STALE before starting the volume.

9. After re-creating the volumes in disk group dg2, modify the /etc/fstab file or the /etc/fdmns directory to use the appropriate special device file.

C.14 Initializing LSM and Encapsulating the Boot Disk

To initialize LSM and encapsulate all of the partitions on the boot disk, run the volencap utility, using the name of the system boot disk as an argument.

In the following example, `rz3` is the boot disk and has the following label:

```
# /dev/rrz3a:
type: SCSI
disk: RZ26
label:
flags:
bytes/sector: 512
sectors/track: 57
tracks/cylinder: 14
sectors/cylinder: 798
cylinders: 2570
sectors/unit: 2050860
rpm: 3600
interleave: 1
trackskew: 0
cylinderskew: 0
headswitch: 0           # milliseconds
track-to-track seek: 0  # milliseconds
drivedata: 0

8 partitions:
#       size   offset   fstype  [fsize bsize  cpg]
 a:   131072        0    AdvFS                      # (Cyl.    0 - 164*)
 b:   262144   131072     swap                      # (Cyl.  164*- 492*)
 c:  2050860        0   unused     0     0          # (Cyl.    0 - 2569)
 d:   552548   393216   unused     0     0          # (Cyl.  492*- 1185*)
 e:   552548   945764   unused     0     0          # (Cyl. 1185*- 1877*)
 f:   552548  1498312   unused     0     0          # (Cyl. 1877*- 2569)
 g:   819200   393216   AdvFS                       # (Cyl.  492*- 1519*)
 h:   838444  1212416   unused     0     0          # (Cyl. 1519*- 2569)
```

Follow these steps:

1. Initialize LSM and add all partitions that are in use as LSM volumes:

   ```
   # volencap rz3

   Setting up encapsulation for rz3.
   - Disk rz3 is the system boot disk and LSM is not initialized.
     Creating simple disk rz3d to initialize LSM and rootdg.
   - Partition rz3a is the root partition which requires 2 passes
     to encapsulate and the temporary use of a free partition.
     Using partition rz3e for temporary root encapsulation.
   - Creating nopriv disk for primary swap device rz3b.
   - Creating nopriv disk for rz3g.

   The following disks are queued up for encapsulation or use by LSM.
   You must reboot the system to perform the actual encapsulations.
       rz3d rz3a rz3e rz3b rz3g
   ```

2. Reboot the system to execute the encapsulation scripts. Note that this process takes three passes to complete. The system is automatically

rebooted for each pass.

```
# shutdown -r now

ADVFS: using 566 buffers containing 4.42 megabytes of memory
starting LSM
LSM: /sbin/swapdefault has been moved to /sbin/swapdefault.encap.
LSM: Rebooting system to initialize LSM.
syncing disks... 3 done
rebooting.... (transferring to monitor)

ADVFS: using 566 buffers containing 4.42 megabytes of memory
vm_swap_init: warning /sbin/swapdefault swap device not found
vm_swap_init: swap is set to lazy (over commitment) mode
starting LSM
LSM: Initializing rz3d.
LSM: Encapsulating first pass for root using rz3e.
LSM: Encapsulating primary swap device rz3b.
LSM: Encapsulating rz3g.
LSM:
LSM: The following disks were encapsulated successfully:
LSM: rz3b rz3g
LSM:
LSM: The following disks are queued for encapsulation at reboot:
LSM: rz3a

The system is rebooting for the following reason(s):
 Second pass root encapsulation to move rootvol from rz3e to rz3a.
 To enable swapvol.

syncing disks... 2 done
rebooting.... (transferring to monitor)

ADVFS: using 566 buffers containing 4.42 megabytes of memory
starting LSM
LSM: Moving root volume from rz3e to rz3a.
LSM:   This may take a few minutes.
LSM:
LSM: The following disks were encapsulated successfully:
LSM: rz3a
Checking local filesystems
/sbin/ufs_fsck -p
```

The following listing shows the configuration that results from this procedure:

```
# volprint -ht

DG NAME         GROUP-ID
DM NAME         DEVICE       TYPE     PRIVLEN  PUBLEN   PUBPATH
V  NAME         USETYPE      KSTATE   STATE    LENGTH   READPOL  PREFPLEX
PL NAME         VOLUME       KSTATE   STATE    LENGTH   LAYOUT   ST-WIDTH
MODE
SD NAME         PLEX         PLOFFS   DISKOFFS LENGTH   DISK-NAME  DEVICE

dg rootdg       821915478.1025.lsmtest

dm rz3a         rz3a         nopriv   0        131072  /dev/rrz3a
dm rz3b         rz3b         nopriv   0        261120  /dev/rrz3b
dm rz3d         rz3d         simple   1024     0       /dev/rrz3d
```

```
dm rz3g         rz3g         nopriv   0        819200  /dev/rrz3g

v  rootvol      root         ENABLED  ACTIVE   131072  ROUND  -
pl rootvol-01   rootvol      ENABLED  ACTIVE   131072  CONCAT -    RW
sd rz3a-01      rootvol-01   0        0        131072  rz3a      rz3a

v  swapvol      swap         ENABLED  ACTIVE   261120  ROUND  -
pl swapvol-01   swapvol      ENABLED  ACTIVE   261120  CONCAT -    RW
sd rz3b-01      swapvol-01   0        0        261120  rz3b      rz3b

v  vol-rz3g     fsgen        ENABLED  ACTIVE   819200  SELECT -
pl vol-rz3g-01  vol-rz3g     ENABLED  ACTIVE   819200  CONCAT -    RW
sd rz3g-01      vol-rz3g-01  0        0        819200  rz3g      rz3g
```

C.15 Encapsulating the Root and Swap Partitions

This example shows how to encapsulate the root and swap partitions when LSM is already initialized. The example is based on the LSM configuration shown in the following listing:

```
# volprint -ht

DG NAME      GROUP-ID
DM NAME      DEVICE    TYPE     PRIVLEN  PUBLEN   PUBPATH
V  NAME      USETYPE   KSTATE   STATE    LENGTH   READPOL  PREFPLEX
PL NAME      VOLUME    KSTATE   STATE    LENGTH   LAYOUT   ST-WIDTH
MODE
SD NAME      PLEX      PLOFFS   DISKOFFS LENGTH   DISK-NAME  DEVICE

dg rootdg       821984014.1025.lsmtest

dm rz3d         rz3d         simple   1024     0        /dev/rrz3d
```

To encapsulate the root and swap partitions, run the `volencap` utility with the partition names as arguments. For example:

```
# volencap rz3a rz3b

Setting up encapsulation for rz3a.
- Partition rz3a is the root partition which requires 2 passes
  to encapsulate and the temporary use of a free partition.
  Using partition rz3e for temporary root encapsulation.

Setting up encapsulation for rz3b.
- Creating nopriv disk for primary swap device rz3b.

The following disks are queued up for encapsulation or use by LSM.
You must reboot the system to perform the actual encapsulations.
    rz3a rz3e rz3b

# shutdown -r now

ADVFS: using 566 buffers containing 4.42 megabytes of memory
vm_swap_init: warning /sbin/swapdefault swap device not found
vm_swap_init: swap is set to lazy (over commitment) mode
starting LSM
LSM: Encapsulating first pass for root using rz3e.
LSM: Encapsulating primary swap device rz3b.
```

```
LSM:
LSM: The following disks were encapsulated successfully:
LSM: rz3b
LSM:
LSM: The following disks are queued for encapsulation at reboot:
LSM: rz3a

The system is rebooting for the following reason(s):
 Second pass root encapsulation to move rootvol from rz3e to rz3a.
 To enable swapvol.

syncing disks... 2 done
rebooting.... (transferring to monitor)
ADVFS: using 566 buffers containing 4.42 megabytes of memory
starting LSM
LSM: Moving root volume from rz3e to rz3a.
LSM:   This may take a few minutes.
LSM:
LSM: The following disks were encapsulated successfully:
LSM: rz3a
Checking local filesystems
/sbin/ufs_fsck -p
```

volprint -ht

```
DG NAME         GROUP-ID
DM NAME         DEVICE       TYPE     PRIVLEN  PUBLEN   PUBPATH
V  NAME         USETYPE      KSTATE   STATE    LENGTH   READPOL  PREFPLEX
PL NAME         VOLUME       KSTATE   STATE    LENGTH   LAYOUT   ST-WIDTH
MODE
SD NAME         PLEX         PLOFFS   DISKOFFS LENGTH   DISK-NAME    DEVICE

dg rootdg       821984014.1025.lsmtest

dm rz3a         rz3a         nopriv   0        131072   /dev/rrz3a
dm rz3b         rz3b         nopriv   0        261120   /dev/rrz3b
dm rz3d         rz3d         simple   1024     0        /dev/rrz3d

v  rootvol      root         ENABLED  ACTIVE   131072   ROUND    -
pl rootvol-01   rootvol      ENABLED  ACTIVE   131072   CONCAT   -        RW
sd rz3a-01      rootvol-01   0        0        131072   rz3a         rz3a

v  swapvol      swap         ENABLED  ACTIVE   261120   ROUND    -
pl swapvol-01   swapvol      ENABLED  ACTIVE   261120   CONCAT   -        RW
sd rz3b-01      swapvol-01   0        0        261120   rz3b         rz3b
```

Note

Although it is possible to encapsulate the root and swap partitions individually, encapsulating only root or swap is not a supported configuration.

C.16 Using volencap for a Complex Configuration

This example shows how you can use the `volencap` utility to set up encapsulation for a complex configuration containing a mix of disk, domain, and partition types. The example uses the LSM setup shown in the following listing:

```
# volprint -ht

DG NAME         GROUP-ID
DM NAME         DEVICE       TYPE     PRIVLEN  PUBLEN   PUBPATH
V  NAME         USETYPE      KSTATE   STATE    LENGTH   READPOL  PREFPLEX
PL NAME         VOLUME       KSTATE   STATE    LENGTH   LAYOUT   ST-WIDTH
MODE
SD NAME         PLEX         PLOFFS   DISKOFFS LENGTH   DISK-NAME    DEVICE

dg rootdg       821915478.1025.lsmtest

dm rz3d         rz3d         simple   1024     0        /dev/rrz3d
```

In this example, `/usr` is an AdvFS file system on `rz3g`.

```
# ls /etc/fdmns/usr_domain
rz3g@
```

The following LSM command will attempt to set up encapsulation for numerous disks and partitions as well as an AdvFS domain.

```
# volencap rz3d usr_domain rz3h rz3g rz9a rz9b rz10 rz9 rz3

Setting up encapsulation for rz3d.
  Cannot encapsulate rz3d since the following disks/partitions are already
  in use by LSM: rz3d

Setting up encapsulation for usr_domain.
- Creating nopriv disk for rz3g.

Setting up encapsulation for rz3h.
- Creating nopriv disk for rz3h.

Setting up encapsulation for rz3g.
  Cannot encapsulate rz3g since the following disks/partitions are already
  setup for encapsulation: rz3g

Setting up encapsulation for rz9a.
- Creating nopriv disk for rz9a.

Setting up encapsulation for rz9b.
- Creating nopriv disk for rz9b.

Setting up encapsulation for rz10.
- Creating simple disk for rz10c.

Setting up encapsulation for rz9.
  Cannot encapsulate rz9c since the following disks/partitions are already
  setup for encapsulation: rz9a rz9b

Setting up encapsulation for rz3.
```

```
  Cannot encapsulate rz3 since the following disks/partitions are already
  in use by LSM: rz3d

The following disks are queued up for encapsulation or use by LSM.
You must reboot the system to perform the actual encapsulations.
    rz3g rz3h rz9a rz9b rz10c
```

When the system reboots, the `init` process executes `/sbin/vol-reconfig`, which performs the encapsulations. All the necessary system files are updated (for example, `/etc/fstab` and `/etc/fdmns`).

Note the following results of the encapsulation process:

- Partitions are encapsulated as nopriv disks (for example, partitions `rz3g`, `rz3h`, `rz9a`, and `rz9b`).
- Disks are encapsulated as simple disks if there is appropriate space to store the private region (for example, disk `rz10`).
- The individual partitions within an AdvFS domain are encapsulated as if the partitions were specified directly. For example, specifying `volencap usr_domain` resulted in encapsulation of the partition, `rz3g`.

C.17 Mirroring the Root Disk

This section shows how to mirror the root disk using the `volmirror` utility. The examples assume that the root disk has already been encapsulated as described in Section C.15.

The disk used as the mirror must not have any partitions in use. The `volrootmir` utility will only mirror the root disk to an unused disk. For example:

```
# volrootmir rz10

  Some partitions on rz10 seem to be in use.  Reinitialize
  the disklabel before using rz10 for mirroring the root disk.
```

The following example shows how to initialize the disk label to use the disk as a root mirror.

```
# disklabel -z rz10
# disklabel -wr rz10 rz26
# volrootmir rz10

  Mirroring rootvol to rz10a.
  Mirroring swapvol to rz10b.
```

In the following example, the `-a` option is used with `volrootmir` to mirror the entire root disk to the target disk. This includes copying the disk partition map and mirroring all volumes on the disk.

```
# volrootmir -a rz10

  Mirroring system disk rz3 to disk rz10.

  This operation will destroy all contents on disk rz10.
  The disk label from rz3 will be copied to rz10 and
  all volumes associated with rz3 will be mirrored.

  Do you want to continue with this operation? (y or n) y

  Initializing rz10.

  Mirroring rootvol to rz10a.
  Mirroring swapvol to rz10b.
  Mirroring vol-rz3g to rz10g.
```

When mirroring the entire root disk, the target disk must be of the same type as the root disk. If the disks differ, `volrootmir` prints an error and exits. For example:

```
# volrootmir -a rz13

  ERROR:   disk rz3 is an RZ26 type device while
           disk rz13 is an RZ73 type device.
           Both disks must be of the same type.
```

If you want to mirror only the root and swap partitions, the target disk can be a different type. For example:

```
# volrootmir rz13

  Mirroring rootvol to rz13a.
  Mirroring swapvol to rz13b.
```

C.18 Deleting Queued Encapsulation Requests

This example shows how to delete queued encapsulation requests.

Follow these steps:

1. List the partitions and disks that are queued for encapsulation.

   ```
   # volencap -s

   The following disks are queued up for encapsulation or use by LSM.
   You must reboot the system to perform the actual encapsulations.
       rz3g rz3h rz9a rz9b
   ```

2. Specify the partition or disk names of the requests you want to remove. If you specify a disk name, the requests for all the partitions on that disk will be removed.

   To remove the encapsulation requests for specific partitions, use the following commands:

```
# volencap -k rz3g rz9a
# volencap -s

The following disks are queued up for encapsulation or use by LSM.
You must reboot the system to perform the actual encapsulations.
    rz3h rz9b
```

To remove the encapsulation requests for an entire disk, use the following commands:

```
# volencap -k rz3
# volencap -s

The following disks are queued up for encapsulation or use by LSM.
You must reboot the system to perform the actual encapsulations.
    rz9a rz9b
```

C.19 Unencapsulating the System Boot Disk

This example shows how to unencapsulate the system boot disk using the `volunroot` utility. The following listing shows the configuration used in the example.

```
# volprint -ht
DG NAME      GROUP-ID
DM NAME      DEVICE       TYPE     PRIVLEN  PUBLEN   PUBPATH
V  NAME      USETYPE      KSTATE   STATE    LENGTH   READPOL   PREFPLEX
PL NAME      VOLUME       KSTATE   STATE    LENGTH   LAYOUT    ST-WIDTH
MODE
SD NAME      PLEX         PLOFFS   DISKOFFS LENGTH   DISK-NAME  DEVICE

dg rootdg    821915478.1025.lsmtest

dm rz10a     rz10a        nopriv   0        131072   /dev/rrz10a
dm rz10b     rz10b        nopriv   0        261120   /dev/rrz10b
dm rz10d     rz10d        simple   1024     0        /dev/rrz10d
dm rz10g     rz10g        nopriv   0        819200   /dev/rrz10g
dm rz3a      rz3a         nopriv   0        131072   /dev/rrz3a
dm rz3b      rz3b         nopriv   0        261120   /dev/rrz3b
dm rz3d      rz3d         simple   1024     0        /dev/rrz3d
dm rz3g      rz3g         nopriv   0        819200   /dev/rrz3g

v  rootvol    root         ENABLED  ACTIVE   131072   ROUND   -
pl rootvol-01 rootvol      ENABLED  ACTIVE   131072   CONCAT  -        RW
sd rz3a-01p   rootvol-01   0        0        16       rz3a        rz3a
sd rz3a-01    rootvol-01   16       16       131056   rz3a        rz3a
pl rootvol-02 rootvol      ENABLED  ACTIVE   131072   CONCAT  -        RW
sd rz10a-01p  rootvol-02   0        0        16       rz10a       rz10a
sd rz10a-01   rootvol-02   16       16       131056   rz10a       rz10a

v  swapvol    swap         ENABLED  ACTIVE   261120   ROUND   -
pl swapvol-01 swapvol      ENABLED  ACTIVE   261120   CONCAT  -        RW
sd rz3b-01    swapvol-01   0        0        261120   rz3b        rz3b
pl swapvol-02 swapvol      ENABLED  ACTIVE   261120   CONCAT  -        RW
sd rz10b-01   swapvol-02   0        0        261120   rz10b       rz10b
```

```
v  vol-rz3g    fsgen        ENABLED  ACTIVE   819200   SELECT  -
pl vol-rz3g-01 vol-rz3g     ENABLED  ACTIVE   819200   CONCAT  -        RW
sd rz3g-01     vol-rz3g-01  0        0        819200   rz3g        rz3g
pl vol-rz3g-02 vol-rz3g     ENABLED  ACTIVE   819200   CONCAT  -        RW
sd rz10g-01    vol-rz3g-02  0        0        819200   rz10g       rz10g
```

The `volunroot` utility will not unencapsulate volumes that do not map directly to a partition or that are mirrored. For example:

```
# volunroot

  There are 2 plexes associated with volume rootvol.
  rootvol should have only 1 plex to use volunroot.
  The volunroot operation cannot proceed. Please refer to volunroot(8).
```

To unencapsulate the boot disk, the mirrors must first be removed from the volumes. For example:

```
# volplex dis rootvol-02 swapvol-02 vol-rz3g-02
# voledit -rf rm rootvol-02 swapvol-02 vol-rz3g-02
# voldg rmdisk rz10a rz10b rz10g rz10d
# voldisk rm rz10a rz10b rz10g rz10d
# volunroot

  This operation will convert the following file systems on the
  system disk rz3 from LSM volumes to regular disk partitions:
        Replace volume rootvol with rz3a.
        Replace volume swapvol with rz3b.

  This operation will require a system reboot.  If you choose to
  continue with this operation, your system files will be updated
  to discontinue the use of the above listed LSM volumes.  You must
  then reboot the system.  /sbin/vol-reconfig should be present in
  /etc/inittab to remove the named volumes during system reboot.

  Do you wish to do this now ? (y or n) n
```

If the `volunroot` command is used with no arguments, `volunroot` unencapsulates only the `rootvol` and `swapvol` volumes.

If the `-a` argument is supplied, `volunroot` attempts to unencapsulate all previously-encapsulated volumes on the boot disk.

The `volunroot` utility modifies the necessary system files to remove the selected volumes. Then `volunroot` creates a script which is run by `/sbin/vol-reconfig` during `init` processing. The script then deletes the selected volumes and associated disks from LSM.

Note that `volunroot` shuts down to the boot prompt and does not automatically reboot the system. This allows you to update the default boot device if necessary.

```
# volunroot -a

This operation will convert the following file systems on the
system disk rz3 from LSM volumes to regular disk partitions:
      Replace volume rootvol with rz3a.
      Replace volume swapvol with rz3b.
      Replace volume vol-rz3g with rz3g.

This operation will require a system reboot.  If you choose to
continue with this operation, your system files will be updated
to discontinue the use of the above listed LSM volumes.  You must
then reboot the system.  /sbin/vol-reconfig should be present in
/etc/inittab to remove the named volumes during system reboot.

 Do you wish to do this now ? (y or n) y

Changing rootvol in /etc/fdmns/root_domain to /dev/rz3a.
Removing 'lsm_rootdev_is_volume=' entry in /etc/sysconfigtab.
Changing /dev/vol/rootdg/swapvol in /etc/fstab to /dev/rz3b.
Removing 'lsm_swapdev_is_volume=' entry in /etc/sysconfigtab.
Changing vol-rz3g in /etc/fdmns/usr_domain to /dev/rz3g.

A shutdown is now required to complete the unencapsulation process.
Please shutdown before performing any additional LSM or disk reconfiguration.

When would you like to shutdown (in minutes)
e.g. now, q, 1,2,3,4 (default: 2) ? now
Shutdown at 19:34 (in 0 minutes) [pid 874]

        *** FINAL System shutdown message from root@lsmtest ***

System going down IMMEDIATELY
        ...

System shutdown time has arrived
        ...

System shutdown time has arrived
# syncing disks... 2 done
CPU 0: Halting... (transferring to monitor)

?05 HLT INSTR
  PC= FFFFFC00.004401C0 PSL= 00000000.00000005

>>>b

ADVFS: using 566 buffers containing 4.42 megabytes of memory
starting LSM
LSM: Can't open device rz3a, device busy or inaccessible.
Checking local filesystems
/sbin/ufs_fsck -p
```

Notice the error message LSM prints when attempting to access device `rz3a`. This is expected behavior. The system has mounted `rz3a` for root and LSM still recognizes `rz3a` as an LSM disk. When LSM starts up, it will attempt to open `rz3a`, resulting in the error message. Once LSM has started, it will remove `rz3a` from its configuration to complete the root disk unencapsulation process. After this, the error message will not be seen again.

```
# volprint -ht

DG NAME         GROUP-ID
DM NAME         DEVICE       TYPE     PRIVLEN  PUBLEN   PUBPATH
V  NAME         USETYPE      KSTATE   STATE    LENGTH   READPOL  PREFPLEX
PL NAME         VOLUME       KSTATE   STATE    LENGTH   LAYOUT   ST-WIDTH
MODE
SD NAME         PLEX         PLOFFS   DISKOFFS LENGTH   DISK-NAME    DEVICE

dg rootdg       821915478.1025.lsmtest

dm rz3d         rz3d         simple   1024     0        /dev/rrz3d
```

C.20 Unencapsulating the /usr and /var File Systems

To use disk partitions instead of LSM volumes for the `/usr` or `/var` file systems, follow these steps:

1. Boot the system to single-user mode.
2. If the file system is AdvFS, change the link in the `/etc/fdmns/`*`domain`* directory to use a disk partition instead of an LSM volume. For example:

   ```
   # cd /etc/fdmns/usr_domain
   # ls -l
   total 0
   lrwxrwxrwx   1 root     system        24 Dec 12 14:37 vol-
   rz3h-> /dev/vol/ rootdg/vol-rz3h
   # rm -f vol-rz3h
   # ln -s /dev/rz10g rz10g
   ```

   If the file system is UFS, change the entry for the `/usr` file system in `/etc/fstab` from an LSM volume to a disk partition.
3. Restart LSM (since the system is in single-user mode at this point). Then remove the LSM volume and the associated disks. For example:

   ```
   # /sbin/lsmbstartup
   # volume stop vol-rz3h
   # voledit -r vol-rz3h
   # voldg rmdisk rz10g rz11g
   # voldisk rm rz10g rz11g
   ```

C.21 Creating a Back Up Copy of an LSM Configuration

Digital recommends that you back up the current LSM configuration on a regular basis, using the `volsave` utility. You can use the default backup directory (`/usr/var/lsm/db`), or specify a location of your choice.

To create a backup copy of the current LSM configuration using the default backup directory, enter the `volsave` command with no options. Note that the backslash in the following example is for line continuation and is not in the actual command.

```
# volsave
LSM configuration being saved to  \
  /usr/var/lsm/db/LSM.19951226203620.skylark

volsave does not save configuration for volumes used for
root, swap, /usr or /var.
LSM configuration for following system disks not saved:
rz8a rz8b

LSM Configuration saved successfully.

# cd /usr/var/lsm/db/LSM.19951226203620.skylark
# ls
dg1.d         header        volboot
dg2.d         rootdg.d      voldisk.list
```

In this example, the `volsave` utility created the following files and directories:

- A timestamped subdirectory, `LSM.19951226203620.skylark`, containing the `header`, `volboot`, and `voldisk.list` description files
- A *`diskgroup`*`.d` subdirectory for each of the system's three disk groups, `dg1`, `dg2`, and `rootdg`.
- An `allvol.DF` file in each of the *`diskgroup`*`.d` subdirectories. This file is a `volmake` description file for all volumes, plexes, and subdisks in that disk group.

Note that `volsave` does not save volumes associated with the root, swap, `/usr` and `/var` file systems. After the `rootdg` disk group is restored, the partitions that are in use on the system disk will have to be reencapsulated using the procedure described in Chapter 4.

To save the LSM configuration in a timestamped subdirectory in a directory other than `/usr/var/lsm/db`, use the following command syntax:

```
# volsave -d  /usr/var/dirname/LSM.%date
```

For example, the following command saves the LSM configuration in the `/usr/var/config` subdirectory. Note that the backslash in the following example is for line continuation and is not in the actual display.

```
# volsave -d /usr/var/config/LSM.%date
LSM configuration being saved to \
  /usr/var/config/LSM.19951226203658
   .
   .
   .
LSM Configuration saved successfully.
```

To save an LSM configuration to a specific directory, use the following command syntax:

```
# volsave -d dirname
```

For example, the following command saves the LSM configuration in the `/usr/var/LSM.config1` subdirectory:

```
# volsave -d /usr/var/LSM.config1
   LSM configuration being saved to /usr/var/LSM.config1
   .
   .
   .
  LSM Configuration saved successfully.
```

C.22 Listing a Saved LSM Configuration

To list the LSM configuration from the last timestamped subdirectory in `/usr/var/lsm/db`, use the following command:

```
# volrestore -l
```

To list the LSM configuration in any other directory use the `-l` and `-d` options with `volrestore` as shown below:

```
# volrestore -l -d /usr/var/config/LSM.19951226203658.skylark
```

or

```
# volrestore -l -d /usr/LSM.config1
```

The `volrestore` utility lists the LSM configuration in a format similar to the output of `volprint -htA`.

For example:

```
# volrestore -l

  LSM Configuration Save Utility Version 1.
  Configuration Information stored in directory
  /usr/var/lsm/db/LSM.19951226203620.skylark
  Created at Tue Dec 26 20:36:30 EST 1995 on HOST skylark

  LSM Configuration for Diskgroup dg1.

Working .

dm      rz10    rz10    sliced  512     2050332 /dev/rz10g
dm      rz11g   rz11g   simple  512     818688  /dev/rz11g
dm      rz11h   rz11h   nopriv  0       838444  /dev/rz11h

v  vol1         fsgen        ENABLED  ACTIVE   20480    SELECT ...
pl vol1-01      vol1         ENABLED  ACTIVE   20480    CONCAT ...
sd rz11g-02     vol1-01      0        20480    20480    rz11g  ...
pl vol1-02      vol1         ENABLED  ACTIVE   20480    CONCAT ...
```

```
sd rz10-02      vol1-02      0        20480    20480    rz10   ...

v  vol2         fsgen        ENABLED  ACTIVE   20480    SELECT ...
pl vol2-01      vol2         ENABLED  ACTIVE   20480    CONCAT ...
sd rz11g-01     vol2-01      0        0        20480    rz11g  ...
pl vol2-02      vol2         ENABLED  ACTIVE   20480    CONCAT ...
sd rz10-01      vol2-02      0        0        20480    rz10   ...

v  vol3         fsgen        ENABLED  ACTIVE   20480    SELECT ...
pl vol3-01      vol3         ENABLED  ACTIVE   20480    CONCAT ...
sd rz11g-03     vol3-01      0        40960    20480    rz11g  ...

  LSM Configuration for Diskgroup dg2.

Working .

dm      rz11b   rz11b   simple  128     262016  /dev/rz11b
dm      rz9     rz9     sliced  512     2050332 /dev/rz9g

v  vol1         fsgen        ENABLED  ACTIVE   100      SELECT ...
pl vol1-01      vol1         ENABLED  ACTIVE   100      CONCAT ...
sd rz11b-01     vol1-01      0        0        100      rz11b  ...

v  vol6         fsgen        ENABLED  ACTIVE   100      SELECT ...
pl vol6-01      vol6         ENABLED  ACTIVE   100      CONCAT ...
sd rz11b-06     vol6-01      0        500      100      rz11b  ...
pl vol6-02      vol6         ENABLED  ACTIVE   100      CONCAT ...
sd rz9-02       vol6-02      0        100      100      rz9    ...

v  vol9         fsgen        ENABLED  ACTIVE   100      SELECT ...
pl vol9-01      vol9         ENABLED  ACTIVE   100      CONCAT ...
sd rz11b-09     vol9-01      0        800      100      rz11b  ...
pl vol9-02      vol9         ENABLED  ACTIVE   100      CONCAT ...
sd rz9-01       vol9-02      0        0        100      rz9    ...

  LSM Configuration for Diskgroup rootdg.

Working .

dm      rz12    rz12    sliced  512     2050332 /dev/rz12g
dm      rz8a    rz8a    nopriv  0       131072  /dev/rz8a
dm      rz8b    rz8b    nopriv  0       262144  /dev/rz8b
dm      disk01  rz8h    simple  512     837932  /dev/rz8h

v  rootvol      root         ENABLED  ACTIVE   131072   ROUND  ...
pl rootvol-01   rootvol      ENABLED  ACTIVE   131072   CONCAT ...
sd rz8a-01      rootvol-01   0        0        131072   rz8a   ...

v  swapvol      swap         ENABLED  ACTIVE   262144   ROUND  ...
pl swapvol-01   swapvol      ENABLED  ACTIVE   262144   CONCAT ...
sd rz8b-01      swapvol-01   0        0        262144   rz8b   ...

v  vol1         fsgen        ENABLED  ACTIVE   20480    SELECT ...
pl vol1-01      vol1         ENABLED  ACTIVE   20480    CONCAT ...
sd disk01-01    vol1-01      0        0        20480    disk01 ...
pl vol1-02      vol1         ENABLED  ACTIVE   20480    CONCAT ...
sd rz12-01      vol1-02      0        0        20480    rz12   ...

v  vol2         fsgen        ENABLED  ACTIVE   20480    SELECT ...
pl vol2-01      vol2         ENABLED  ACTIVE   20480    CONCAT ...
```

```
sd disk01-02    vol2-01      0        20480    20480    disk01 ...
pl vol2-02      vol2         ENABLED  ACTIVE   20480    CONCAT ...
sd rz12-02      vol2-02      0        20480    20480    rz12   ...
```

C.23 Restoring an LSM Configuration

To restore an LSM configuration, you first need to determine the location of the most recent copy of the LSM configuration that was saved using `volsave`.

- If you used the default directory (`/usr/var/lsm/db`), `volrestore` automatically retrieves the last timestamped subdirectory in that directory.
- If you used the `-d` option to specify another directory during the `volsave` operation, you must use the `-d` option to specify that same directory to `volrestore`.

To restore a specific disk group, enter the `volrestore` command with the `-g` option. The `volrestore` utility will attempt to reimport the disk group. If the import operation succeeds, any volumes that do not exist will be re-created. If the import operation fails, `volrestore` re-creates the disk group.

The following example shows a successful import operation of disk group `dg1`.

```
# volrestore -g dg1
Using LSM configuration from /usr/var/lsm/db/LSM.19951226203620.skylark
Created at Tue Dec 26 20:36:30 EST 1995 on HOST skylark

Would you like to continue ? [y,n,q,?] (default: n) y

Working .

  Restoring dg1

  vol1 in diskgroup dg1 already exists. (Skipping ..)

  vol2 in diskgroup dg1 already exists. (Skipping ..)

  vol3 in diskgroup dg1 already exists. (Skipping ..)
```

The following example shows what happens when the disk group import operation fails:

```
# volrestore -g dg1

Using LSM configuration from /usr/var/lsm/db/LSM.19951226203620.skylark
Created at Tue Dec 26 20:36:30 EST 1995 on HOST skylark

Would you like to continue ? [y,n,q,?] (default: n) y

+ voldg init dg1 rz10=rz10
Working .
```

```
Restoring dg1

Checking vol1

Checking vol2

Checking vol3

The following volumes in diskgroup dg1 cannot be
started. Refer to LSM documentation on how to set
the plex states before starting the volumes.

vol1
vol2
vol3
```

The `volrestore` utility creates all volumes and plexes in the disk group in the DISABLED and EMPTY state. If a volume is mirrored, before starting the volume make sure to set the state to STALE for plexes with outdated data.

Take care to set the plex state appropriately, since the plex state can change between the time the LSM configuration was saved using `volsave` and the time when the configuration is restored.

For example, in the configuration listed below, if disk `rz10` had a failure just prior to the restoration of the LSM configuration using `volrestore`, you would set the plexes `vol1-02` and `vol2-02` to the STALE state before starting the volumes `vol1` and `vol2`.

```
# volprint -ht -g dg1
DG NAME         GROUP-ID
DM NAME         DEVICE       TYPE     PRIVLEN  PUBLEN   PUBPATH
V  NAME         USETYPE      KSTATE   STATE    LENGTH   READPOL  ...
PL NAME         VOLUME       KSTATE   STATE    LENGTH   LAYOUT   ...
SD NAME         PLEX         PLOFFS   DISKOFFS LENGTH   DISK-NAME...

dg dg1          820028958.80337.lsm

dm rz10         rz10         sliced   512      2050332  /dev/rrz10g
dm rz11g        rz11g        simple   512      818688   /dev/rrz11g
dm rz11h        rz11h        nopriv   0        838444   /dev/rrz11h

v  vol1         fsgen        DISABLED EMPTY    20480    SELECT   ...
pl vol1-01      vol1         DISABLED EMPTY    20480    CONCAT   ...
sd rz11g-02     vol1-01      0        20480    20480    rz11g    ...
pl vol1-02      vol1         DISABLED EMPTY    20480    CONCAT   ...
sd rz10-02      vol1-02      0        20480    20480    rz10     ...

v  vol2         fsgen        DISABLED EMPTY    20480    SELECT   ...
pl vol2-01      vol2         DISABLED EMPTY    20480    CONCAT   ...
sd rz11g-01     vol2-01      0        0        20480    rz11g    ...
pl vol2-02      vol2         DISABLED EMPTY    20480    CONCAT   ...
sd rz10-01      vol2-02      0        0        20480    rz10     ...

v  vol3         fsgen        DISABLED EMPTY    20480    SELECT   ...
pl vol3-01      vol3         DISABLED EMPTY    20480    CONCAT   ...
sd rz11g-03     vol3-01      0        40960    20480    rz11g    ...
```

```
# volume -g dg1 init clean vol1 vol1-01
# volume -g dg1 init clean vol2 vol2-01
# volprint -ht -g dg1
DG NAME         GROUP-ID
DM NAME         DEVICE       TYPE     PRIVLEN  PUBLEN   PUBPATH
V  NAME         USETYPE      KSTATE   STATE    LENGTH   READPOL  ...
PL NAME         VOLUME       KSTATE   STATE    LENGTH   LAYOUT   ...
SD NAME         PLEX         PLOFFS   DISKOFFS LENGTH   DISK-NAME...

dg dg1          820028958.80337.lsm

dm rz10         rz10         sliced   512      2050332  /dev/rrz10g
dm rz11g        rz11g        simple   512      818688   /dev/rrz11g
dm rz11h        rz11h        nopriv   0        838444   /dev/rrz11h

v  vol1         fsgen        DISABLED CLEAN   20480    SELECT    ...
pl vol1-01      vol1         DISABLED CLEAN   20480    CONCAT    ...
sd rz11g-02     vol1-01      0        20480    20480    rz11g    ...
pl vol1-02      vol1         DISABLED STALE   20480    CONCAT    ...
sd rz10-02      vol1-02      0        20480    20480    rz10     ...

v  vol2         fsgen        DISABLED CLEAN    20480    SELECT   ...
pl vol2-01      vol2         DISABLED CLEAN    20480    CONCAT   ...
sd rz11g-01     vol2-01      0        0        20480    rz11g    ...
pl vol2-02      vol2         DISABLED STALE    20480    CONCAT   ...
sd rz10-01      vol2-02      0        0        20480    rz10     ...

v  vol3         fsgen        DISABLED EMPTY    20480    SELECT   ...
pl vol3-01      vol3         DISABLED EMPTY    20480    CONCAT   ...
sd rz11g-03     vol3-01      0        40960    20480    rz11g    ...
```

Once the plex states for mirrored volumes have been set appropriately, the volumes can be started. Starting the volumes will resynchronize the stale plexes in the volume. For example:

```
# volume -g dg1 start vol1
# volume -g dg1 start vol2
# volume -g dg1 start vol3
# volprint -ht
DG NAME         GROUP-ID
DM NAME         DEVICE       TYPE     PRIVLEN  PUBLEN   PUBPATH
V  NAME         USETYPE      KSTATE   STATE    LENGTH   READPOL  ...
PL NAME         VOLUME       KSTATE   STATE    LENGTH   LAYOUT   ...
SD NAME         PLEX         PLOFFS   DISKOFFS LENGTH   DISK-NAME...

dg rootdg       818201657.1025.lsm

dm disk01       rz8h         simple   512      837932   /dev/rrz8h
dm rz12         rz12         sliced   512      2050332  /dev/rrz12g
dm rz8a         rz8a         nopriv   0        131072   /dev/rrz8a
dm rz8b         rz8b         nopriv   0        262144   /dev/rrz8b

v  rootvol      root         ENABLED  ACTIVE   131072   ROUND    ...
pl rootvol-01   rootvol      ENABLED  ACTIVE   131072   CONCAT   ...
sd rz8a-01      rootvol-01   0        0        131072   rz8a     ...

v  swapvol      swap         ENABLED  ACTIVE   262144   ROUND    ...
pl swapvol-01   swapvol      ENABLED  ACTIVE   262144   CONCAT   ...
sd rz8b-01      swapvol-01   0        0        262144   rz8b     ...
```

```
v  vol1         fsgen        ENABLED  ACTIVE   20480    SELECT   ...
pl vol1-01      vol1         ENABLED  ACTIVE   20480    CONCAT   ...
sd disk01-01    vol1-01      0        0        20480    disk01   ...
pl vol1-02      vol1         ENABLED  ACTIVE   20480    CONCAT   ...
sd rz12-01      vol1-02      0        0        20480    rz12     ...

v  vol2         fsgen        ENABLED  ACTIVE   20480    SELECT   ...
pl vol2-01      vol2         ENABLED  ACTIVE   20480    CONCAT   ...
sd disk01-02    vol2-01      0        20480    20480    disk01   ...
pl vol2-02      vol2         ENABLED  ACTIVE   20480    CONCAT   ...
sd rz12-02      vol2-02      0        20480    20480    rz12     ...
```

C.24 Re-creating a Volume in a Disk Group

If a volume is deleted by mistake and needs to be re-created, you can use the `volrestore` utility to re-create the volume configuration.

To re-create a specific volume, use the `-g` and `-v` options with `volrestore`.

For example, to re-create volume `vol2` you would use the following command:

```
# volrestore -g dg1 -v vol2

Using LSM configuration from /usr/var/lsm/db/LSM.19951226203620.skylark
Created at Tue Dec 26 20:36:30 EST 1995 on HOST skylark

Would you like to continue ? [y,n,q,?] (default: n) y

Working .

  Restoring dg1

  Checking vol2

  The following volumes in diskgroup dg1 cannot be
  started. Refer to LSM documentation on how to set
  the plex states before starting the volumes.

  vol2
```

If both plexes of the volume contain valid data, they can be set to the ACTIVE state and no plex recovery will be carried out.

For example:

```
# volume -g dg1 init active vol2
# volprint -ht -g dg1
DG NAME         GROUP-ID
DM NAME         DEVICE       TYPE     PRIVLEN  PUBLEN   PUBPATH
V  NAME         USETYPE      KSTATE   STATE    LENGTH   READPOL  ...
PL NAME         VOLUME       KSTATE   STATE    LENGTH   LAYOUT   ...
SD NAME         PLEX         PLOFFS   DISKOFFS LENGTH   DISK-NAME...

dg dg1          820028958.80337.lsm

dm rz10         rz10         sliced   512      2050332  /dev/rrz10g
```

```
dm rz11g        rz11g       simple   512      818688   /dev/rrz11g
dm rz11h        rz11h       nopriv   0        838444   /dev/rrz11h

v  vol1         fsgen       ENABLED  ACTIVE   20480    SELECT   ...
pl vol1-01      vol1        ENABLED  ACTIVE   20480    CONCAT   ...
sd rz11g-02     vol1-01     0        20480    20480    rz11g    ...
pl vol1-02      vol1        ENABLED  ACTIVE   20480    CONCAT   ...
sd rz10-02      vol1-02     0        20480    20480    rz10     ...

v  vol2         fsgen       ENABLED  ACTIVE   20480    SELECT   ...
pl vol2-01      vol2        ENABLED  ACTIVE   20480    CONCAT   ...
sd rz11g-01     vol2-01     0        0        20480    rz11g    ...
pl vol2-02      vol2        ENABLED  ACTIVE   20480    CONCAT   ...
sd rz10-01      vol2-02     0        0        20480    rz10     ...

v  vol3         fsgen       ENABLED  ACTIVE   20480    SELECT   ...
pl vol3-01      vol3        ENABLED  ACTIVE   20480    CONCAT   ...
sd rz11g-03     vol3-01     0        40960    20480    rz11g    ...

# volinfo -g dg1
vol1           fsgen    Started
vol3           fsgen    Started
vol2           fsgen    Started
```

Volumes can also be re-created by specifying just the `-g` option with `volrestore`.

For example:

```
# volrestore -g dg1

Using LSM configuration from /usr/var/lsm/db/LSM.19951226203620.skylark
Created at Tue Dec 26 20:36:30 EST 1995 on HOST skylark

Would you like to continue ? [y,n,q,?] (default: n) y

Working .

  Restoring dg1

  vol1 in diskgroup dg1 already exists. (Skipping ..)

  Checking vol2

  vol3 in diskgroup dg1 already exists. (Skipping ..)

  The following volumes in diskgroup dg1 cannot be
  started. Refer to LSM documentation on how to set
  the plex states before starting the volumes.

  vol2
```

After setting the plex states appropriately, restart the re-created volume `vol2`.

C.25 Restoring the rootdg Disk Group Configuration

To restore the `rootdg` disk group configuration, use the following command:

```
# volrestore -g rootdg

Using LSM configuration from /usr/var/lsm/db/LSM.19951226203620.skylark
Created at Tue Dec 26 20:36:30 EST 1995 on HOST skylark

Would you like to continue ? [y,n,q,?] (default: n) y

  Disk rz8a is in use for a system (root, /usr, /var)
  volume. volrestore will not restore disks and volumes
  used for root, swap, /usr, or /var. Refer to volrestore(8).

  Disk rz8b is in use for a system (root, /usr, /var)
  volume. volrestore will not restore disks and volumes
  used for root, swap, /usr, or /var. Refer to volrestore(8).

Working .

  Restoring rootdg

  Checking vol1

  Checking vol2

  The following volumes in diskgroup rootdg cannot be
  started. Refer to LSM documentation on how to set
  the plex states before starting the volumes.

  vol1
  vol2
```

In this example, you would need to reencapsulate the system disk `rz8` to use LSM volumes. Also, you would need to set the plex states appropriately for volumes `vol1` and `vol2` in the `rootdg` disk group and then restart the volumes.

C.26 Handling volrestore Failures

The `volrestore` utility fails if a disk is unavailable or if an LSM object name is already in use, causing a conflict between the current configuration and the saved configuration. When `volrestore` encounters a failure in restoring a disk group, it backs out the changes made for that disk group and proceeds with the next disk group that needs to be restored.

C.26.1 Overriding a volrestore Failure

In the following example, `volrestore` fails because disk `rz10` in disk group dg1 cannot be initialized.

```
# volrestore -g dg1

Using LSM configuration from /usr/var/lsm/db/LSM.19951226203620.skylark
Created at Tue Dec 26 20:36:30 EST 1995 on HOST skylark

Would you like to continue ? [y,n,q,?] (default: n) y

Initializing disk rz10 failed.
voldisk: Device rz10: define failed: Device path not valid

  Quitting ....
```

To override the failure, use the -b option with volrestore. This option specifies the ‘‘best’’ possible configuration despite the failures. All failures are reported, as shown below:

```
# volrestore -b -g dg1

Using LSM configuration from /usr/var/lsm/db/LSM.19951226203620.skylark
Created at Tue Dec 26 20:36:30 EST 1995 on HOST skylark

Would you like to continue ? [y,n,q,?] (default: n) y

Initializing disk rz10 failed.
voldisk: Device rz10: define failed: Device path not valid
Working .

  Restoring dg1

  vol1 in diskgroup dg1 could not be
  restored because of the following errors:
volmake: Failed to obtain locks:
        rz10: no such object in the configuration

  The configuration of vol1 in /usr/var/lsm/db/LSM.19951226203620 is

v  vol1         fsgen        ENABLED  ACTIVE   20480    SELECT   ...
pl vol1-01      vol1         ENABLED  ACTIVE   20480    CONCAT   ...
sd rz11g-02     vol1-01      0        20480    20480    rz11g    ...
pl vol1-02      vol1         ENABLED  ACTIVE   20480    CONCAT   ...
sd rz10-02      vol1-02      0        20480    20480    rz10     ...

  vol2 in diskgroup dg1 could not be
  restored because of the following errors:
volmake: Failed to obtain locks:
        rz10: no such object in the configuration

  The configuration of vol2 in /usr/var/lsm/db/LSM.19951226203620 is

v  vol2         fsgen        ENABLED  ACTIVE   20480    SELECT   ...
pl vol2-01      vol2         ENABLED  ACTIVE   20480    CONCAT   ...
sd rz11g-01     vol2-01      0        0        20480    rz11g    ...
pl vol2-02      vol2         ENABLED  ACTIVE   20480    CONCAT   ...
sd rz10-01      vol2-02      0        0        20480    rz10     ...

  vol3 in diskgroup dg1 already exists. (Skipping ..)

  The following volumes in diskgroup dg1 cannot be
```

```
  started. Refer to LSM documentation on how to set
  the plex states before starting the volumes.

  vol3
```

In this example, volumes `vol1` and `vol2` could not be restored because the disk that these volumes use (`rz10`) was unavailable. To restore these volumes, the LSM configuration that was saved using `volsave` needs to be edited. See Section C.26.3 for further information.

C.26.2 Resolving Conflicts that Cause volrestore Failure

In the following example, `volrestore` fails because plex `vol3-01` is associated both with volume `vol3` in the saved LSM configuration and with volume `vol2` in the current configuration.

```
# volrestore -g dg1 -v vol3

Using LSM configuration from /usr/var/lsm/db/LSM.19951226203620.skylark
Created at Tue Dec 26 20:36:30 EST 1995 on HOST skylark

Would you like to continue ? [y,n,q,?] (default: n) y

Working .

  Restoring dg1

  vol3 in diskgroup dg1 could not be
  restored because of the following errors:
volmake: Plex vol3-01 already exists
volmake: Error associating plex vol3-01 with vol3:
        Record is associated

The configuration of vol3 in /usr/var/lsm/db/LSM.19951226203620.skylark is

v  vol3         fsgen        ENABLED  ACTIVE   20480    SELECT   ...
pl vol3-01      vol3         ENABLED  ACTIVE   20480    CONCAT   ...
sd rz11g-03     vol3-01      0        40960    20480    rz11g    ...
```

Conflicts such as the one shown in this example occur when LSM configuration changes are made after saving the LSM configuration with `volsave`.

To resolve such conflicts, you can either change the existing LSM configuration or edit the saved LSM configuration.

C.26.3 Editing an LSM Configuration After volrestore Failure

In the example in Section C.26.1, volumes `vol1` and `vol2` were not restored even though disk `rz11g` was still available. To restore these volumes with the one plex on `rz11g`, the LSM configuration in `/usr/var/lsm/db/LSM.19951226203620.skylark` must be edited, as follows:

1. Using an editor such as `vi`, edit the `allvol.DF` file in `/usr/var/lsm/db/LSM.19951226203620.skylark/dg1.d` as follows:
   - Remove `vol1-02` from the description of volume `vol1`
   - Remove `vol2-02` from the description of volume `vol2`
   - Remove the description of plexes `vol1-02` and `vol2-02` and the subdisks associated with them
2. Enter the `volrestore` command with the `-f` option. For example:

   ```
   # volrestore -b -f -g dg1 -v vol1 vol2
   ```

   You must use the `-f` option with `volrestore` to override checksum validation because editing the `allvol.DF` file causes checksum validation to fail.

C.27 Replicating an LSM Configuration

An LSM configuration created on one system can be replicated on other systems. Any volume can be replicated except volumes used for the root, `/usr`, and `/var` file systems and for the primary swap volume. The disk partitions used for root, `/usr`, `/var`, and primary swap must be encapsulated to LSM volumes as described in Chapter 4.

Follow these steps to replicate an LSM configuration:

1. Create an LSM configuration on the first system and save the configuration using `volsave`. For example:

   ```
   # volsave -d /usr/var/LSMCONFIG1

       LSM configuration being saved to /usr/var/LSMCONFIG1

       volsave does not save configuration for volumes used for
       root, swap, /usr or /var.
       LSM configuration for following system disks not saved:
       rz8a rz8b

       LSM Configuration saved successfully.
   ```

2. Copy the directory `/usr/var/LSMCONFIG1` to the second system. The `voldisk.list` file in this directory has a description of all disks used by LSM. If the physical disk names are different, the `voldisk.list` file will be different.

The voldisk.list file looks as follows:

```
# cd /usr/var/LSMCONFIG1
# cat voldisk.list

DEVICE  DISK  TYPE  GROUP  PRIVLEN  NCONFIG  CONFIGLEN  NLOG  LOGLEN
PUBLEN PUBPATH SYSTEMDISK
rz11b  rz11b  simple  dg2  128  2  31  2  4  262016  /dev/rz11b  NO
rz12  rz12  sliced  rootdg  512  0  0  0  0  2050332  /dev/rz12g  NO
rz8a  rz8a  nopriv  rootdg  131072  /dev/rz8a  YES
rz8b  rz8b  nopriv  rootdg  262144  /dev/rz8b  YES
rz8h  disk01  simple  rootdg  512  2  173  2  26  837932  /dev/rz8h...
rz9  rz9  sliced  dg2  512  2  173  2  26  2050332  /dev/rz9g  NO
```

3. Assume that on the second system the disks to be used by LSM are rz11, rz12, rz24, and rz25.

   Edit the voldisk.list file on the second system to change the physical device name. Do not change the disk media name.

   After editing, the voldisk.list file looks like this:

```
# cat voldisk.list

DEVICE  DISK  TYPE  GROUP  PRIVLEN  NCONFIG  CONFIGLEN  NLOG  LOGLEN
PUBLEN PUBPATH SYSTEMDISK
rz11b  rz11b  simple  dg2  128  2  31  2  4  262016  /dev/rz11b  NO
rz12  rz12  sliced  rootdg  512  0  0  0  0  2050332  /dev/rz12g  NO
rz24a  rz8a  nopriv  rootdg  131072  /dev/rz8a  YES
rz24b  rz8b  nopriv  rootdg  262144  /dev/rz8b  YES
rz24h  disk01  simple  rootdg  512  2  173  2  26  837932  /dev/rz8h...
rz25  rz9  sliced  dg2  512  2  173  2  26  2050332  /dev/rz9g  NO
```

4. Enter the volrestore command using the -f option to override checksum validation, as follows:

```
#  volrestore -f -d /usr/var/LSMCONFIG1

Using LSM configuration from /usr/var/LSMCONFIG1
Created at Wed Dec 27 20:23:29 EST 1995 on HOST skylark

Would you like to continue ? [y,n,q,?] (default: n) y

  /etc/vol/volboot does not exist. To restart LSM this
  file is required.

Restore saved copy of /etc/vol/volboot? [y,n,q,?] (default: y) y

  System does not have a valid rootdg configuration.

  Would you like to re-create rootdg from
  LSM description set in /usr/var/LSMCONFIG1 ??

Would you like to continue ? [y,n,q,?] (default: n) y

  Disk rz8a is in use for a system (root, /usr, /var)
  volume. volrestore will not restore disks and volumes
  used for root, swap, /usr, or /var. Refer to volrestore(8).

  Disk rz8b is in use for a system (root, /usr, /var)
```

```
  volume. volrestore will not restore disks and volumes
  used for root, swap, /usr, or /var. Refer to volrestore(8).

+ voldg adddisk disk01=rz24h
+ voldctl enable
Working .

  Restoring dg2

  vol1 in diskgroup dg2 already exists. (Skipping ..)

  vol6 in diskgroup dg2 already exists. (Skipping ..)

  vol9 in diskgroup dg2 already exists. (Skipping ..)

  The following volumes in diskgroup dg2 cannot be
  started. Refer to LSM documentation on how to set
  the plex states before starting the volumes.

  vol1
  vol6
  vol9

Working .

  Restoring rootdg

  Checking vol1

  Checking vol2

  The following volumes in diskgroup rootdg cannot be
  started. Refer to LSM documentation on how to set
  the plex states before starting the volumes.
```

C.28 Deinstalling LSM

Follow these steps to deinstall LSM:

1. Make sure that data on LSM volumes has been backed up and that none of the volumes are in use.
2. Deport all disk groups except the `rootdg` disk group and remove the disks used in the disk groups from LSM. For example:

   ```
   # voldg deport dg1 dg2
   # voldisk rm rz16 rz17 rz18 rz19 rz20 rz21 rz22 rz23
   ```

3. Remove all volumes in the `rootdg` disk group and remove all but the

last disk from the `rootdg` disk group. For example:

```
# voledit -g rootdg -rf rm vol1 vol2 vol3
# voldg rmdisk rz1h rz2 rz3 rz4
# voldisk rm rz1h rz2 rz3 rz4
```

4. Remove the `/etc/vol/volboot` file:

   ```
   # rm /etc/vol/volboot
   ```

5. Edit the `/etc/inittab` file and remove LSM entries.
6. Reboot the system.

   ```
   # shutdown -r now
   ```

The system now reboots without starting LSM and all the disks that LSM previously used are no longer in use. If required, you can removed the LSM subsets at this point.

C.29 Using ioctl Requests to Determine Volume Size

LSM supports the `ioctl` requests DEVIOCGET and DEVGETGEOM for use by applications that need to determine the volume size.

The DEVIOCGET request can be used to determine if a special device file is associated with an LSM volume.

The DEVGETGEOM request can be used to determine the size of the LSM volume.

For example, the following code segment shows the use of DEVGETGEOM to determine the volume size.

```
#include <sys/stat.h>
#include <sys/ioctl.h>
#include <sys/types.h>
#include <sys/file.h>

int
getsize(char *name)
{
   int lsm_fd;
   DEVGEOMST volgeom;
   struct stat st;
   long volsize = 0;

   /* Check if the device exists */

   if (stat(name, &st) < 0) {
     perror("stat");
   }

   /* open the volume */
```

```
    if ((lsm_fd = open(name, O_RDONLY)) == -1) {
      perror("open");
    }

    if ( ioctl(lsm_fd, DEVGETGEOM , (char *)(&volgeom)) == 0) {
      volsize =  volgeom.geom_info.dev_size;
    } else {
      /*ioctl failed */
      close(lsm_fd);
      perror("ioctl");
      return -1;
    }

    close(lsm_fd);

    return volsize; /* volsize in sectors */
}
```

Glossary

Some of the terms and objects that are used with the Logical Storage Manager are defined here.

concatenated plex
: A plex whose subdisks are associated at specific offsets within the address range of the plex, and extend in the plex address range for the length of the subdisk. This layout allows regions of one or more disks to create a plex, rather then a single big region.

description set
: A set of files that are saved using the `volsave`(8) command and can be used to restore an LSM configuration. By default, an LSM description set is saved in a timestamped directory under the `/usr/var/lsm/db` directory.

disk
: Disks exist as two entities. One is the physical disk on which all data is ultimately stored and which exhibits all the behaviors of the underlying technology. The other is the Logical Storage Manager presentation of disks which, while mapping one-to-one with the physical disks, are just presentations of units from which allocations of storage are made. As an example, a physical disk presents the image of a device with a definable geometry with a definable number of cylinders, heads etc. whereas a Logical Storage Manager disk is simply a unit of allocation with a name and a size.

disk access record
: A configuration record that defines a pathway to a disk. Disk access records most often name a unit number. The list of all disk access records stored in a system is used to find all disks attached to the system. Disk access records do not identify particular physical disks.

Through the use of disk IDs, the Logical Storage Manager allows disks to be moved between controllers, or to different locations on a controller. When a disk is moved, a different disk access record will be used when accessing the disk, although the disk media record will continue to track the actual physical disk.

On some systems, the Logical Storage Manager will build a list of disk access records automatically, based on the list of all devices attached to the system. On these systems, it is not necessary to define disk access records explicitly. On other systems, disk access records must be defined explicitly with the `/sbin/voldisk define` operation. Specialty disks (such as RAM disks or floppy disks) are likely to require explicit `/sbin/voldisk define` operations on all systems.

Disk access records are identified by their disk access names (also known as DA names).

disk group

A group of disks that share a common configuration. A configuration consists of a set of records describing objects including disks, volumes, plexes, and subdisks that are associated with one particular disk group. Each disk group has an administrator-assigned name that can be used by the administrator to reference that disk group. Each disk group has an internally defined unique disk group ID, which is used to differentiate two disk groups with the same administrator-assigned name.

Disk groups provide a method to partition the configuration database, so that the database size is not too large and so that database modifications do not affect too many drives. They also allow the Logical Storage Manager to operate with groups of physical disk media that can be moved between systems.

Disks and disk groups have a circular relationship: disk groups are formed from disks, and disk group configurations are stored on disks. All disks in a disk group are stamped with a disk group ID, which is a unique identifier for naming disk groups. Some or all disks in a disk group also store copies of the configuration of the disk group.

disk group configuration

A disk group configuration is a small database that contains all volume, plex, subdisk, and disk media records. These configurations are replicated onto some or all disks in the disk group, often with two copies on each disk. Because these databases are stored within disk groups, record associations cannot span disk groups. Thus, a subdisk defined on a disk in one disk group cannot be associated with a volume in another disk group.

disk group ID

A 64 byte universally unique identifier that is assigned to a disk group when the disk group is created with `/sbin/voldg init`. This identifier is in addition to the disk group name, which is assigned by the administrator. The disk group ID is used to check for disk groups that have the same administrator-assigned name but are actually distinct.

disk group records

Disk group records define several different types of names for a disk group. The different types of names are as follows:

- The *real name* is the name of the disk group, as the name is defined on disk. This name is stored in the disk group configuration, and is also stored in the disk headers of all disks in the disk group.
- The *alias name* is the standard name that the system uses when referencing the disk group. References to the disk group name usually mean the alias name. Volume and plex device directories are structured into subdirectories based on the disk group alias name. Typically, the disk group's alias name and real name are identical. A local alias can be useful for gaining access to a disk group with a name that conflicts with other disk groups in the system, or that conflicts with records in the `rootdg` disk group.
- The *disk group ID* is a 64-byte identifier that represents the unique ID of the disk group. All disk groups on all systems should have a different disk group ID, even if they have the same real name. This identifier is stored in the disk headers of all disks in the disk group. It is used to ensure that the Logical Storage Manager does not confuse two disk groups which were created with the same name.

disk header

A block stored in a private region of a disk and that defines several properties of the disk. The disk header defines the size of the private region, the location and size of the public region, the unique disk ID for the disk, the disk group ID and disk group name (if the disk is currently associated with a disk group), and the host ID for a host that has exclusive use of the disk.

disk ID

A 64 byte universally unique identifier that is assigned to a physical disk when its private region is initialized with the `/sbin/voldisk init` operation. The disk ID is stored in the disk media record so that the physical disk can be related to the disk media record at system startup.

disk media record

A reference to a physical disk, or possibly a disk partition. This record can be thought of as a physical disk identifier for the disk or partition. Disk media records are configuration records that provide a name (known as the disk media name or DM name) that an administrator can use to reference a particular disk independent of its location on the system's various disk controllers. Disk media records reference particular physical disks through a disk ID, which is a unique identifier that is assigned to a disk when it is initialized for use with the Logical Storage Manager.

Operations are provided to set or remove the disk ID stored in a disk media record. Such operations have the effect of removing or replacing disks, with any associated subdisks being removed or replaced along with the disk.

host ID

A name, usually assigned by the administrator, that identifies a particular host. Host IDs are used to assign ownership to particular physical disks. When a disk is part of a disk group that is in active use by a particular host, the disk is stamped with that host's host ID. If another system attempts to access the disk, it will detect that the disk has a non-matching host ID and will disallow access until the first system discontinues use of the disk. To allow for system failures that do not clear the host ID, the `/sbin/voldisk clearimport` operation can be used to clear the host ID stored on a disk.

If a disk is a member of a disk group and has a host ID that matches a particular host, then that host will import the disk group as part of system startup.

kernel log

A log kept in the private region on the disk and that is written by the Logical Storage Manager kernel. The log contains records describing the state of volumes in the disk group. This log provides a mechanism for the kernel to persistently register state changes so that `vold` can be guaranteed to detect the state changes even in the event of a system failure.

plex

A copy of a volume's logical data address space, also sometimes known as a *mirror*. A volume can have up to eight plexes associated with it. Each plex is, at least conceptually, a copy of the volume that is maintained consistently in the presence of volume I/O and reconfigurations. Plexes represent the primary means of configuring storage for a volume. Plexes can have a striped or concatenated organization (layout).

plex consistency

If the plexes of a volume contain different data, then the plexes are said to be inconsistent. This is only a problem if the Logical Storage Manager is unaware of the inconsistencies, as the volume can return differing results for consecutive reads. Plex inconsistency is a serious compromise of data integrity. This inconsistency can be caused by write operations that start around the time of a system failure, if parts of the write complete on one plex but not the other. Plexes can also be inconsistent after creation of a mirrored volume, if the plexes are not first synchronized to contain the same data. An important part of Logical Storage Manager operation is ensuring that consistent data is returned to any application that reads a volume. This may require that plex consistency of a volume be ‘‘recovered’’ by copying data between plexes so that they have the same contents. Alternatively, the volume can be put into a state such that reads from one plex are automatically written back to the other plexes, thus making the data consistent for that volume offset.

private region

Disks used by the Logical Storage Manager contain two special regions: a private region and a public region. Usually, each region is formed from a complete partition of the disk; however, the private and public regions can be allocated from the same partition.

The private region of a disk contains various on-disk structures that are used by the Logical Storage Manager for various internal purposes. Each private region begins with a disk header which identifies the disk and its disk group. Private regions can also contain copies of a disk group's configuration, and copies of the disk group's kernel log.

public region

The public region of a disk is the space reserved for allocating subdisks. Subdisks are defined with offsets that are relative to the beginning of the public region of a particular disk. Only one contiguous region of disk can form the public region for a particular disk.

read policy

A configurable policy for switching between plexes for volume reads. When a volume has more than one enabled associated plex, the Logical Storage Manager can distribute reads between the plexes to distribute the I/O load and thus increase total possible bandwidth of reads through the volume. The read policy can be set by administrator. Possible policies are:

- round-robin

  For every other read operation, switch to a different plex from the previous read operation. Given three plexes, this will switch between each of the three plexes, in order.

  preferred plex

  This read policy specifies a particular named plex that is used to satisfy read requests. In the event that a read request cannot be satisfied by the preferred plex, this policy changes to round-robin.

- select

  This read policy is the default policy, and adjusts to use an appropriate read policy based on the set of plexes associated with the volume. If exactly one enabled read-write striped plex is associated with the volume, then that plex is chosen automatically as the preferred plex; otherwise, the round-robin policy is used. If a volume has one striped plex and one non-striped plex, preferring the striped plex often yields better throughput.

root disk group

Each system requires one special disk group, named `rootdg`. This group is generally the default for most utilities. In addition to defining the regular disk group information, the configuration for the root disk group contains local information that is specific to a disk group and that is not intended to be movable between systems.

striped plex

A plex that scatters data evenly across each of its associated subdisks. A plex has a characteristic number of stripe columns (represented by the number of associated subdisks) and a characteristic stripe width. The stripe width defines how data with a particular address is allocated to one of the associated subdisks. Given a stripe width of 128 blocks, and two stripe columns, the first group of 128 blocks would be allocated to the first subdisk, the second group of 128 blocks would be allocated to the second subdisk, the third group to the first subdisk, again, and so on.

subdisk

A region of storage allocated on a disk for use with a volume. Subdisks are associated to volumes through plexes. One or more subdisks are layout to form plexes based on the plex layout: striped or concatenated. Subdisks are defined relative to disk media records.

`volboot` file

The `volboot` file is a special file (usually stored in `/etc/vol/volboot`) that is used to bootstrap the root disk group and to define a system's host ID. In addition to a host ID, the `volboot` file contains a list of disk access records. On system startup, this list of disks is scanned to find a disk that is a member of the `rootdg` disk group and that is stamped with this system's host ID. When such a disk is found, its configuration is read and is used to get a more complete list of disk access records that are used as a second-stage bootstrap of the root disk group, and to locate all other disk groups.

volume

A virtual disk device that looks to applications and file systems like a regular disk partition device. Volumes present block and raw device interfaces that are compatible in their use, with disk partition devices. However, a volume is a virtual device that can be mirrored, spanned across disk drives, moved to use different storage, and striped using administrative commands. The configuration of a volume can be changed, using the Logical Storage Manager utilities, without causing disruption to applications or file systems that are using the volume.

volume records

Volume records define the characteristics of particular volume devices. The name of a volume record defines the node name used for files in the `/dev/vol` and `/dev/rvol` directories. The block device for a particular volume (which can be used as an argument to the `mount` command (see `mount`(8)) has the path `/dev/vol/`*`groupname/volume`*.

Index

A

D

E

G

H

I

M

R

W

Z

How to Order Additional Documentation

Technical Support

If you need help deciding which documentation best meets your needs, call 800-DIGITAL (800-344-4825) before placing your electronic, telephone, or direct mail order.

Electronic Orders

To place an order at the Electronic Store, dial 800-234-1998 using a 1200- or 2400-bps modem from anywhere in the USA, Canada, or Puerto Rico. If you need assistance using the Electronic Store, call 800-DIGITAL (800-344-4825).

Telephone and Direct Mail Orders

Your Location	Call	Contact
Continental USA, Alaska, or Hawaii	800-DIGITAL	Digital Equipment Corporation P.O. Box CS2008 Nashua, New Hampshire 03061
Puerto Rico	809-754-7575	Local Digital subsidiary
Canada	800-267-6215	Digital Equipment of Canada Attn: DECdirect Operations KAO2/2 P.O. Box 13000 100 Herzberg Road Kanata, Ontario, Canada K2K 2A6
International	—————	Local Digital subsidiary or approved distributor
Internal[a]	—————	SSB Order Processing – NQO/V19 *or* U. S. Software Supply Business Digital Equipment Corporation 10 Cotton Road Nashua, NH 03063-1260

[a] For internal orders, you must submit an Internal Software Order Form (EN-01740-07).

Reader's Comments

Digital UNIX
Logical Storage Manager
AA-Q3NCE-TE

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

- This postage-paid form
- Internet electronic mail: `readers_comment@zk3.dec.com`
- Fax: (603) 881-0120, Attn: UEG Publications, ZKO3-3/Y32

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

Please rate this manual:	Excellent	Good	Fair	Poor
Accuracy (software works as manual says)				
Completeness (enough information)				
Clarity (easy to understand)				
Organization (structure of subject matter)				
Figures (useful)				
Examples (useful)				
Index (ability to find topic)				
Usability (ability to access information quickly)				

Please list errors you have found in this manual:

Page Description

Additional comments or suggestions to improve this manual:

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

Name/Title Dept.
Company Date
Mailing Address
Email Phone

Do Not Cut or Tear − Fold Here and Tape

digital™

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY MAIL

FIRST−CLASS MAIL PERMIT NO. 33 MAYNARD MA

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
UEG PUBLICATIONS MANAGER
ZKO3−3/Y32
110 SPIT BROOK RD
NASHUA NH 03062−9987

Do Not Cut or Tear − Fold Here

Cut on Dotted Line