HP-UX System Administrator's Guide: Logical Volume Management

HP-UX 11i Version 3

HP Part Number: 5992-4589
Published: September 2008
Edition: 3

Table of Contents

List of Figures

List of Tables

About This Document

HP-UX System Administrator's Guide: Logical Volume Management describes how to configure, administer, and troubleshoot the Logical Volume Manager (LVM) product on HP-UX 11i Version 3 platforms.

Intended Audience

The *HP-UX System Administrator's Guide* is written for administrators of HP-UX systems of all skill levels needing to administer HP-UX systems beginning with HP-UX Release 11i Version 3.

While many topics in this set apply to previous releases, much has changed in HP-UX 11i Version 3. Therefore, for information about prior releases, see *Managing Systems and Workgroups: A Guide for System Administrators*.

New and Changed Information in This Edition

The third edition of *HP-UX System Administrator's Guide: Logical Volume Management* addresses the following new topics:

- Version 2.1 volume groups, introduced in the September 2008 release of HP-UX 11i Version 3. See "LVM Volume Group Versions" (page 18).
- Recommendations for provisioning volume groups. See Appendix C (page 131).

About this Series

The *HP-UX System Administrator's Guide* documents the core set of tasks (and associated concepts) necessary to administer systems running HP-UX 11i Version 3.

The *HP-UX System Administrator's Guide* is a set of documents comprised of the following volumes:

Overview	Provides a high-level view of HP-UX 11i, its components, and how they relate to each other.
Configuration Management	Describes many of the tasks you must perform to configure and customize system settings and the behavior of subsystems.
Logical Volume Management	Documents how to configure physical volumes, volume groups, and logical volumes using the HP Logical Volume Manager (LVM).
Security Management	Documents the data and system security features of HP-UX 11i.
Routine Management Tasks	Documents many of the ongoing tasks you must perform to keep your system running smoothly.

Document Organization

This document is organized as follows:

Chapter 1: Introduction	Use this chapter to learn about LVM concepts.
Chapter 2: Configuring LVM	Use this chapter to learn how to configure a system to use LVM.
Chapter 3: Administering LVM	Use this chapter to learn how to administer LVM on an ongoing basis.
Chapter 4: Troubleshooting LVM	Use this chapter to learn how to troubleshoot problems in an LVM configuration.
Appendix A: LVM Specifications and Limitations	Use this appendix to find product specifications.

Appendix B: LVM Command Summary	Use this appendix to find frequently used LVM commands.
Appendix C: Volume Group Provisioning Tips	Use this appendix to find recommendations for volume group creation parameters.
Glossary	Use this appendix to find definitions of frequently used LVM terms.

Typographic Conventions

This document uses the following typographical conventions:

`%`, `$`, or `#`	A percent sign represents the C shell system prompt. A dollar sign represents the system prompt for the Bourne, Korn, and POSIX shells. A number sign represents the superuser prompt.
audit(5)	A manpage. The manpage name is *audit*, and it is located in Section 5.
`Command`	A command name or qualified command phrase.
`Computer output`	Text displayed by the computer.
Ctrl+x	A key sequence. A sequence such as **Ctrl+x** indicates that you must hold down the key labeled **Ctrl** while you press another key or mouse button.
Document Title	The title of a document. On the web and on the Instant Information media, it may be a hot link to the document itself.
`ENVIRONMENT VARIABLE`	The name of an environment variable, for example, `PATH`.
[ERROR NAME]	The name of an error, usually returned in the `errno` variable.
Key	The name of a keyboard key. **Return** and **Enter** both refer to the same key.
Term	The defined use of an important word or phrase.
`User input`	Commands and other text that you type.
`Variable`	The name of a placeholder in a command, function, or other syntax display that you replace with an actual value.
[]	The contents are optional in syntax. If the contents are a list separated by |, you must choose one of the items.
{}	The contents are required in syntax. If the contents are a list separated by |, you must choose one of the items.
...	The preceding element can be repeated an arbitrary number of times.
◎	Indicates the continuation of a code example.
|	Separates items in a list of choices.
WARNING	A warning calls attention to important information that if not understood or followed will result in personal injury or nonrecoverable system problems.
CAUTION	A caution calls attention to important information that if not understood or followed will result in data loss, data corruption, or damage to hardware or software.
IMPORTANT	This alert provides essential information to explain a concept or to complete a task.
NOTE	A note contains additional information to emphasize or supplement important points of the main text.

Examples and Shells

This document describes practices used by the system administrator. Since the `root` user (superuser) is required to use the POSIX shell `/sbin/sh`, all command examples use that shell. The POSIX shell is defined in *sh-posix*(1). For information on other shells, see the *Shells User's Guide* and *sh*(1).

Command Syntax

`Literal`	A word or character that you enter literally.
`Replaceable`	A word or phrase that you replace with an appropriate value.
-*chars*	One or more grouped command options, such as `-ikx`. The *chars* are usually a string of literal characters that each represent a specific option. For example, the entry `-ikx` is equivalent to the individual options `-i`, `-k`, and `-x`. The plus character (+) is sometimes used as an option prefix.
-*word*	A single command option, such as `-help`. The *word* is a literal keyword. The difference from -*chars* is usually obvious and is clarified in an Options description. The plus character (+) and the double hyphen (`--`) are sometimes used as option prefixes.
[ ]	The bracket metacharacters enclose optional content in formats and command descriptions.
{ }	The brace metacharacters enclose required content in formats and command descriptions.
|	The bar metacharacter separates alternatives in a list of choices, usually in brackets or braces.
`...`	The ellipsis metacharacter after a token (`abc...`) or a right bracket (`[ ]...`) or a right brace (`{ }...`) metacharacter indicates that the preceding element and its preceding whitespace, if any, may be repeated an arbitrary number of times.
`...`	Ellipsis is sometimes used to indicate omitted items in a range.

Related Information

Additional information about LVM and HP-UX can be found at http://docs.hp.com in the *HP-UX Operating Environment* collection. In particular, the following white papers are available:

- *LVM Limits*
- *LVM Migration from Legacy to Agile Naming Model*
- *LVM Online Disk Replacement (LVM OLR)*
- *LVM Version 2.0 Volume Groups in HP-UX 11i v3*
- *LVM Volume Group Dynamic LUN expansion (DLE) / vgmodify*
- *LVM Volume Group Quiesce/Resume*
- *SLVM Single-Node Online Reconfiguration (SLVM SNOR)*
- *When Good Disks Go Bad: Dealing with Disk Failures Under LVM*

Finding HP-UX Information

The following table outlines where to find system administration information for HP-UX. This table does not include information for specific products.

Table 1 Finding HP-UX Information

If you need to	Refer To	Located at .
Find out: • What has changed in HP-UX releases • The contents of the Operating Environments • Firmware requirements and supported systems for a specific release	The HP-UX 11i Release Notes specific to your version of HP-UX.	• HP Instant Information media • HP-UX Technical Documentation website at:http://docs.hp.com
Install or update HP-UX	• *Read Before Installing or Updating to HP-UX* • *HP-UX 11i Installation and Update Guide*	• Media Kit (supplied with the Operating Environment) • HP Instant Information media • HP-UX Technical Documentation website at:http://docs.hp.com
Administer an HP-UX system	Releases beginning with HP-UX 11i Version 3: • *HP-UX System Administrator's Guide* (a multivolume set) Other sources of system administration information: • *nPartition Administrator's Guide* • Planning Superdome Configurations White Paper	• HP Instant Information media • HP-UX Technical Documentation website at:http://docs.hp.com • "Planning Superdome Configurations" is available at http://docs.hp.com/hpux/onlinedocs/os/11i/superdome.pdf

HP-UX 11i Release Names and Operating System Version Identifiers

With HP-UX 11i, HP delivers a highly available, secure, and manageable operating system that meets the demands of end-to-end Internet-critical computing. HP-UX 11i supports enterprise, mission-critical, and technical computing environments. HP-UX 11i is available on both HP 9000 systems and HP Integrity systems.

Each HP-UX 11i release has an associated release name and release identifier. The `uname` command with the `-r` option returns the release identifier. Table 2 shows the releases available for HP-UX 11i.

Table 2 HP-UX 11i Releases

OS Version Identifier	Release Name	Supported Processor Architecture
B.11.11	HP-UX 11i Version 1	HP 9000
B.11.23	HP-UX 11i Version 2	Integrity
B.11.23.0409	HP-UX 11i Version 2 September 2004 Update	HP 9000 and Integrity
B.11.31	HP-UX 11i Version 3	HP 9000 and Integrity

For information on supported systems and processor architecture for various versions of HP-UX 11i, see the HP-UX 11i system release notes specific to your version of HP-UX.

Determining Your System Version

The `uname`, `model`, and `swlist` commands can help you determine information about your system, including its hardware type, machine model, operating system version, and operating environment update status. See *uname*(1), *model*(1), and *swlist*(1M).

For OS naming conventions, see "HP-UX 11i Release Names and Operating System Version Identifiers" (page 12).

Table 3 OS Version, System Architecture, and Machine Model

Topic	Command	Sample Output
OS Version	`$ uname -r`	`B.11.31`[1]
Architecture	`$ uname -m`	`ia64`[2] `9000/800`[2]
Machine Model	`$ model`[3]	`ia64 hp server rx5670` `9000/800/S16K-A`
Operating Environment	`$ swlist HPUX*OE*`	`# HPUX11i-OE-MC B.11.31 HP-UX Mission Critical Operating Environment`[1]
OS Version.Update	`$ swlist HPUX*OE*`	`# HPUX11i-TCOE B.11.23.0409 HP-UX Technical Computing OE Component`[1]

1 HP-UX 11i OS version identifiers have the form B.11.23 or B.11.23.0409, where B.11.23 is the OS version and 0409 is the year-month of the operating environment (OE) update.

2 `ia64` = Integrity. All others = HP 9000.

3 The `getconf MACHINE_MODEL` command gives the same output. See *getconf*(1).

Publication History

The document printing date and part number indicate the document's current edition. The printing date changes when a new edition is printed. Minor changes can be made at reprint without changing the printing date. The document part number changes when extensive changes are made. Document updates can be issued between editions to correct errors or document product changes. To ensure that you receive the updated or new editions, subscribe to the appropriate product support service. See your HP sales representative for details. You can find the latest version of this document online at:

http://www.docs.hp.com.

Manufacturing Part Number	Supported Operating Systems	Supported Versions	Edition Number	Publication Date
5991-6481	HP-UX	HP-UX 11i Version 3	1	February 2007
5992-3385	HP-UX	HP-UX 11i Version 3	2	March 2008
5992-4589	HP-UX	HP-UX 11i Version 3	3	September 2008

NOTE: The volumes in the *HP-UX System Administrator's Guide* can be updated independently. Therefore, the latest versions of the volumes in the set can vary with time and with respect to each other. The latest versions of each volume are available at http://docs.hp.com.

HP Encourages Your Comments

HP encourages your comments concerning this document. We are committed to providing documentation that meets your needs. Send any errors found, suggestions for improvement, or compliments to:

http://docs.hp.com/en/feedback.html

Include the document title, manufacturing part number, and any comment, error found, or suggestion for improvement you have concerning this document.

1 Introduction

This chapter addresses the following topics:

LVM Features

Logical Volume Manager (LVM) is a storage management system that lets you allocate and manage disk space for file systems or raw data. Historically, you treated your disks individually and in terms of fixed-sized partitions; each disk or partition held a file system, swap space, boot area, or raw data. With LVM, you do not need to assign a disk or fixed-sized partition to a single purpose. Instead, consider the disks as a pool (or volume) of data storage, consisting of equal-sized extents. Extents are allocated into virtual storage devices known as *logical volumes*, which can be treated as disks.

LVM provides the following capabilities:

- A logical volume size can be dynamically reduced or expanded to meet changing data needs. For example, a logical volume can be as small or large as the file system mounted to it requires. The file system can be extended without rebuilding it or the logical volume; reducing a file system is more complex, and may require recreating the file system.
- Small chunks of unused space from several disks can be combined to create a usable volume.
- A logical volume can exceed the size of a single physical disk. This feature is called *disk spanning*, because a single file system (and individual files) can span disks.
- Up to six copies of identical data can be stored and updated simultaneously using LVM. This feature is called *mirroring* a logical volume, and requires an optional product, HP MirrorDisk/UX. See "Increasing Data Availability Through Mirroring" (page 29).
- Mirrored data can be configured to automatically create a new mirror to a separate disk when one of the mirror copies fails. This feature is called **sparing**, and requires an optional product, HP MirrorDisk/UX. See "Increasing Disk Redundancy Through Disk Sparing" (page 32).
- A logical volume can be created so that logically contiguous data blocks (for example, chunks of the same file) are distributed across multiple disks, which speeds I/O throughput for large files when they are read and written sequentially. This feature is called **striping**. Striping can be used in conjunction with mirroring. See "Increasing Performance Through Disk Striping" (page 35).
- Devices accessed through multiple links can be configured to improve availability. If the primary link to a device fails, LVM can switch automatically to an alternate link. This feature is called **multipathing**. See "Increasing Hardware Path Redundancy Through Multipathing" (page 33).

LVM Architecture

An LVM system starts by initializing disks for LVM usage. An LVM disk is known as a *physical volume* (PV). A disk is marked as an LVM physical volume using either the HP System Management Homepage (HP SMH) or the `pvcreate` command. Physical volumes use the same device special files as traditional HP-UX disk devices.

LVM divides each physical volume into addressable units called *physical extents* (PEs). Starting after the LVM metadata at the beginning of the disk, extents are allocated sequentially, with an index starting at zero and incrementing by one for each unit. The physical extent size is configurable at the time you form a volume group and applies to all disks in the volume group. You can select a size from 1 MB to 256 MB.

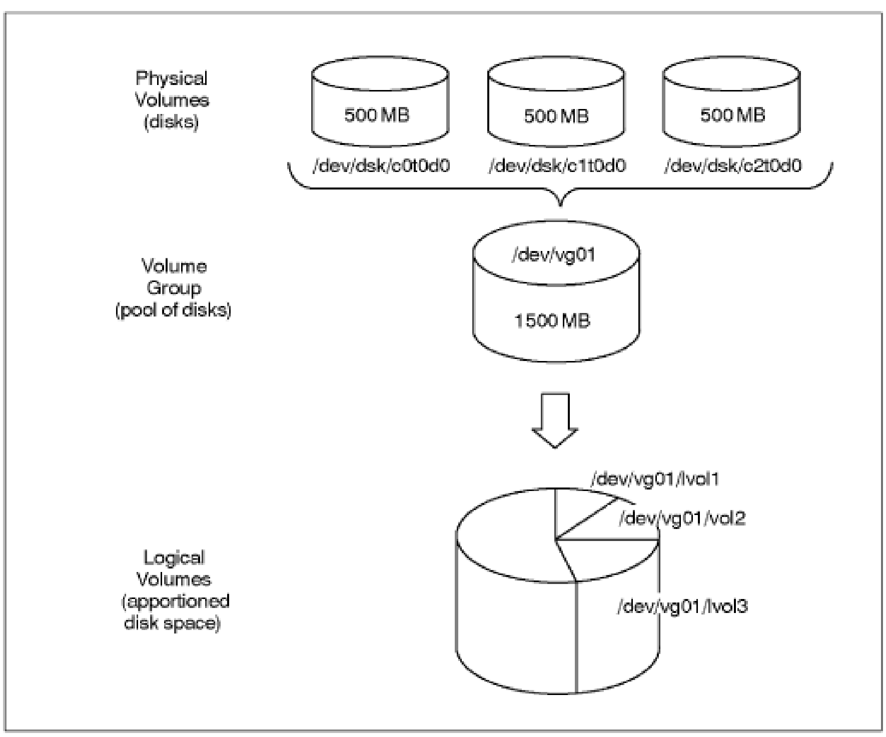
Physical volumes are organized into *volume groups* (VGs). A volume group can consist of one or more physical volumes, and there can be more than one volume group in the system. Once created, the volume group, not the disk, is the entity that represents data storage. Thus, whereas earlier you moved disks from one system to another, with LVM, you move a volume group from one system to another. Therefore, it is often convenient to have multiple volume groups on a system.

The pool of disk space that is represented by a volume group can be divided into *logical volumes* (LVs) of various sizes. Once created, logical volumes can be treated just like disk partitions. They are accessible through device special files. A logical volume can span a number of physical volumes in a volume group or represent only part of one physical volume.

The basic allocation units for a logical volume are called *logical extents* (LEs). A logical extent is mapped to a physical extent. Thus, if the physical extent size is 4 MB, the logical extent size is also 4 MB. The size of a logical volume is determined by the number of logical extents configured.

You assign file systems, swap, dump, or raw data to logical volumes. For example, in Figure 1-1, logical volume `/dev/vg01/lvol1` might contain a file system, logical volume `/dev/vg01/lvol2` might contain swap space, and logical volume `/dev/vg01/lvol3` might contain raw data. You can use HP SMH to create a file system in a logical volume of a specified size, then mount the file system. Alternately, you can use LVM commands to create, then extend a logical volume to allocate sufficient space for file systems or raw data. You then create and mount new file systems or install your application in the logical volume.

Figure 1-1 Disk Space Partitioned Into Logical Volumes



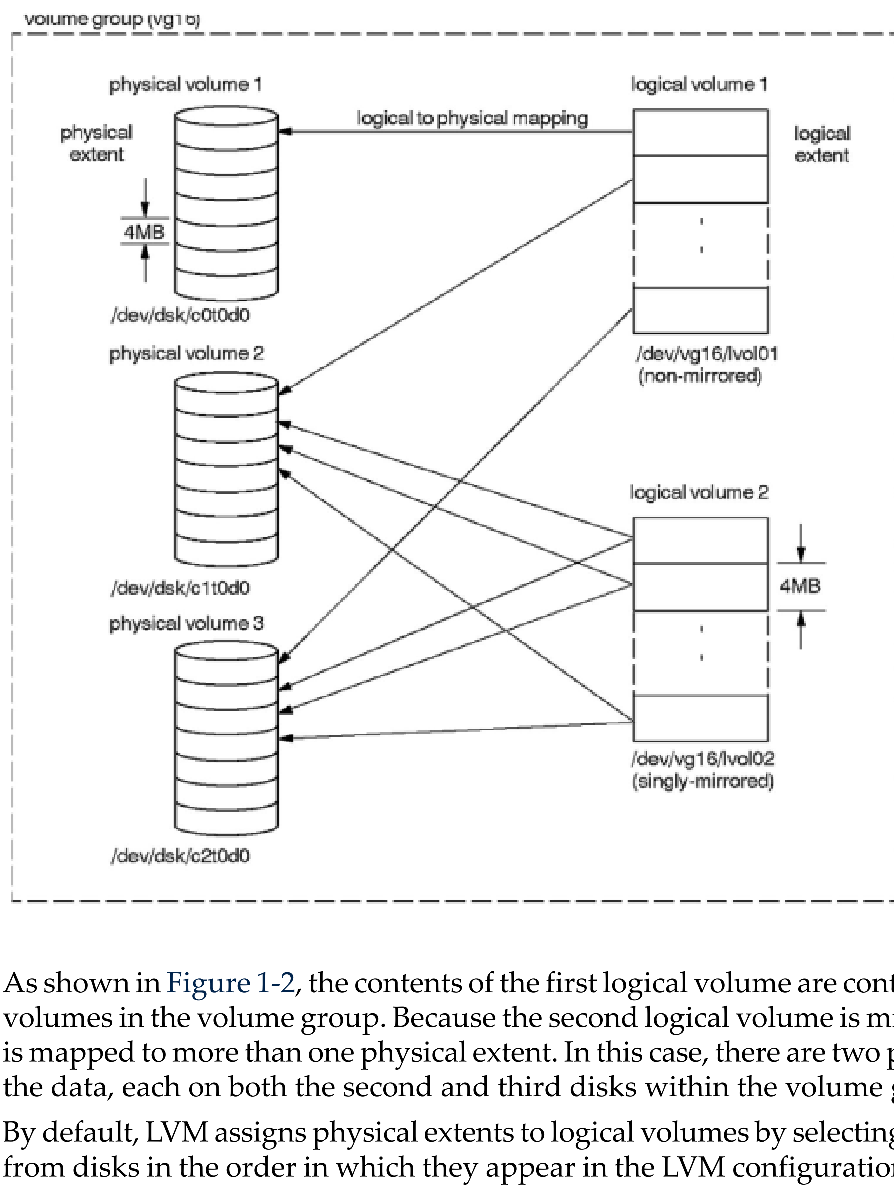
Physical versus Logical Extents

When LVM allocates disk space to a logical volume, it automatically creates a mapping of the logical extents to physical extents. This mapping depends on the policy chosen when creating the logical volume. Logical extents are allocated sequentially, starting at zero, for each logical volume. LVM uses this mapping to access the data, regardless of where it physically resides. Commands are provided for you to examine this mapping; see *pvdisplay*(1M) and *lvdisplay*(1M).

Except for mirrored, striped, or striped-mirrored logical volumes, each logical extent is mapped to one physical extent. For mirrored logical volumes, each logical extent is mapped to multiple physical extents, depending on the number of mirror copies. For example, if one mirror copy exists, then each logical extent maps to two physical extents, one extent for the original and one for the mirror copy. For more information on mirroring, see "Increasing Data Availability Through Mirroring" (page 29). For information on striped logical volumes, see "Increasing Performance Through Disk Striping" (page 35). Also refer to the book *Disk and File Management Tasks on HP-UX*.

Figure 1-2 shows an example of several types of mapping available between physical extents and logical extents within a volume group.

Figure 1-2 Physical Extents and Logical Extents



As shown in Figure 1-2, the contents of the first logical volume are contained on all three physical volumes in the volume group. Because the second logical volume is mirrored, each logical extent is mapped to more than one physical extent. In this case, there are two physical extents containing the data, each on both the second and third disks within the volume group.

By default, LVM assigns physical extents to logical volumes by selecting available physical extents from disks in the order in which they appear in the LVM configuration files, `/etc/lvmtab` and `/etc/lvmtab_p`. As a system administrator, you can bypass this default assignment and control which disks are used by a logical volume (see "Extending a Logical Volume to a Specific Disk" (page 51)).

If a logical volume is to be used for root, boot, primary swap, or dump, the physical extents must be **contiguous**, which means that the physical extents must be allocated in increasing order with no gaps on a single physical volume. For logical volumes that are not being used for root, boot, primary swap or dump, physical extents that correspond to contiguous logical extents within a logical volume can be **noncontiguous** on a physical volume or reside on entirely different disks. As a result, a file system created within one logical volume can reside on more than one disk.

LVM Volume Group Versions

As of the September 2008 release of HP-UX 11i Version 3, LVM supports three versions of volume groups. All information and tasks in this document apply to all volume group versions except where noted.

Version 1.0 is the version supported on all current and previous versions of HP-UX 11i. The procedures and command syntax for managing Version 1.0 volume groups are unchanged from previous releases. When creating a new volume group, `vgcreate` defaults to Version 1.0.

Version 2.0 and Version 2.1 enable the configuration of larger volume groups, logical volumes, physical volumes, and other parameters. Version 2.1 is identical to Version 2.0, but allows a greater number of volume groups, physical volumes, and logical volumes. Version 2.x volume groups are managed exactly like Version 1.0 volume groups, with the following exceptions:

- Version 2.x volume groups have simpler options to the `vgcreate` command. When creating a Version 2.x volume group, you specify only the extent size and the maximum size to which the volume group can grow. This gives LVM greater flexibility in managing space; you can use the same parameters for a volume group with many small PVs and for a volume group with a few large PVs. For more information on volume group creation, see "Creating a Volume Group" (page 46)
- Version 2.0 volume groups are not recognized on previous releases of HP-UX, including versions of HP-UX 11i Version 3 before March 2008. Version 2.1 volume groups are not recognized on previous releases of HP-UX, including versions of HP-UX 11i Version 3 before September 2008.
- Version 2.x volume groups do not support root, boot, swap, or dump logical volumes. The `lvlnboot` and `lvrmboot` commands display an error message if run on a Version 2.x volume group.
- Version 2.x volume groups do not support bootable physical volumes. You cannot add a physical volume created with `pvcreate -B` to a Version 2.x volume group.
- Version 2.x volume groups do not support disk sparing. Using the `-z` option to the `vgextend` or `pvchange` command displays an error message.
- The `pvck` and `vgmodify` commands are not supported on Version 2.x volume groups.
- Some HP-UX products do not support Version 2.x volume groups. For more information, see the *HP-UX Logical Volume Manager and MirrorDisk/UX Release Notes* for your release.

Table 1-1 compares Version 1.0 and Version 2.x volume groups:

Table 1-1 LVM Volume Group Version Properties

	Version 1.0 Volume Groups	Version 2.0 Volume Groups	Version 2.1 Volume Groups
Maximum number of volume groups on a system	256	512[1]	2048[1]
Maximum number of physical volumes in a volume group	255	511	2048
Maximum number of logical volumes in a volume group	255	511	2047

Table 1-1 LVM Volume Group Version Properties *(continued)*

	Version 1.0 Volume Groups	Version 2.0 Volume Groups	Version 2.1 Volume Groups
Maximum size of a physical volume	2 TB	16 TB	16 TB
Maximum size of a volume group	510 TB	2048 TB	2048 TB
Maximum size of a logical volume	16 TB	256 TB	256 TB
Maximum size of a physical extent	256 MB	256 MB	256 MB
Maximum size of a stripe	32 MB	256 MB	256 MB
Maximum number of stripes	255	511	511
Maximum number of logical extents per logical volume	65535	33554432	33554432
Maximum number of physical extents per physical volume	65535	16777216	16777216
Number of mirror copies (MirrorDisk/UX product required)	0–2	0–5	0–5
LVM configuration file	`/etc/lvmtab`	`/etc/lvmtab_p`	`/etc/lvmtab_p`
Device file major number	64	128	128

1 The limit of 2048 volume groups is shared between Version 2.0 and Version 2.1 volume groups. Volume groups of both versions can be created with volume group numbers ranging from 0-2047. However, the maximum number of Version 2.0 volume groups that can be created is 512.

You can display the volume group limits with the `lvmadm` command. For more information, see *lvmadm*(1M).

LVM Device File Usage

All LVM components are represented by device special files located in the `/dev` directory. Device special files act as agents for managing the interactions with the disk space. The LVM device files are created by both HP SMH and HP-UX commands. This section describes the device special files used by LVM, and naming conventions for LVM objects.

Legacy Device Files versus Persistent Device Files

As of HP-UX 11i Version 3, disk devices can be represented by two different types of device files in the `/dev` directory, **legacy** and **persistent**.

Legacy device files were the only type of mass storage device files in releases prior to HP-UX 11i Version 3. They have hardware path information such as SCSI bus, target, and LUN encoded in the device file name and minor number. For example, the legacy device file `/dev/dsk/c3t2d0` represents the disk at card instance 3, target address 2, and lun address 0.

Persistent device files are not tied to the physical hardware path to a disk, but instead map to the disk's unique worldwide identifier (WWID). Thus, the device file is unchanged if the disk is moved from one interface to another, moved from one switch or hub port to another, or presented from a different target port to the host. The name of a persistent device file follows a simpler naming convention: `/dev/disk/disk`*`n`*, where *`n`* is the instance number assigned to the disk. Neither the device file name nor the minor number contain any hardware path information.

In addition, if the disk has multiple hardware paths, it is represented by a single persistent device file. Persistent device files transparently handle multipathed disks and supersede LVM's multipathing functionality described in "Increasing Hardware Path Redundancy Through Multipathing" (page 33). If a disk has multiple hardware paths, which LVM refers to as **pvlinks**, the persistent device special file acts as a single access point for all the links. I/O requests are distributed across all available links by the mass storage stack, with a choice of load balancing algorithms. If a link fails, the mass storage stack automatically disables the failed link and I/O continues on all remaining links. Any failed or nonresponsive links are monitored, so that when a failed link recovers, it is automatically and transparently reincorporated into any load balancing. New disks and links are also automatically discovered and added to load balancing. If the disk's connectivity changes—addition, removal, or modification of a link—applications using the persistent device file are not affected, provided at least one link is still active. New disks are automatically discovered.

You can use either persistent or legacy device files for LVM disks. LVM recommends the use of persistent device special files, because they support a greater variety of load balancing options.

NOTE: To use LVM's alternate link functionality, you must use legacy device files, and disable multipathing through those legacy device files, as described in "Increasing Hardware Path Redundancy Through Multipathing" (page 33).

Naming Conventions for LVM

You must refer to LVM devices or volume groups by name when using them within HP SMH or with HP-UX commands. By default, the LVM device files created by both HP SMH and HP-UX commands follow a standard naming convention. However, you can choose customized names for volume groups and logical volumes.

Physical Volume Names

Physical volumes are identified by their device file names, as follows:

Table 1-2 Physical Volume Naming Conventions

Device File Name	Type of Device
`/dev/disk/disk`*n*	Persistent block device file
`/dev/disk/disk`*n*`_p2`	Persistent block device file, partition 2
`/dev/rdisk/disk`*n*	Persistent character device file
`/dev/rdisk/disk`*n*`_p2`	Persistent character device file, partition 2
`/dev/dsk/c`*n*`t`*n*`d`*n*	Legacy block device file
`/dev/dsk/c`*n*`t`*n*`d`*n*`s2`	Legacy block device file, partition 2
`/dev/rdsk/c`*n*`t`*n*`d`*n*	Legacy character device file
`/dev/rdsk/c`*n*`t`*n*`d`*n*`s2`	Legacy character device file, partition 2

Each disk has a block device file and a character or raw device file, the latter identified by the `r`. Which name you use depends on what task you are doing with the disk.

For the boot disk on HP Integrity servers, make sure to use the device files with the `_p2` suffix or `s2` suffix, because they represent the HP-UX partition on the boot disk. On HP 9000 servers, use the device file without a partition number.

Use a physical volume's raw device file for the following tasks only:

- Preparing a physical volume for LVM using the `pvcreate` command. Here you use the device file for the disk; for example, `/dev/rdisk/disk14`. (The absence of a partition suffix indicates that you are referring to the entire disk.)
- Removing LVM information from a physical volume using the `pvremove` command.
- Restoring your volume group configuration using the `vgcfgrestore` command.
- Performing a consistency check on a physical volume using the `pvck` command.
- Modifying the volume group identifier on a physical volume using the `vgchgid` command.
- Changing the disk type of a physical volume using the `vgmodify` command.

For all other tasks, use the block device file. For example, when you add a physical volume to a volume group using the `vgextend` command, you use the disk's block device file for the disk, such as `/dev/disk/disk14`.

All disk device files are created automatically when a new disk is discovered. For more information, see *insf*(1M).

Volume Group Names

Each volume group must have a unique name, up to 255 characters. For example, typical volume group names are `vg01`, `vgroot`, or `vg_sales`. Although the name does not need to start with `vg`, HP recommends using this prefix. By default, HP SMH uses the names of the form `/dev/vg`*`nn`*. The number *`nn`* starts at 00 and is incremented in the order that volume groups are created. By default, the root volume group is `vg00`.

Logical Volume Names

Logical volumes are identified by their device file names, which can either be assigned by you or assigned by default when you create a logical volume using the `lvcreate` command.

When assigned by you, you can choose any name up to 255 characters.

When assigned by default, these names take the form `/dev/vg`*`nn`*`/lvol`*`N`* (the block device file form) and `/dev/vg`*`nn`*`/rlvol`*`N`* (the character device file form). The number *`N`* starts at 1 and increments in the order that logical volumes are created within each volume group.

When LVM creates a logical volume, it creates both block and character device files. LVM then places the device files for a logical volume in the appropriate volume group directory. For example, the default block name for the first logical volume created in volume group `vg01` has the following full path name:

`/dev/vg01/lvol1`

If you create a logical volume to contain a sales database, you can name it explicitly as follows:

`/dev/vg01/sales_db_lv`

After the logical volume in the previous example is created, it has two device files: `/dev/vg01/sales_db_lv` for the block device file and `/dev/vg01/rsales_db_lv` for the character, or raw, device file.

Physical Volume Group Names

Physical volume groups are useful for mirroring and are discussed in "Increasing Performance Through I/O Channel Separation" (page 37). The only naming restriction is that within a volume group, each physical volume group must have its own unique name. For example, the volume group `/dev/vg02` might have two physical volume groups named `pvg1` and `pvg2`.

Device Number Format

The device files associated with LVM reside in the `/dev` directory. For each volume group, a directory under `/dev` is named after the volume group. In that directory is a single "group" device file and separate block and character device files for each logical volume.

The following is a sample listing:

```
# ls -l /dev/vg01
total 0
crw-r--r--    1 root     root      64 0x010000 Mar 28  2004 group
brw-r-----    1 root     root      64 0x010001 Jul 29 16:53 lvol1
brw-r-----    1 root     root      64 0x010002 Jul 29 16:53 lvol2
crw-r-----    1 root     root      64 0x010001 Mar 28  2004 rlvol1
crw-r-----    1 root     root      64 0x010002 Mar 28  2004 rlvol2
```

By default, volume group numbering begins with zero (`vg00`), while logical volumes begin with one (`lvol1`). This is because the logical volume number corresponds to the minor number and the volume group's group file is assigned minor number 0.

Physical volumes use the device files associated with their disk. LVM does not create device files for physical volumes.

Version 1.0 Device Number Format

Table 1-3 lists the format of the device file number for Version 1.0 volume groups.

Table 1-3 Version 1.0 Device Number Format

Major Number	Volume Group Number	Reserved	Logical Volume Number
64	0–0xff	0	0–0xff 0=group file

For Version 1.0 volume groups, the major number for LVM device files is 64. The volume group number is encoded into the top eight bits of the minor number, and the logical volume number is encoded into the low eight bits. Logical volume number 0 is reserved for the group file.

Version 2.x Device Number Format

Table 1-4 lists the format of the device file number for Version 2.x volume groups.

Table 1-4 Version 2.x Device Number Format

Major Number	Volume Group Number	Logical Volume Number
128	0–0x7ff	0–0x7ff 0=group file

For Version 2.x volume groups, the major number for LVM device files is 128. The volume group number is encoded into the top twelve bits of the minor number, and the logical volume number is encoded into the low twelve bits. Logical volume number 0 is reserved for the group file.

NOTE: The most significant bit of the volume group number and logical volume number fields are reserved and must be zero.

The device number format is subject to change.

LVM Disk Layout

NOTE: This information applies only to disks belonging to Version 1.0 volume groups.

There are two kinds of LVM disk layouts, one for boot disks and another for all other LVM disks. These differ in their data structures. Nonbootable disks have two reserved areas: the physical volume reserved area (PVRA) and the volume group reserved area (VGRA). Bootable disks have a PVRA and VGRA, and additional sectors reserved for the boot data reserved area (BDRA) and boot LIF.

Boot Data Reserved Area

The BDRA contains the information needed to configure the root, primary swap, and dump logical volumes, and to mount the root file system.

Information about the LVM disk data structures in the BDRA is maintained with the `lvlnboot` and `lvrmboot` commands. The following is a sample output:

```
# lvlnboot -v
Boot Definitions for Volume Group /dev/vg00:
Physical Volumes belonging in Root Volume Group:
                /dev/dsk/c3t0d0 -- Boot Disk
                /dev/dsk/c4t0d0 -- Boot Disk
                /dev/dsk/c5t0d0
                /dev/dsk/c12t0d0 -- Boot Disk
Root: lvol1     on: /dev/dsk/c3t0d0
                    /dev/dsk/c4t0d0
Swap: lvol2     on: /dev/dsk/c3t0d0
                    /dev/dsk/c4t0d0
Dump: lvol2     on: /dev/dsk/c3t0d0
```

The physical volumes designated "Boot Disk" are bootable, having been initialized with `mkboot` and `pvcreate -B`. Multiple lines for `lvol1` and `lvol2` indicate that the root and swap logical volumes are being mirrored.

Logical Interface Format Area

LVM boot disks contain a Logical Interface Format (LIF) area, in which is stored a `LABEL` file. On HP 9000 servers, the LIF area contains boot utilities such as the initial system loader (`ISL`), the kernel boot loader (`HPUX`), the autoboot file (`AUTO`), and offline diagnostics.

The `LABEL` file is created and maintained by `lvlnboot` and `lvrmboot`. It contains information about the starting point and size of boot-relevant logical volumes, including the boot file system (`/stand`). Utilities can use the `LABEL` file to access the root, primary swap, and dump logical volumes without actually using LVM.

Physical Volume Reserved Area

The physical volume reserved area (PVRA) contains information describing the physical volume, such as its unique identifier, physical extent information, and pointers to other LVM structures on the disk.

Volume Group Reserved Area

The volume group reserved area (VGRA) describes the volume group to which the disk belongs. The information is replicated on all of the physical volumes and updated whenever a configuration change is made. Among other data, it contains the following information:

- A list of physical volumes in the volume group, including physical volume status and size, and a map of physical extents to logical volumes.
- A list of logical volumes in the volume group (including the status and capabilities of each logical volume), its scheduling and allocation policies, and the number of mirror copies.
- A volume group header containing the VGID and three configurable parameters:
  - the number of physical volumes allowed in the volume group
  - the maximum number of logical volumes allowed in the volume group
  - the maximum number of physical extents allowed per physical volume

Since each physical extent is recorded in the VGRA, the extent size has a direct bearing on the size of the VGRA. In most cases, the default extent size is sufficient. However, if you encounter problems, consider that the VGRA is a fixed size and a high-capacity physical volume might exceed the total number of physical extents allowed. As a result, you might need to use a larger-than-default extent size on high-capacity LVM disks. Conversely, if all LVM disks in a volume group are small, the default number of extents might make the VGRA too large, wasting disk and memory space. A smaller-than-default extent size or number of physical extents might be preferable. A high-capacity physical volume might be unusable in a volume group whose extent size is small or set with a small number of physical extents per disk.

User Data Area

The user data area is the region of the LVM disk used to store all user data, including file systems, virtual memory system (swap), or user applications.

LVM Limitations

LVM is a sophisticated subsystem. It requires time to learn, it requires maintenance, and in rare cases, things can go wrong.

HP recommends using logical volumes as the preferred method for managing disks. Use LVM on file and application servers. On servers that have only a single disk and are used only to store the operating system and for swap, a "whole-disk" approach is simpler and easier to manage. LVM is not necessary on such systems.

By default, LVM configurations are automatically backed up every time you change them in `/etc/lvmconf`. Mirroring provides insurance against data loss that is not available under the whole-disk method.

Additional limitations to LVM include the following:

- Both LVM disks and non-LVM disks can exist simultaneously on your system, but a given disk or partition must be managed entirely by either LVM or non-LVM methods. That is, you cannot combine these techniques for use with a single disk or partition.
- On an HP Integrity server, LVM supports partitioning of the root disk and its mirrors only, and supports only one HP-UX partition on any disk.
- Floppy disks, optical disks, and CD-ROMs do not support logical volumes.
- You must use an LVM or VERITAS™ Volume Manager (VxVM) disk for your root disk.
- To use LVM, a disk must be first initialized into a physical volume.
- To be allocatable for storage, a physical volume must be assigned to a volume group.
- A physical volume can belong to only one volume group.
- The extent size of a volume group is fixed when the volume group is created. It cannot be changed without recreating the volume group.

2 Configuring LVM

By default, the LVM commands are already installed on your system. This chapter discusses issues to consider when setting up your logical volumes. It addresses the following topics:

Planning Your LVM Configuration

Using logical volumes requires some planning. Some of the issues to consider for planning purposes are discussed in this chapter. Consider these issues before setting up or modifying logical volumes on your system.

- For what purpose will you use a logical volume? For raw data or a file system? As a swap area or dump area? See "Setting Up Different Types of Logical Volumes" (page 25).
- How big will you make a logical volume?
- Does your data require high availability? If so, consider mirroring your logical volume across multiple disks, as described in "Increasing Data Availability Through Mirroring" (page 29). Also consider setting up spare disks to handle mirror failure, as described in "Increasing Disk Redundancy Through Disk Sparing" (page 32).
- Is I/O performance very important to you? If so, consider striping your logical volumes across multiple disks, as described in "Increasing Performance Through Disk Striping" (page 35), and separating I/O channels, as described in "Increasing Performance Through I/O Channel Separation" (page 37). For additional recommendations for performance, read "Planning for Performance" (page 33).
- Do you need to balance high availability and I/O performance? If so, consider striping and mirroring your logical volume, as described in "Increasing Data Availability Through Mirroring" (page 29) and "Increasing Performance Through Disk Striping" (page 35).
- Is it important to recover from disk failures quickly? If so, see "Planning for Recovery" (page 38).

Setting Up Different Types of Logical Volumes

This section contains information on setting up special logical volumes.

Setting Up Logical Volumes for Raw Data Storage

You can optimize raw I/O performance by planning your logical volumes specifically for raw data storage. To create a raw data logical volume (such as for a database), consider how large the logical volume must be and how such a logical volume is distributed over your disks.

Typically, you specify the size of a logical volume in megabytes. However, a logical volume must be a multiple of the extent size used in the volume group. For example, if a database partition requires 2002 MB and the logical extent size is 4 MB, LVM creates a logical volume that is 2004 MB (or 501 logical extents).

If you plan to use logical volumes heavily for raw data storage (such as for setting up database partitions), consider how the logical volumes are distributed over your disks.

By default, LVM assigns disk space for a logical volume from one physical volume, uses the space on this physical volume entirely, then assigns space from each successive physical volume in the same manner. LVM uses the physical volumes in the order in which they appear in `/etc/`

lvmtab and /etc/lvmtab_p, which means that data of a logical volume might not be evenly distributed over all the physical volumes within your volume group.

As a result, when I/O access to the logical volumes occurs, one or more disks within the volume group might be heavily used, while the others might be lightly used, or not used at all. This arrangement does not provide optimum I/O performance.

As a better alternative, you can set up your logical volume on specific disks in an interleaved manner, thus balancing the I/O access and optimizing performance (see "Extending a Logical Volume" (page 50)).

Because there are no HP-UX commands that identify that the contents of a logical volume are being used for raw data, create recognizable names for he logical volumes you create for raw data. In this way, you can recognize the contents of such a logical volume.

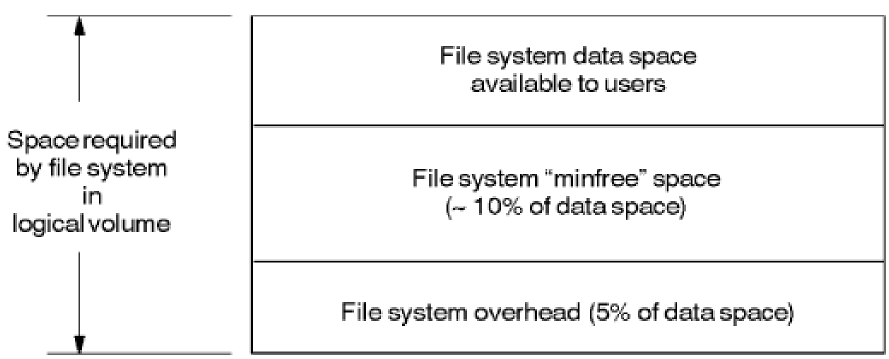
Setting Up Logical Volumes for File Systems

File systems reside in a logical volume just as they do within disk partitions or nonpartitioned disks. Two types of file systems can be used in a logical volume: Hierarchical File Systems (HFS) and Journaled File Systems (JFS) (VxFS).

Choosing the Initial Size of File System Logical Volumes

When determining the required space for a file system, consider the three major components shown in Figure 2-1.

Figure 2-1 File System Space Components



To estimate how big to make a logical volume that will contain your file system, follow these steps:

1. Estimate how much disk space users will need for their data in the future. Allow for any anticipated changes, which usually include additional growth. (Use the du command to see how much disk space is currently used.)
2. Add 10% to the above amount for a "minfree" area; this area is reserved to maintain performance.
3. Add another 5% for file system overhead; this includes all data structures required to maintain the file system.
4. Round up to the next integer multiple of the logical extent size used in this logical volume to find the size in logical extents. (This step is performed automatically when you create a logical volume.)

For example, if a group of users will require 60 MB space for file system data, the following estimate allows for expected growth. Add 6 MB for the minfree space. Then add 3 MB for file system overhead for a total estimate of 69 MB required by the file system, and for the logical volume that contains the file system. If you are creating the logical volume in a volume group that has an extent size of 4 MB, round 69 to 72 to make it divisible by 4 MB.

Although these estimates are not precise, they suffice for planning a file system size. Create your file system large enough to be useful for some time before increasing its size.

TIP: Because increasing the size of a file system is usually easier than reducing its size, be conservative in estimating how large to create a file system.

An exception is the root file system. As a contiguous logical volume, the root file system is difficult to extend.

Resizing File System Logical Volumes

If your users have outgrown the space originally allocated for the file system, you can increase its size by first enlarging the logical volume it resides in using the `lvextend` command, then using the `extendfs` command to enlarge the file system contained in the logical volume.

Decreasing the size of a file system can be difficult. Based on the type of file system, you might not be able to decrease its size. However, you can create a *new* smaller file system to take its place.

For more information on resizing file system logical volumes, see "Administering File System Logical Volumes" (page 84).

File System Logical Volume Guidelines

Use the following guidelines when configuring file system logical volumes:

- If you create a file system that spans LVM disks, be sure that the logical volume in which the file system resides spans identical disk types for best system performance.
- By default, LVM creates logical volumes on available disks, not necessarily with regard for best performance. A file system can span two disks with different characteristics, in which case the file system performance can possibly be impaired.

  You can control which physical volumes contain the physical extents of a logical volume by following these steps:

  1. Create a logical volume without specifying a size using the `lvcreate` command or HP SMH. When you do not specify a size, by default, no physical extents are allocated for the logical volume.
  2. Extend the logical volume (that is, allocate space) to the specific physical volumes you want to contain the file system using the `lvextend` command.
- The root or boot logical volume is limited to either 2 GB or 4 GB, depending on your processor.

Setting Up Logical Volumes for Swap

NOTE: Version 2.x volume groups do not support swap logical volumes.

This section explains what to consider when using logical volumes as swap devices. For information on managing your system swap space, including determining how much and what type of swap space the system needs, see *HP-UX System Administrator's Guide: Configuration Management*.

When configured as swap, logical volumes are treated as **device swap space**. Device swap space occupies a logical volume or partition, which is typically reserved expressly for swapping purposes. This space can also be configured as a dump area (See "Dump Logical Volume Guidelines" (page 28)).

Swap Logical Volume Guidelines

Use the following guidelines when configuring swap logical volumes:

- Interleave device swap areas for better performance.

  Two swap areas on different disks perform better than one swap area with the equivalent amount of space. This configuration allows *interleaved* swapping, which means the swap areas are written to concurrently, thus enhancing performance.

  When using LVM, set up secondary swap areas within logical volumes that are on different disks using `lvextend`.

  If you have only one disk and must increase swap space, try to move the primary swap area to a larger region.

- Similar-sized device swap areas work best.

  Device swap areas must have similar sizes for best performance. Otherwise, when all space in the smaller device swap area is used, only the larger swap area is available, making interleaving impossible.

- By default, primary swap is located on the same disk as the root file system. By default, the `/stand/system` system kernel configuration file contains the configuration information for primary swap.
- If you are using logical volumes as secondary swap, allocate the secondary swap to reside on a disk other than the root disk for better performance.

Setting Up Logical Volumes for Dump

NOTE: Version 2.x volume groups do not support dump logical volumes.

This section explains what to consider when using logical volumes as dump devices. A dump area is disk space used to write an image of the core memory after a system crash. The analysis of a core dump can be useful in troubleshooting and restoring the system to working order.

By default, the primary swap device also serves as a dump area when no dump area is specifically designated. Although you are not required to retain primary swap as your dump area, doing so conserves disk space. You can configure a different or multiple dump devices on your system. To do this, create a logical volume as a dump device. This device can also be used for swap.

For information on adding, removing, or modifying dump devices, and configuring the dump algorithms, see the *HP-UX System Administrator's Guide: Configuration Management*.

Dump Logical Volume Guidelines

Use the following guidelines when configuring dump logical volumes:

- HP recommends using logical volumes for dump area rather than disk partitions.
- A dump logical volume can exist only within the root volume group; that is, the volume group that contains the root logical volume.
- You can use any secondary swap logical volume as a dump area too, provided the swap area is in the root volume group.

Planning for Availability

This section describes LVM features that can improve the availability and redundancy of your data. It addresses the following topics:

- "Increasing Data Availability Through Mirroring" (page 29)
- "Increasing Disk Redundancy Through Disk Sparing" (page 32)
- "Increasing Hardware Path Redundancy Through Multipathing" (page 33)

Increasing Data Availability Through Mirroring

NOTE: Mirroring requires an optional product, HP MirrorDisk/UX.

Mirroring means storing identical copies of data in logical volumes, preferably on separate disks. This redundancy has several advantages:

- If you mirror the root file system and swap, the operating system can tolerate a root disk failure because critical data is available on more than one LVM disk.
- If you mirror the logical volumes used by a particular application, the application continues to run even if a disk fails.
- If an I/O channel fails, LVM can recover the data from the duplicate source.
- Mirroring speeds read-intensive applications by enabling the hardware to read data from the most convenient LVM disk.
- You can back up one copy of the data while another copy continues to run.

Mirroring maps one logical extent to two or more sets of physical extents. The number of logical extents remains constant, but the number of used physical extents (and therefore, occupied disk space) changes depending on the number of mirrored copies. Mirroring increases data protection and system availability, but consumes twice as much disk space (or as many times more as there are mirror copies), so use disk mirroring for volatile, mission-critical data only.

Mirrored logical volumes must belong to the same volume group; you cannot mirror across volume groups.

This section contains the following information:

- "Mirror Write Behavior Control" (page 29)
- "Synchronizing a Mirrored Logical Volume" (page 31)

To learn more about basic mirroring tasks, see *Disk and File Management Tasks on HP-UX* published by Prentice Hall PTR, 1997.

Mirror Write Behavior Control

Three policies govern how mirrored logical extents are written to physical extents: the *allocation policy*, the **scheduling policy** for disk writes, and the **synchronization policy** for crash recovery. These policies can be set using HP SMH, the `lvcreate` command, or the `lvchange` command.

Allocation Policy

Mirrored extents can be allocated on physical volumes by **strict** or **nonstrict**, **contiguous** or **noncontiguous** policies. By default, the allocation policy of mirrored logical volumes is set to strict, noncontiguous.

Strict and Nonstrict Allocation

Strict allocation requires logical extents to be mirrored to physical extents on different physical volumes. Nonstrict allocation allows logical extents to be mirrored to physical extents that may be on the same physical volume. The `-s y` and `-s n` options to the `lvcreate` or `lvchange` commands set strict or nonstrict allocation.

CAUTION: Using nonstrict allocation can reduce the redundancy created by LVM mirroring because a logical extent can be mirrored to different physical extents on the same disk. Therefore, the failure of this one disk makes both copies of the data unavailable.

Contiguous and Noncontiguous Allocation

Contiguous allocation has three characteristics: the physical extents are allocated in ascending order, no gap exists between physical extents within a mirror copy, and all physical extents of a mirror copy reside on a single physical volume. Noncontiguous allocation allows logical extents to be mapped to nonconsecutive physical extents. The `-C y` and `-C n` options to the `lvcreate` or `lvchange` commands set contiguous or noncontiguous allocation.

NOTE: When the logical volumes allocated from the root volume group are mirrored, each must be set up with contiguous allocation.

Scheduling Policy

The LVM scheduler converts logical I/O requests into one or more physical I/O requests, then schedules them for processing at the hardware level. Scheduling occurs for both mirrored and nonmirrored data.

Two I/O scheduling policies are available: **parallel** and **sequential**.

Parallel Scheduling

The parallel scheduling policy is used by default with mirroring for maximum I/O performance. Parallel scheduling causes mirror operations to write simultaneously to all copies. LVM optimizes reads by reading from the physical volume with the fewest outstanding I/O operations. The `-d p` option to the `lvcreate` or `lvchange` command sets the scheduling policy to parallel for a logical volume.

Sequential Scheduling

The sequential scheduling policy causes mirror write operations to proceed sequentially; that is, LVM waits for one mirror write to complete before it begins the next mirror write. Likewise, LVM mirrors are read in a predefined order. On a practical level, sequential policy is used only for extreme caution in maintaining consistency of mirrors. The `-d s` option to the `lvcreate` or `lvchange` command sets the scheduling policy to sequential for a logical volume.

Synchronization Policy

You can maintain consistency of mirrored data by enabling or disabling two features of your logical volume: the Mirror Write Cache and Mirror Consistency Recovery.

Synchronization Using Mirror Write Cache

The **Mirror Write Cache (MWC)** provides a fast resynchronization of data following a system crash or failure, but at a potential performance cost for routine system use.

The MWC keeps track of where I/O writes are occurring on the volume group, and periodically records this activity in an ondisk data structure. An extra disk write is required for every mirrored write not already recorded on the physical volume. This slows down runtime I/O write processing and degrades performance when you access the disk at random; when writing to an area of the disk that is already recorded, the performance is not impaired. Upon system reboot after crash, the operating system uses the MWC to resynchronize inconsistent data blocks quickly.

The frequency of extra disk writes is small for sequentially accessed logical volumes (such as database logs), but increases when access is more random. Therefore, logical volumes containing database data or file systems with few or infrequently written large files (greater than 256K) must not use the MWC when runtime performance is more important than crash recovery time.

The `-M` option to the `lvcreate` or `lvchange` command controls the MWC.

Synchronization Using Mirror Consistency Recovery

When the **Mirror Consistency Recovery** is enabled, LVM does not impact runtime I/O performance. However, following a system crash, for any logical volumes using Mirror Consistency Recovery, the entire data space resynchronizes when you activate the volume group. Synchronization can be performed in the background without interfering with reboot or access; however, during this time I/O performance and redundancy are degraded.

Synchronization with No Mirror Consistency Mechanism

When Mirror Consistency Recovery is disabled, the operating system's runtime behavior is identical to that of the previous approach. However, following a crash, LVM does *not* perform any resynchronization of data. This approach is useful for swap volumes and for volumes used by an application (such as a database) with its own means of maintaining or recovering consistent data, such as transaction log files. However, database log files themselves can be configured as a mirrored logical volume to use the MWC.

The `-c` option to the `lvcreate` or `lvchange` command controls the use of the Mirror Consistency Recovery.

Synchronizing a Mirrored Logical Volume

The data in a mirrored copy or copies of a logical volume can become out of sync, or “stale.” For example, mirrored data becomes stale if LVM cannot access a disk as a result of disk power failure. Under such circumstances, for each mirrored copy to re-establish identical data, synchronization must occur. Usually, synchronization occurs automatically, but sometimes it must be done manually.

Automatic Synchronization

If you activate a volume group that is *not currently active*, either automatically at boot time or later with the `vgchange` command, then LVM automatically synchronizes the mirrored copies of all logical volumes with the Mirror Consistency Recovery policy enabled. It replaces data in physical extents marked as stale with data from nonstale extents. Otherwise, no automatic synchronization occurs and manual synchronization is necessary.

LVM also automatically synchronizes mirrored data in the following cases:

- When you increase the number of mirror copies of a logical volume using the `–m` option of `lvmerge`, the newly added physical extents are synchronized.
- When a disk comes back online after experiencing a power failure.

Manual Synchronization

If you look at the status of a logical volume using `lvdisplay -v`, you can verify if the logical volume contains any stale data. You can then identify which disk contains the stale physical extents. Manually synchronize the data in one or more logical volumes using either the `lvsync` command or all logical volumes in one or more volume groups using the `vgsync` command. For more information, see *lvdisplay*(1M), *vgsync*(1M), and *lvsync*(1M).

Parallel Synchronization

By default, the `lvsync` command synchronizes logical volumes serially. In other words, it acts on the logical volumes specified on the command line one at a time, waiting until a volume finishes synchronization before starting the next. Starting with the September 2007 release of HP-UX 11i Version 3, you can use the `–T` option to synchronize logical volumes in parallel. With the `–T` option, `lvsync` spawns multiple threads to simultaneously synchronize all logical volumes belonging to the same volume group, often reducing the total synchronization time.

TIP: The `vgchange`, `lvmerge`, and `lvextend` commands support the `–s` option to suppress the automatic synchronization of stale extents. If you are performing multiple mirror-related tasks, you can suppress the extent synchronization until you have finished all the tasks, then run `lvsync` with `–T` to synchronize all the mirrored volumes in parallel. For example, you can use `vgchange -s` with `lvsync -T` to reduce the activation time for volume groups with mirrored logical volumes. For another example, see "Mirroring the Boot Disk" (page 79).

Increasing Disk Redundancy Through Disk Sparing

NOTE: Version 2.x volume groups do not support disk sparing.

Disk sparing requires the optional product HP MirrorDisk/UX.

MirrorDisk/UX is not available for shared LVM environments within a high availability cluster across more than two nodes. You cannot configure sparing within these environments. In such cases, HP recommends that you use hardware mirroring through RAID devices, which can support their own form of sparing.

If a disk containing mirrored data fails, replace the disk as soon as possible, as described in "Replacing a Bad Disk" (page 102). Before you replace the disk, the data in your logical volume does not have an extra mirrored copy unless you have set up more than one mirror copy. Even with multi–way mirroring, your level of security is reduced because of the loss of one mirror copy.

To prevent this possibility, you can use one or more spare disks within each of your volume groups to serve as substitute devices in the event of disk failure. With this configuration, LVM automatically reconfigures the volume group so that the spare physical volume takes the place of a failed device without any intervention required. That is, a copy of the data from all the logical volumes currently on the failed disk is created on the substitute physical volume. This process is referred to as automatic sparing, or **sparing**. Sparing occurs while the logical volume remains available to users. You can then schedule the replacement of the failed disk at a time of minimal inconvenience to you and your users. At that time, you copy the data from the spare disk back to the original disk or its replacement and return the spare disk to its role as a standby empty disk.

For sparing to occur, the following conditions must be met:

- All logical volumes in the volume group must have been configured with strict mirroring so that mirrored copies are maintained on separate disks because LVM copies the data onto the spare from an undamaged disk rather than from the defective disk itself.
- At least one physical volume must be available as a standby spare; if your last spare is already in use as a result of a prior disk failure, it cannot serve as a currently available spare.
- The available spare must be at least as large as the failed disk.

The spare physical volume disk space is not available for extent allocation for any other purpose than in the event of serving as a substitute disk in the event of disk failure. Therefore, its physical extents are not included in the counts shown under `total PE` or `free PE` in the output of the `pvdisplay` and `vgdisplay` commands.

NOTE: If it is important to maintain comparable performance in the event of disk failure, configure a spare physical volume to each bus. However, if more than one disk on the same bus fails, even with this strategy, performance will be impacted.

The `pvdisplay` and `vgdisplay` commands provide information on whether a given physical volume is an empty standby spare or currently holding data as a spare in use, along with information on any physical volume that is currently unavailable but had data spared.

Increasing Hardware Path Redundancy Through Multipathing

Your hardware might provide the capability for dual cabling (dual controllers) to the same physical volume. If so, LVM can be configured with multiple paths to the same physical volume. If the primary link fails, an automatic switch to an alternate link occurs. Using multipathing increases availability.

NOTE:
As of HP-UX 11i Version 3, the mass storage stack supports native multipathing without using LVM pvlinks. Native multipathing provides more load balancing algorithms and path management options than LVM. HP recommends using native multipathing to manage multipathed devices instead of using LVM's alternate links.

For backward compatibility, you can use existing pvlinks. However, you must use legacy device special files for physical volumes and disable native multipathing for those legacy device special files using the `scsimgr` command. For more information, see the white paper *LVM Migration from Legacy to Agile Naming Model*, available at http://docs.hp.com.

Setting Up Multipathing to a Physical Volume

To use an alternate link, you can create a volume group with `vgcreate`, specifying both the primary link and the alternate link device file names. Both links must represent paths to the same physical volume. (Do not run `pvcreate` on the alternate link; it must already be the same physical volume as the primary link.) When you indicate two device file names, both referring to the same disk using `vgcreate`, LVM configures the first one as the primary link and the second one as the alternate link.

For example, if a disk has two cables and you want to make one the primary link and the other an alternate link, enter the following command:

```
# vgcreate /dev/vg01 /dev/dsk/c3t0d0 /dev/dsk/c5t0d0
```

To add an alternate link to a physical volume that is already part of a volume group, use `vgextend` to indicate the new link to the physical volume. For example, if `/dev/dsk/c2t0d0` is already part of your volume group but you want to add another connection to the physical volume, enter the following command:

```
# vgextend /dev/vg02 /dev/dsk/c4t0d0
```

If the primary link fails, LVM automatically switches from the primary controller to the alternate controller. However, you can also tell LVM to switch to a different controller at any time using the `pvchange` command. For example:

```
# pvchange -s /dev/dsk/c2t1d0
```

After the primary link has recovered, LVM automatically switches back from the alternate controller to the original controller unless you previously instructed it not to by using `pvchange` as follows:

```
# pvchange -S n /dev/dsk/c2t2d0
```

NOTE: You can also disable the automatic switchback by using the `-p` option to `pvchange` to disable proactive polling. For more information, see *pvchange*(1M).

View the current links to a physical volume using `vgdisplay` with the `-v` option.

Planning for Performance

This section describes strategies to obtain the best possible performance using LVM. It addresses the following topics:

- "General Performance Factors" (page 34)
- "Internal Performance Factors" (page 34)

- "Increasing Performance Through Disk Striping" (page 35)
- "Increasing Performance Through I/O Channel Separation" (page 37)

General Performance Factors

The following factors affect overall system performance, but not necessarily the performance of LVM.

Memory Usage

The amount of memory used by LVM is based on the values used at volume group creation time and on the number of open logical volumes. The largest portion of LVM memory is used for extent maps. The memory used is proportional to the maximum number of physical volumes multiplied by the maximum number of physical extents per physical volume for each volume group.

The other factors to be concerned with regarding memory parameters are expected system growth and number of logical volumes required. You can set the volume group maximum parameters to exactly what is required on the system today. However, if you want to extend the volume group by another disk (or perhaps replace one disk with a larger disk), you must use the `vgmodify` command.

CPU Usage

Compared to the non-LVM case, no significant impact to system CPU usage (by observing idle time) has been observed.

With LVM, extra CPU cycles are required to perform mirror write consistency cache operations, which is the only configurable option that impacts CPU usage.

Disk Space Usage

LVM reserves some disk space on each physical volume for its own metadata. The amount of space used is proportional to the maximum values used at volume group creation time.

Internal Performance Factors

The following factors directly affect the performance of I/O through LVM.

Scheduling Policy

The scheduling policy is significant only with mirroring. When mirroring, the sequential scheduling policy requires more time to perform writes proportional to the number of mirrors. For instance, a logical volume with three copies of data requires three times as long to perform a write using the sequential scheduling policy, as compared to the parallel policy. Read requests are always directed to only one device. Under the parallel scheduling policy, LVM directs each read request to the least busy device. Under the sequential scheduling policy, LVM directs all read requests to the device shown on the left hand side of an `lvdisplay -v` output.

Mirror Write Consistency Cache

The purpose of the Mirror Write Consistency cache (MWC) is to provide a list of mirrored areas that might be out of sync. When a volume group is activated, LVM copies all areas with an entry in the MWC from one of the good copies to all the other copies. This process ensures that the mirrors are consistent but does not guarantee the quality of the data.

On each write request to a mirrored logical volume that uses MWC, LVM potentially introduces one extra serial disk write to maintain the MWC. Whether this condition occurs depends on the degree to which accesses are random.

The more random the accesses, the higher the probability of missing the MWC. Getting an MWC entry can involve waiting for one to be available. If all the MWC entries are currently being used by I/O in progress, a given request might have to wait in a queue of requests until an entry becomes available.

Another performance consideration for mirrored logical volumes is the method of reconciling inconsistencies between mirror copies after a system crash. Two methods of resynchronization are available: Mirror Consistency Recovery (MCR) and none. Whether you use the MWC depends on which aspect of system performance is more important to your environment, run time or recovery time.

For example, a customer using mirroring on a database system might choose "none" for the database logical volume because the database logging mechanism already provides consistency recovery. The logical volume used for the log uses the MWC if quick recovery time was an issue, or MCR if higher runtime performance is required. A database log is typically used by one process and is sequentially accessed, which means it suffers little performance degradation using MWC because the cache is hit most of the time.

Disk Spanning

For disk areas that see the most intensive use by multiple processes, HP recommends spreading the data space for this disk area across as many physical volumes as possible.

Number of Volume Groups

The number of volume groups is directly related to the MWC issues. Because there is only one MWC per volume group, disk space that is used for many small random write requests must be kept in distinct volume groups if possible when the MWC is being used. This is the only performance consideration that affects the decision regarding the number of volume groups.

Physical Volume Groups

This factor can be used to enforce the separation of different mirror copies across I/O channels. You must define the physical volume groups. This factor increases the availability by decreasing the single points of failure and provides faster I/O throughput because of less contention at the hardware level.

For example, in a system with several disk devices on each card and several cards on each bus converter, create physical volume groups so that all disks off of one bus converter are in one group and all the disks on the other are in another group. This configuration ensures that all mirrors are created with devices accessed through different I/O paths.

Increasing Performance Through Disk Striping

Disk striping distributes logically contiguous data blocks (for example, chunks of the same file) across multiple disks, which speeds I/O throughput for large files when they are read and written sequentially (but not necessarily when access is random).

The disadvantage of disk striping is that the loss of a single disk can result in damage to many files because files are purposely spread across two or more disks.

Consider using disk striping on file systems where large files are stored, if those files are normally read and written sequentially and I/O performance is important.

When you use disk striping, you create a logical volume that spans multiple disks, allowing successive blocks of data to go to logical extents on different disks. For example, a three-way striped logical volume has data allocated on three disks, with each disk storing every third block of data. The size of each of these blocks is called the **stripe size** of the logical volume. The stripe size (in K) must be a power of two in the range 4 to 32768 for a Version 1.0 volume group, and a power of two in the range 4 to 262144 for a Version 2.x volume group.

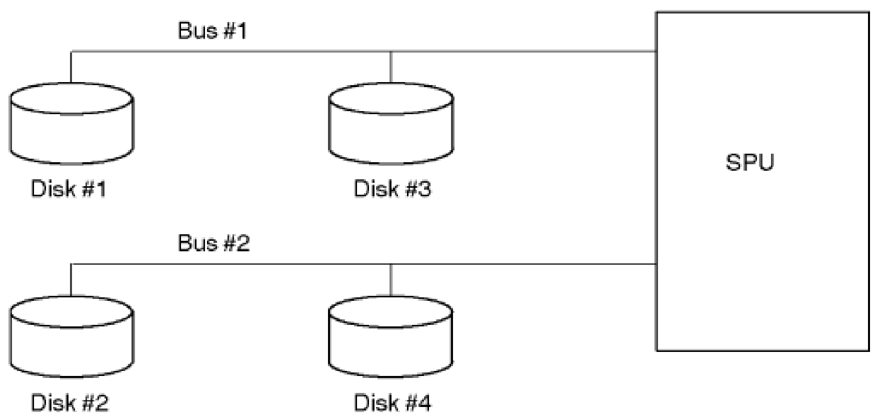
Disk striping can increase the performance of applications that read and write *large, sequentially accessed* files. Data access is performed over the multiple disks simultaneously, resulting in a decreased amount of required time as compared to the same operation on a single disk. If all of the striped disks have their own controllers, each can process data simultaneously.

You can use standard commands to manage your striped logical volumes. For example, the `lvcreate`, `diskinfo`, `newfs`, `fsck`, and `mount` commands all work with striped logical volumes.

The following guidelines, most of which apply to LVM disk usage, apply to striped logical volumes for performance reasons:

- Best performance results from a striped logical volume that spans similar disks. The more closely you match the disks by speed, capacity, and interface type, the better the performance you can expect. When striping across several disks of varying speeds, performance is no faster than that of the *slowest* disk.
- If you have more than one interface card or bus to which you can connect disks, distribute the disks as evenly as possible among them. That is, each interface card or bus must have roughly the same number of disks attached to it. You can achieve the best I/O performance when you use more than one bus and interleave the stripes of the logical volume. For example, if you have two buses with two disks on each bus, order the disks so that disk 1 is on bus 1, disk 2 is on bus 2, disk 3 is on bus 1, and disk 4 is on bus 2, as shown in Figure 2-2.

Figure 2-2 Interleaving Disks Among Buses



- Increasing the number of disks might not improve performance because the maximum efficiency that can be achieved by combining disks in a striped logical volume is limited by the maximum throughput of the file system itself and by the buses to which the disks are attached.
- Disk striping is highly beneficial for applications with few users and large, sequential transfers. However, applications that exhibit small, concurrent, random I/O (such as databases) often see no performance gain through disk striping. Consider four disks with a stripe size of 512 bytes. Each 2K request is sent to all disks. One 2K request completes in about the same time when the disk has been striped. However, several 2K requests are all serialized because all the disks must seek for each request. On the nonstriped system, performance might actually be better because each disk might service separate requests in parallel.

Determining Optimum Stripe Size

The logical volume stripe size identifies the size of each of the blocks of data that make up the stripe. You can set the stripe size to a power of two in the range 4 to 32768 for a Version 1.0 volume group, or a power of two in the range 4 to 262144 for a Version 2.x volume group. The default is 8192.

NOTE: The stripe size of a logical volume is not related to the physical sector size of a disk, which is typically 512 bytes.

How you intend to use the striped logical volume determines what stripe size you assign to it.

For best results follow these guidelines:

- If you plan to use the striped logical volume for an HFS file system, select the stripe size that most closely reflects the block size of the file system. The `newfs` command enables you to specify a block size when you build the file system and provides a default block size of 8K for HFS.
- If you plan to use the striped logical volume for a JFS (VxFS) file system, use the largest available size, 64K. For I/O purposes, VxFS combines blocks into extents, which are variable in size and may be very large. The configured block size, 1K by default, is not significant in this context.
- If you plan to use the striped logical volume as swap space, set the stripe size to 16K for best performance. For more information on configuring swap, see "Administering Swap Logical Volumes" (page 88).
- If you plan to use the striped logical volume as a raw data partition (for example, for a database application that uses the device directly), the stripe size must be as large or larger than the I/O size for the application.

You might need to experiment to determine the optimum stripe size for your particular situation. To change the stripe size, re-create the logical volume.

Interactions Between Mirroring and Striping

Mirroring a striped logical volume improves the read I/O performance in a same way that it does for a nonstriped logical volume. Simultaneous read I/O requests targeting a single logical extent are served by two or three different physical volumes instead of one. A striped and mirrored logical volume follows a strict allocation policy; that is, the data is always mirrored on different physical volumes.

Increasing Performance Through I/O Channel Separation

I/O channel separation is an approach to LVM configuration requiring that mirrored copies of data reside on LVM disks accessed using separate host bus adapters (HBAs) and cables. I/O channel separation achieves higher availability and better performance by reducing the number of single points of possible hardware failure. If you mirror data on two separate disks, but through one card, your system can fail if the card fails.

You can separate I/O channels on a system with multiple HBAs and a single bus, by mirroring disks across different HBAs. You can further ensure channel separation by establishing a policy called **PVG-strict** allocation, which requires logical extents to be mirrored in separate physical volume groups. **Physical volume groups** are subgroups of physical volumes within a volume group.

An ASCII file, `/etc/lvmpvg`, contains all the mapping information for the physical volume group, but the mapping is not recorded on disk. Physical volume groups have no fixed naming convention; you can name them PVG0, PVG1, and so on. The `/etc/lvmpvg` file is created and updated using the `vgcreate`, `vgextend`, and `vgreduce` commands, but you can edit the file with a text editor.

I/O channel separation is useful for databases, because it heightens availability (LVM has more flexibility in reading data on the most accessible logical extent), resulting in better performance. If you define your physical volume groups to span I/O devices, you ensure against data loss even if one HBA fails.

When using physical volume groups, consider using a PVG-strict allocation policy for logical volumes.

Planning for Recovery

Flexibility in configuration, one of the major benefits of LVM, can also be a source of problems in recovery. The following are guidelines to help create a configuration that minimizes recovery time:

- Keep the number of disks in the root volume group to a minimum; HP recommends using three disks, even if the root volume group is mirrored.

  Root volume groups with many disks make reinstallation difficult because of the complexity of recovering LVM configurations of accessory disks within the root volume group.

  A small root volume group is quickly recovered. In some cases, you can reinstall a minimal system, restore a backup, and be back online within three hours of diagnosis and replacement of hardware. Another benefit is that exact match to the previous root disk layout is not required.

  Three disks in the root volume group are better than two, because of quorum restrictions. With a two-disk root volume group, the loss of one disk can require you to override quorum to activate the volume group; if you must reboot to replace the disk, overriding quorum requires you to interrupt the boot process. If you have three disks in the volume group and they are isolated from each other such that a hardware failure only affects one of them, then failure of only one disk enables the system to maintain quorum.

  There are two reasons to expand the root volume group beyond a minimal size.

  — A very small root disk.

    In this case, HP recommends migrating or installing to a larger disk.

  — Providing for dump-to-swap for large memory systems.

    Swap volumes targeted for dump must be in the root volume group. A better solution is to configure an extra dedicated disk for dump up front.

- Do not use network-based backup programs such as Omniback or Networker for basic root volume group backups. The complexity associated with these utilities can significantly delay the resumption of processing.

  HP recommends using Ignite-UX and the `make_net_recovery` command to back up and recover your root volume group.

- Create adequate documentation.

  Output from `ioscan -kf`, `vgcfgrestore -lv` for all groups and `vgscan -pv`, and `lvlnboot -v` are very minimal requirements. Recovery from almost any problem is possible if these and output from `vgdisplay -v` for all groups and `lvdisplay -v` for all volumes are available. Extent mappings are critical to recovery of LVM volumes with corrupted headers. Additionally, output from `pvdisplay -v` for all physical volumes, although not as important, provides complete volume group information. Hard copy is not required or even necessarily practical, but accessibility during recovery is important and should be planned for.

- Many small groups are better than fewer large groups when configuring auxiliary volume groups.

  Reloading dozens of gigabytes of data because one disk is missing out of a group after a root disk rebuild can be difficult. Also the scope of a catastrophic single disk failure within a group is minimized with many small groups.

  Size implies complexity. The larger the number of disks in a single group, the more chances an administrator has to create an error that affects the entire group. It is simpler to identify and import smaller groups if necessary. It is also simpler to conceptualize and map smaller groups when required.

Preparing for LVM System Recovery

To ensure that the system data and configuration are recoverable in the event of a system failure, follow these steps:

1. Load any patches for LVM.
2. Use Ignite-UX to create a recovery image of your root volume group. Although Ignite-UX is not intended to be used to back up all system data, you can use it with other data recovery applications to create a method of total system recovery.
3. Perform regular backups of the other important data on your system.

   Without a valid backup, you risk losing some or all of your data.

4. Regularly print out your system configuration.

   The configuration details stored on the system might not be accessible during a recovery. A printed copy is an invaluable reference. HP recommends printing the configuration details once a week *and* every time a change is made. Some of the commands create large amounts of output. An alternative to printing them is to output the information to a file and then storing the file on tape which enables quick recovery of the information when needed. You can include this configuration file with the backup in step 3.

   The easiest way to save the configuration is to set up a `cron` job to run regularly, so that the system backs it up automatically.

   Use the following commands to get a useful output:

   `/usr/sbin/ioscan -fk`

   `/usr/sbin/vgdisplay -v`

   `/usr/sbin/lvlnboot -v`

   `/usr/sbin/lvdisplay -v /dev/vgXX/lvYY` (for every logical volume)

   `/usr/sbin/pvdisplay -v /dev/dsk/c#t#d0` (for every LVM disk)

   `lp /etc/fstab`

   As an alternative, you can write an intelligent script that detects any changes in the configuration and only print out those changes. An example script is included at the end of this section.

5. Back up the LVM configuration after every configuration change.

   The `vgcfgbackup` command copies the LVM headers from the system area of the disk to a disk file, which resides in the `/etc/lvmconf` directory. After this information is stored in a disk file, it can be backed up to tape during file system backups.

   The information in this file enables you to replace the LVM headers on the disk in the event of the disk being replaced or your LVM configuration becoming corrupted.

   It is important that these configuration backups are taken whenever you make a change to any part of the LVM configuration. By default all the commands perform a backup so a manual `vgcfgbackup` after each command is not required.

   Do this task on a regular basis, whether you have made changes or not. Enter the following command:

   ```
   # /usr/sbin/vgcfgbackup /dev/vgXX   (for every volume group)
   ```

6. Update the boot structures after every change to the root volume group.

   This task is only required if you are using LVM on your boot disk. Whenever you make changes to the root volume group, which is usually named `/dev/vg00`, the BDRA on the boot disk must be updated. This is normally performed automatically by the LVM commands. To update the BDRA manually, enter the following command:

```
# /usr/sbin/lvlnboot -R
```

Example Script for LVM Configuration Recording

The following example script captures the current LVM and I/O configurations. If they differ from the previously captured configuration, the script prints the updated configuration files and notifies the system administrator.

```
#!/usr/bin/ksh
WORKDIR=/lvmbackup # directory is regularly backed up
LOG=$WORKDIR/log
SYSADM=root
if [ -f "$LOG" ]
then
    rm -f "$LOG"
fi
if [ ! -d "$WORKDIR" ]
then
    echo "missing directory $WORKDIR" exit 1
fi
cd $WORKDIR
/usr/sbin/vgdisplay -v -F > vgdisplay.new
LVMVGS=`grep vg_name vgdisplay.new | cut -d: -f1 | cut -d= -f2`
LVMPVOLS=`grep pv_name vgdisplay.new | cut -d: -f1 | cut -d= -f2 | cut -d, -f1`
LVMLVOLS=`grep lv_name vgdisplay.new | cut -d: -f1 | cut -d= -f2`
/usr/sbin/pvdisplay -v $LVMPVOLS > pvdisplay.new
/usr/sbin/lvdisplay -v $LVMLVOLS > lvdisplay.new
/usr/sbin/lvlnboot -v > lvlnboot.new 2> /dev/null
/usr/sbin/ioscan -fk > ioscan.new
cp /etc/fstab fstab.new
for CURRENT in *new.
do
    ORIG=${CURRENT%.new}
    if diff $CURRENT $ORIG > /dev/null then
        # files are the same....do nothing
        rm $CURRENT
    else
        # files differ...make the new file the current file, move old
        # one to file.old.
        echo `date` "The config for $ORIG has changed." >> $LOG
        echo "Copy of the new $ORIG config has been printed" >> $LOG
        lp $CURRENT
        mv $ORIG ${ORIG}old.
        mv $CURRENT $ORIG
    fi
done
if [ -s "$LOG" ]
then
    mailx -s "LVM configs have changed" $SYSADM < $LOG
fi
exit 0
```

3 Administering LVM

This section contains information on the day-to-day operation of LVM. It addresses the following topics:

Administration Tools

HP-UX provides two tools to manage your LVM configuration:

- **HP System Management Homepage (HP SMH)**: An HP-UX tool that provides an easy-to-use GUI for performing most LVM tasks. HP SMH minimizes or eliminates the need for detailed knowledge of administration commands, thus saving time. Use HP SMH to manage your LVM configuration whenever possible, especially when performing a new task. HP SMH can help you with the following tasks:
  - Creating or removing volume groups
  - Adding or removing disks within volume groups
  - Activating and deactivating volume groups
  - Exporting and importing volume groups
  - Creating, removing, or modifying logical volumes
  - Increasing the size of logical volumes
  - Creating or increasing the size of a file system in a logical volume
  - Creating and modifying mirrored logical volumes
  - Creating striped logical volumes
  - Splitting a mirrored logical volume and merging a mirror copy
  - Adding and removing mirror copies within mirrored logical volumes
  - Creating and modifying physical volume groups

  To use HP SMH, enter the command `/usr/sbin/smh`.

  For help using HP SMH, see the HP SMH online help.

- **LVM command-line interface**: LVM has a number of low-level user commands to perform LVM tasks, described in "Physical Volume Management Commands" (page 41), "Volume Group Management Commands" (page 42), and "Logical Volume Management Commands" (page 42).

  The following tables provide an overview of which commands perform a given task. For more information, see the *HP-UX Reference*.

Table 3-1 Physical Volume Management Commands

Task	Command
Changing the characteristics of a physical volume in a volume group	`pvchange`
Creating a physical volume for use in a volume group	`pvcreate`
Displaying information about physical volumes in a volume group	`pvdisplay`

Table 3-1 Physical Volume Management Commands *(continued)*

Task	Command
Moving data from one physical volume to another	`pvmove`
Removing a physical volume from LVM control	`pvremove`
Checking or repairing a physical volume	`pvck`*
Checking if a disk volume is under LVM control	`lvmchk`

Table 3-2 Volume Group Management Commands

Task	Command
Creating a volume group	`vgcreate`
Removing a volume group	`vgremove`
Activating, deactivating, or changing the characteristics of a volume group	`vgchange`
Modifying the configuration parameters of a volume group; handling physical volume size changes	`vgmodify`*
Backing up volume group configuration information	`vgcfgbackup`
Restoring volume group configuration from a configuration file	`vgcfgrestore`
Displaying information about volume groups	`vgdisplay`
Exporting a volume group and its associated logical volumes	`vgexport`
Importing a volume group onto the system; adding an existing volume group back into the LVM configuration files	`vgimport`
Scanning all physical volumes for logical volumes and volume groups; recovering the LVM configuration files	`vgscan`
Adding a disk to a volume group	`vgextend`
Removing a disk from a volume group	`vgreduce`
Synchronizing mirrored logical volumes in a volume group	`vgsync`
Modifying the volume group ID on a physical volume	`vgchgid`
Migrating a volume group from legacy to persistent device files	`vgdsf`
Displaying limits associated with a volume group version	`lvmadm`

Table 3-3 Logical Volume Management Commands

Task	Command
Creating a logical volume	`lvcreate`
Modifying a logical volume	`lvchange`
Displaying information about logical volumes	`lvdisplay`
Increasing the size of a logical volume by allocating disk space	`lvextend`
Decreasing the size of a logical volume	`lvreduce`
Removing the allocation of disk space for one or more logical volumes within a volume group	`lvremove`
Preparing a logical volume to be a root, primary swap, or dump volume; updating the boot information on the boot physical volume	`lvlnboot`*
Removing the link that makes a logical volume a root, primary swap, or dump volume	`lvrmboot`*

Table 3-3 Logical Volume Management Commands *(continued)*

Task	Command
Splitting a mirrored logical volume into two logical volumes	`lvsplit`
Merging two logical volumes into one mirrored logical volume	`lvmerge`
Synchronizing mirror copies in a mirrored logical volume	`lvsync`

*The `pvck`, `vgmodify`, `lvlnboot`, and `lvrmboot` commands are not supported on Version 2.x volume groups.

The command-line interface is more powerful—and thus more dangerous—than HP SMH and offers options that are not available using HP SMH. For example, the following tasks cannot currently be done by HP SMH. For these tasks, use the HP-UX commands:

— "Creating a Spare Disk" (page 68)
— "Reinstating a Spare Disk" (page 68)
— "Synchronizing a Mirrored Logical Volume" (page 31)
— "Moving Data to a Different Physical Volume" (page 66)
— "Replacing a Bad Disk" (page 102)

The remainder of this chapter explains how to do LVM tasks using HP-UX commands. However, HP SMH is the tool of choice for most administration work.

Displaying LVM Information

To display information about volume groups, logical volumes, or physical volumes, use one of three commands. Each command supports the `-v` option to display detailed output and the `-F` option to help with scripting.

Information on Volume Groups

Use the `vgdisplay` command to show information about volume groups. For example:

```
# vgdisplay -v vg01
-- Volume groups --
VG Name                     /dev/vg01
VG Write Access             read/write
VG Status                   available
Max LV                      255
Cur LV                      1
Open LV                     1
Max PV                      16
Cur PV                      1
Act PV                      1
Max PE per PV               1016
VGDA                        2
PE Size (Mbytes)            4
Total PE                    508
Alloc PE                    508
Free PE                     0
Total PVG                   0
Total Spare PVs             0
Total Spare PVs in use      0
VG Version                  1.0
VG Max Size                 1082g
VG Max Extents              69248

-- Logical volumes --
LV Name                     /dev/vg01/lvol1
LV Status                   available/syncd
LV Size (Mbytes)            2032
```

```
Current LE                    125
Allocated PE                  508
Used PV                       1

-- Physical volumes --
PV Name                       /dev/disk/disk42
PV Status                     available
Total PE                      508
Free PE                       0
Autoswitch                    On
Proactive Polling             On
```

Use the `vgdisplay` command to verify whether the LVM configuration in memory has problems. If the LVM configuration is working properly, there are no error messages, and the display shows the following:

- The status is `available` (or `available/exclusive` for Serviceguard volume groups).
- All the physical volumes are active; that is, the current number of physical volumes (`Cur PV`) is equal to number of active physical volumes (`Act PV`).
- All the logical volumes are open; that is, `Cur LV` is equal to `Open LV`.

Information on Physical Volumes

Use the `pvdisplay` command to show information about physical volumes. For example:

```
# pvdisplay -v /dev/disk/disk47
-- Physical volumes --
PV Name                       /dev/disk/disk47
VG Name                       /dev/vg00
PV Status                     available
Allocatable                   yes
VGDA                          2
Cur LV                        9
PE Size (Mbytes)              4
Total PE                      1023
Free PE                       494
Allocated PE                  529
Stale PE                      0
IO Timeout (Seconds)          default
Autoswitch                    On
Proactive Polling             On

-- Distribution of physical volume --
LV Name            LE of LV  PE for LV
/dev/vg00/lvol1    25         25
/dev/vg00/lvol2    25         25
/dev/vg00/lvol3    50         50

--- Physical extents ---
PE   Status   LV                LE
0000 current  /dev/vg00/lvol1    0000
0001 current  /dev/vg00/lvol1    0001
0002 current  /dev/vg00/lvol1    0002
...
1021 free                        0000
1022 free                        0000
```

If the physical volume is functioning properly, `Stale PE` is 0.

Information on Logical Volumes

Use the `lvdisplay` command to show information about logical volumes. For example:

```
# lvdisplay -v /dev/vg00/lvol1
-- Logical volumes --
LV Name                       /dev/vg00/lvol1
```

```
VG Name                     /dev/vg00
LV Permission               read/write
LV Status                   available/syncd
Mirror copies               0
Consistency Recovery        MWC
Schedule                    parallel
LV Size (Mbytes)            100
Current LE                  25
Allocated PE                25
Stripes                     0
Stripe Size (Kbytes)        0
Bad block                   off
Allocation                  strict/contiguous
IO Timeout (Seconds)        default

-- Distribution of logical volume --
PV Name            LE on PV  PE on PV
/dev/disk/disk42   25        25

-- Logical extents --
LE   PV1                PE1  Status 1
0000 /dev/disk/disk42   0000 current
0001 /dev/disk/disk42   0001 current
0002 /dev/disk/disk42   0002 current
```

Common LVM Tasks

The section addresses the following topics:

- "Initializing a Disk for LVM Use" (page 46)
- "Creating a Volume Group" (page 46)
- "Adding a Disk to a Volume Group" (page 48)
- "Removing a Disk from a Volume Group" (page 48)
- "Creating a Logical Volume" (page 49)
- "Extending a Logical Volume" (page 50)
- "Reducing a Logical Volume" (page 52)
- "Adding a Mirror to a Logical Volume" (page 52)
- "Removing a Mirror from a Logical Volume" (page 53)
- "Renaming a Logical Volume" (page 53)
- "Removing a Logical Volume" (page 53)
- "Exporting a Volume Group"
- "Importing a Volume Group" (page 54)
- "Modifying Volume Group Parameters" (page 55)
- "Quiescing and Resuming a Volume Group" (page 59)
- "Renaming a Volume Group" (page 60)
- "Splitting a Volume Group" (page 60)
- "Removing a Volume Group" (page 61)
- "Backing Up a Mirrored Logical Volume" (page 62)
- "Backing Up and Restoring Volume Group Configuration" (page 63)

Initializing a Disk for LVM Use

CAUTION: Initializing a disk using `pvcreate` results in the loss of any existing data currently on the disk.

NOTE: If your disk is already connected to the system, skip the first four steps of this procedure.

To initialize a disk for use as a physical volume, follow these steps:

1. Shut down and power off the system.
2. Connect the disk to the system and power supply. For detailed information and instructions on adding a particular type of disk, see your device documentation.
3. Power on the disk.
4. Boot the system.
5. Determine the disk's associated device file. To show the disks attached to the system and their device file names, enter the `ioscan` command with the `-f`, `-N`, and `-n` options. For example:

   ```
   # ioscan -f -n -N -C disk
   ```

   For more information, see *ioscan*(1M).
6. Initialize the disk as a physical volume using the `pvcreate` command. For example:

   ```
   # pvcreate /dev/rdisk/disk3
   ```

   Use the character device file for the disk.

   If you are initializing a disk for use as a boot device, add the `-B` option to `pvcreate` to reserve an area on the disk for a LIF volume and boot utilities. If you are creating a boot disk on an HP Integrity server, make sure the device file specifies the HP-UX partition number (2). For example:

   ```
   # pvcreate -B /dev/rdisk/disk3_p2
   ```

   NOTE: Version 2.x volume groups do not support bootable physical volumes. Do not use the `-B` option if the disk will be used in a Version 2.x volume group.

After a disk is initialized, it is called a physical volume.

Creating a Volume Group

To create a volume group, use the `vgcreate` command. The options differ depending on whether you are creating a Version 1.0 volume group or a Version 2.x volume group.

Creating the Volume Group Device File

As of the March 2008 release of HP-UX 11i Version 3, the `vgcreate` command automatically creates the device file `/dev/`*`vgname`*`/group` to manage the volume group, regardless of the volume group version. If you are using an HP-UX release before March 2008, or if you want to specify the minor number of the group file, you must create `/dev/`*`vgname`*`/group` before running the `vgcreate` command.

If the group file does not exist and `vgcreate` can not create it, `vgcreate` displays the following message:

```
vgcreate: "/dev/vgname/group": not a character device.
```

To create the volume group device file, follow these steps:

1. Create a directory for the volume group. For example:

   `# `**`mkdir /dev/`*`vgname`***

   By convention, *vgname* is vg*nn*, where *nn* is a unique number across all volume groups. However, you can choose any unique name up to 255 characters.

2. Create a device file named `group` in the volume group directory with the `mknod` command. For example:

   `# `**`mknod /dev/`*`vgname`*`/group c `*`major`*` 0x`*`minor`***

   The `c` following the device file name specifies that `group` is a character device file.

   major is the major number for the `group` device file. For a Version 1.0 volume group, it is 64. For a Version 2.x volume group, it is 128.

   minor is the minor number for the `group` file in hexadecimal. For a Version 1.0 volume group, *minor* has the form 0x*nn*0000, where *nn* is a unique number across all Version 1.0 volume groups. For a Version 2.x volume group, *minor* has the form 0x*nnn*000, where *nnn* is a unique number across all Version 2.x volume groups.

   For more information on `mknod`, see *mknod*(1M); for more information on major numbers and minor numbers, see "Device Number Format" (page 21).

Creating a Version 1.0 Volume Group

To create a Version 1.0 volume group, use the `vgcreate` command, specifying each physical volume to be included. For example:

`# `**`vgcreate /dev/`*`vgname`*` /dev/disk/disk3`**

Use the block device file to include each disk in your volume group. You can assign all the physical volumes to the volume group with one command, or create the volume group with a single physical volume. No physical volume can already be part of an existing volume group.

You can set volume group attributes using the following options:

`-V 1.0`	Version 1.0 volume group (default)
`-s` *pe_size*	Size of a physical extent in MB (default 4)
`-e` *max_pe*	Maximum number of physical extents per physical volume (default 1016)
`-l` *max_lv*	Maximum number of logical volumes (default 255)
`-p` *max_pv*	Maximum number of physical volumes (default 255)

The size of a physical volume is limited by *pe_size* times *max_pe*. If you plan to assign a disk larger than approximately 4 GB (1016 * 4 MB) to this volume group, use a larger value of *pe_size* or *max_pe*.

The size of the LVM metadata on each disk depends on *max_lv*, *max_pv*, and *max_pe*. If the `vgcreate` options would cause the metadata to exceed its available space, `vgcreate` does not create the volume group. You must select new values of *max_lv*, *max_pv*, and *max_pe*. For example, if you plan to use disks larger than 100 GB, consider reducing *max_pv*. For recommendations on choosing an optimal extent size, see Appendix C (page 131).

Creating a Version 2.x Volume Group

For Version 2.x volume groups, the `vgcreate` command does not require maximum values for the number of physical volumes (`-p`), number of logical volumes (`-l`), or extents per physical volume (`-e`). Instead you must specify only the extent size (`-s`) and the maximum size to which the volume group can grow (`-S`). For example:

`# `**`vgcreate -V 2.0 -s `*`pe_size`*` -S `*`vg_size`*` /dev/`*`vgname`*` /dev/disk/disk3`**

Use the block device file to include each disk in your volume group. You can assign all the physical volumes to the volume group with one command, or create the volume group with a single physical volume. No physical volume can already be part of an existing volume group.

You must use the following options:

`-V 2.0` or `-V 2.1`	Version 2.0 or Version 2.1 volume group
`-s` *`pe_size`*	Size of a physical extent in MB
`-S` *`vg_size`*	Maximum future size of the volume group The size of a volume group is the sum of the user data space on all physical volumes assigned to the volume group. *`vg_size`* is not the size of the volume group at creation; it is the size to which the volume group can grow in the future. This value can be specified in megabytes, gigabytes, terabytes, or petabytes, by adding the character `m`, `g`, `t`, or `p`, respectively. For example, to specify a maximum size of two terabytes, use `-S 2t`.

In a Version 2.x volume group, the number of physical extents in a volume group has an architectural limit, so your choice of physical extent size affects the maximum size of the volume group. To display the maximum volume group size for a given physical extent size, use the `-E` option to `vgcreate` with the `-s` option. For example:

```
# vgcreate -V 2.0 -E -s 256
Max_VG_size=2p:extent_size=256m
```

Conversely, to display the minimum physical extent size for a given volume group size, use the `-E` option to `vgcreate` with `-S`. For example:

```
# vgcreate -V 2.0 -E -S 2t
Max_VG_size=2t:extent_size=1m
```

Adding a Disk to a Volume Group

Often, as new disks are added to a system, they must be added to an existing volume group rather than creating a whole new volume group. If new disks are being added for user data, such as file systems or databases, do not to add them to the root volume group. Instead, leave the root volume group as only the disks containing the root file system and system file systems such as `/usr`, `/tmp`, and so on.

To add a disk to a volume group, follow these steps:

1. Initialize the disk as a physical volume by using the `pvcreate` command, as described in "Initializing a Disk for LVM Use" (page 46).
2. Add the physical volume to the volume group using the `vgextend` command and the block device file for the disk. For example:

   ```
   # vgextend /dev/vgname /dev/disk/disk3
   ```

Removing a Disk from a Volume Group

To remove a disk from a volume group, follow these steps:

1. Make sure the disk has no assigned physical extents, using the `pvdisplay` command. For example:

   ```
   # pvdisplay /dev/disk/disk3
   -- Physical volumes --
   PV Name                     /dev/disk/disk3
   VG Name                     /dev/vg00
   PV Status                   available
   Allocatable                 yes
   VGDA                        2
   Cur LV                      9
   PE Size (Mbytes)            4
   Total PE                    1023
   Free PE                     494
   Allocated PE                529
   Stale PE                    0
   ```

```
IO Timeout (Seconds)        default
Autoswitch                  On
Proactive Polling           On

-- Distribution of physical volume --
LV Name            LE of LV  PE for LV
/dev/vg00/lvol1    25        25
/dev/vg00/lvol2    25        25
/dev/vg00/lvol3    50        50

--- Physical extents ---
PE   Status   LV                 LE
0000 current  /dev/vg00/lvol1    0000
0001 current  /dev/vg00/lvol1    0001
0002 current  /dev/vg00/lvol1    0002
1021 free                        0000
1022 free                        0000
```

Check that the number of free physical extents (`Free PE`) matches the total number of physical extents (`Total PE`). If they are not the same, do one of the following tasks:

- Move the physical extents onto another physical volume in the volume group. See "Moving Data to a Different Physical Volume" (page 66).
- Remove the logical volumes from the disk, as described in "Removing a Logical Volume" (page 53). The logical volumes with physical extents on the disk are shown at the end of the `pvdisplay` listing.

2. After the disk no longer holds any physical extents, use the `vgreduce` command to remove it from the volume group. For example:

   `# `**`vgreduce /dev/vg`*`nn`*` /dev/disk/disk3`**

IMPORTANT: If you are using LVM pvlinks, as described in "Increasing Hardware Path Redundancy Through Multipathing" (page 33), you must run the `vgreduce` command for each link to the disk.

Creating a Logical Volume

To create a logical volume, follow these steps:

1. Decide how much disk space the logical volume needs.

   For example, you can add 200 MB of device swap space, or you might have a new project that you expect to grow to 10 GB.

2. Find a volume group that has adequate free space.

   To determine if there is sufficient disk space available for the logical volume within a volume group, use the `vgdisplay` command to calculate this information. `vgdisplay` outputs data on one or more volume groups, including the physical extent size (under `PE Size (MBytes)`) and the number of available physical extents (under `Free PE`). By multiplying these two figures, you get the number of megabytes available within the volume group. For more information, see *vgdisplay*(1M).

3. Create the logical volume using `lvcreate`. For example:

   `# `**`lvcreate -L`** ***`size_in_MB`*** **`/dev/vg`*`nn`***

   This command creates the logical volume `/dev/vg`*`nn`*`/lvol`*`n`* with LVM automatically assigning the *n* in `lvol`*`n`*.

When LVM creates the logical volume, it creates block and character device files for that logical volume and places them in the `/dev/vg`*`nn`* directory.

Creating a Striped Logical Volume

To create a striped logical volume, use `lvcreate` with the `-i` and the `-I` options to specify the number of disks and stripe width, respectively. For example, suppose you want to stripe across three disks with a stripe size of 32K. Your logical volume size is 240 MB. To create the striped logical volume, enter the following command:

```
# lvcreate -i 3 -I 32 -l 240 -n lvol1 /dev/vg01
```

The `lvcreate` command automatically rounds up the size of the logical volume to a multiple of the number of disks times the extent size. For example, if you have three disks you want to stripe across and the extent size is 4 MB, even though you indicate a logical volume size of 200 MB (`-L 200`), `lvcreate` creates a 204 MB logical volume because 200 is not a multiple of 12.

NOTE: When you stripe across multiple disks, the striped volume size cannot exceed the capacity of the smallest disk multiplied by the number of disks used in the striping.

Creating a Mirrored Logical Volume

To create a mirrored logical volume, use `lvcreate` with the `-m` option to select the number of mirror copies. To control how the mirror copies are managed, choose from the following options:

	Strict, Nonstrict, or PVG-strict Extent Allocation
`-s y`	Strict allocation (default)
`-s n`	Nonstrict allocation
`-s g`	PVG-strict allocation
	Contiguous or Noncontiguous Extent Allocation
`-C y`	Contiguous allocation
`-C n`	Noncontiguous allocation (default)
	Mirror Scheduling Policy
`-d p`	Parallel scheduling (default)
`-d s`	Sequential scheduling
	Mirror Consistency Policy
`-M y`	MWC enable (default, optimal mirror resynchronization during crash recovery)
`-M n -c y`	MCR enable (full mirror resynchronization during crash recovery)
`-M n -c n`	MCR disable (no mirror resynchronization during crash recovery)

For example, to create a 240 MB mirrored logical volume with one mirror copy, nonstrict allocation, parallel scheduling, and no mirror resynchronization, enter the following command:

```
# lvcreate -m 1 -s n -d p -M n -c n -L 240 -n lvol1 /dev/vg01
```

TIP: To change the characteristics of an existing mirrored logical volume, use the `lvchange` command. It supports the `-C`, `-c`, `-d`, `-M`, and `-s` options. For more information, see *lvchange*(1M).

Extending a Logical Volume

NOTE: Adding space to a logical volume does not automatically assign that space to the entity using that logical volume. For example, if you want to add space to a file system contained in a logical volume, you must run `extendfs` after extending the logical volume. See “Administering File System Logical Volumes” (page 84) and “Administering Swap Logical Volumes” (page 88) for more information.

---

1. Decide how much more disk space the logical volume needs.

   For example, you can add 200 MB of swap space, or an existing project might need an additional 1 GB.

2. Find out if any space is available using the `vgdisplay` command. For example:

```
# vgdisplay vg00
--- Volume groups ---
VG Name                      /dev/vg00
VG Write Access              read/write
VG Status                    available
Max LV                       255
Cur LV                       8
Open LV                      8
Max PV                       16
Cur PV                       1
Act PV                       1
Max PE per PV                2000
VGDA                         2
PE Size (Mbytes)             4
Total PE                     249
Alloc PE                     170
Free PE                      79
Total PVG                    0
Total Spare PVs              0
Total Spare PVs in use       0
VG Version                   1.0
VG Max Size                  1082g
VG Max Extents               69248
```

   The `Free PE` entry indicates the number of 4 MB extents available, in this case, 79 (316 MB).

3. Extend the logical volume. For example:

```
# lvextend -L 332 /dev/vg00/lvol7
```

   This increases the size of this volume to 332 MB.

Extending a Logical Volume to a Specific Disk

For performance reasons, you can force a logical volume to span disks. For example, if you want to create a 30 GB logical volume and put 10 GB on the first disk, another 10 GB on the second disk, and 10 GB on the third disk, then assuming that the extent size is 4 MB, the logical volume requires a total of 7680 extents. To extend the logical volume, follow these steps:

1. After making the disks physical volumes and creating your volume group, create a logical volume named `lvol1` of size 0. For example:

```
# lvcreate -n lvol1 /dev/vg01
```

2. Allocate a third of the extents to the logical volume on the first physical volume. For example:

```
# lvextend -l 2560 /dev/vg01/lvol1 /dev/disk/disk7
```

3. Increase the total number of physical extents allocated to the logical volume for the remaining physical volumes by 2560. In each case, the additional 2560 extents are allocated to the disk specified. For example:

```
# lvextend -l 5120 /dev/vg01/lvol1 /dev/disk/disk8
# lvextend -l 7680 /dev/vg01/lvol1 /dev/disk/disk9
```

   When you use the `-l` option with `lvextend`, you specify space in logical extents.

For another example, you have two disks in a volume group, both identical models. You currently have a 24 GB logical volume that resides on only one of the disks. You want to extend the logical volume size to 40 GB and ensure that the 16 GB increase is allocated to the other disk.

Extend the logical volume to a specific disk as follows:

```
# lvextend -L 40960 /dev/vg01/lvol2 /dev/disk/disk3
```

Here, when you use the `-L` option (uppercase), you specify space in megabytes, not logical extents.

For complete information on command options, see *lvextend*(1M).

Reducing a Logical Volume

CAUTION: Before you reduce a logical volume, you must notify the users of that logical volume. For example, before reducing a logical volume that contains a file system, *back up the file system.* Even if the file system currently occupies less space than the new (reduced) size of the logical volume, you will almost certainly lose data when you reduce the logical volume. See "Administering File System Logical Volumes" (page 84) and "Administering Swap Logical Volumes" (page 88) for the appropriate procedures for file systems and swap devices.

To reduce a logical volume, follow these steps:

1. To find out what applications are using the logical volume, use the `fuser` command. For example:

   ```
   # fuser -cu /dev/vg01/lvol5
   ```

   If the logical volume is in use, ensure the underlying applications can handle the size reduction. You might have to stop the applications.
2. Decide on the new size of the logical volume.

   For example, if the logical volume is mounted to a file system, the new size must be greater than the space the data in the file system currently occupies. The `bdf` command shows the size of all mounted volumes. The first column shows the space allocated to the volume; the second shows how much is actually being used. The new size of the logical volume must be larger than the size shown in the second column of the `bdf` output.
3. Reduce the size of the logical volume as follows:

   ```
   # lvreduce -L 500 /dev/vg01/lvol5
   ```

   This command reduces the logical volume `/dev/vg01/lvol5` to 500 MB.

Adding a Mirror to a Logical Volume

NOTE: Mirroring requires the optional product HP MirrorDisk/UX.

TIP: This task is easier to perform with HP SMH. HP SMH confirms that enough disk space is available for the mirror copy and that the available space meets any allocation policy.

To add a mirror to a logical volume, follow these steps:

1. Decide how many mirror copies to create.

   For this example, you will create one mirror; that is, you will keep two copies of the data online, the original and one mirror copy.
2. Make sure that there is enough free space in the volume group that contains the logical volume you want to mirror.

   The volume group needs at least as much free space as the logical volume you want to mirror currently has allocated to it—that is, you will double the amount of physical space this volume requires.

   To use strict mirroring (which HP recommends because it keeps the mirror copy on a separate disk) this free space must be on a disk or disks not currently used by the volume you want to mirror.

3. Use the `lvextend` command with the `-m` option to add the number of additional copies you want. For example:

   ```
   # lvextend -m 1 /dev/vg00/lvol1
   ```

   This adds a single mirror copy of the given logical volume.

   To force the mirror copy onto a specific physical volume, add it at end of the command line. For example:

   ```
   # lvextend -m 1 /dev/vg00/lvol1 /dev/disk/disk4
   ```

Removing a Mirror from a Logical Volume

To remove a mirror copy, use the `lvreduce` command, specifying the number of mirror copies you want to leave. For example, to remove all mirrors of a logical volume, enter the following command:

```
# lvreduce -m 0 /dev/vg00/lvol1
```

This reduces the number of mirror copies to 0, so only the original copy is left.

To remove the mirror copy from a specific disk, use `lvreduce` and specify the disk from which to remove the mirror copy. For example:

```
# lvreduce -m 0 /dev/vg00/lvol1 /dev/disk/disk4
```

Renaming a Logical Volume

To change the name of a logical volume, follow these steps:

1. Make sure that the logical volume has two existing device files, a block device file and a character or raw device file. They must have the same name, except that the character device file name has a leading `r`. For example, to rename a logical volume in volume group `vg00` from `lvol1` to `database`, list the contents of the `/dev/vg00` directory. For example:

   ```
   # cd /dev/vg00
   # ls -l
   total 0
   crw-r-----   1 root       sys  64 0x000000 Nov 16 02:49 group
   brw-r-----   1 root       sys  64 0x000001 Nov 16 02:49 lvol1
   brw-r-----   1 root       sys  64 0x000002 Nov 16 02:49 lvol2
   brw-r-----   1 root       sys  64 0x000003 Nov 16 02:49 lvol3
   brw-r-----   1 root       sys  64 0x000004 Nov 16 02:49 lvol4
   crw-r-----   1 root       sys  64 0x000001 Nov 16 02:49 rlvol1
   crw-r-----   1 root       sys  64 0x000002 Nov 16 02:49 rlvol2
   crw-r-----   1 root       sys  64 0x000003 Nov 16 02:49 rlvol3
   crw-r-----   1 root       sys  64 0x000004 Nov 16 02:49 rlvol4
   ```

2. Use the `mv` command to rename both files. For example:

   ```
   # mv /dev/vg00/lvol1 /dev/vg00/database
   # mv /dev/vg00/rlvol1 /dev/vg00/rdatabase
   ```

3. Update all references to the old name in any other files on the system. These include `/etc/fstab` for mounted file systems or swap devices and existing mapfiles from a `vgexport` command.

Removing a Logical Volume

CAUTION: Removing a logical volume makes its contents unavailable and likely to be overwritten. In particular, any file system contained in the logical volume is destroyed.

To remove a logical volume, follow these steps:

1. Make sure that the logical volume is not in use either as a file system or as raw disk space for an application. Use the `fuser` command as follows:

   ```
   # fuser -cu /dev/vg01/lvol5
   ```

   If the logical volume is in use, confirm that the underlying applications no longer need it. You might need to stop the applications.
2. Use the `lvremove` command to remove the logical volume. For example:

   ```
   # lvremove /dev/vg01/lvol5
   ```

   You can now use this space to extend an existing logical volume or build a new logical volume.

Exporting a Volume Group

Exporting a volume group removes all data concerning the volume group from the system, while leaving the data on the disks intact. The disks of an exported volume can be physically moved or connected to another system, and the volume group can be imported there.

Exporting a volume group removes information about the volume group and its associated physical volumes from `/etc/lvmtab` and `/etc/lvmtab_p`, and removes the volume group's directory with device files in the `/dev` directory.

1. Make sure that none of the logical volumes in the volume group are in use. You might need to stop applications using any logical volumes in the volume group, and unmount file systems contained in the volume group.

   Use the `fuser` command on each logical volume. For example:

   ```
   # fuser -cu /dev/vgnn/lvoln
   ```
2. Deactivate the volume group. For example:

   ```
   # vgchange -a n vgnn
   ```
3. Use the `vgexport` command to export the volume group. For example:

   ```
   # vgexport -v -m /tmp/vgnn.map vgnn
   ```

   If you are planning to move the volume group to another system, use the `-m` option to `vgexport` to create a `mapfile`. This ASCII file contains the logical volume names because they are not stored on the disks. You must create a mapfile if you do not use the default names `/dev/vgnn/lvoln` for the logical volumes in the volume group.

   If there are several disks in the volume group, use the `-s` option with `vgexport`; this option adds the volume group identifier (VGID) to the mapfile. When the volume group is imported, you can avoid specifying all the disks by name. See “Importing a Volume Group” (page 54).

When `vgexport` completes, all information about the volume group has been removed from the system. The disks can now be moved to a different system, and the volume group can be imported there.

Importing a Volume Group

To import a volume group, follow these steps:

1. Connect the disks to the system.
2. If you are using an HP-UX release before March 2008, or if you want to specify the minor number of the volume group device file, create it using the procedure in “Creating the Volume Group Device File” (page 46).
3. Use the `vgimport` command to import the volume group:

   ```
   # vgimport -v -N -m /tmp/vgnn.map /dev/vgnn list_of_disks
   ```

   If there are several disks in the volume group and the VGID was saved in the mapfile (that is, the `vgexport` command was run with the `-s` and `-m` options), you can avoid specifying all of them in the `vgimport` command line by using the `-s` option. This causes `vgimport` to scan all the disks on the system. Any physical volumes with a VGID matching the one in the mapfile are included automatically into the volume group.

4. Activate the volume group as follows:

   `# vgchange -a y vgnn`

NOTE: If the volume group contains any multipathed disks, HP recommends using HP-UX's native multipathing that is a superset of LVM's alternate links. See "Increasing Hardware Path Redundancy Through Multipathing" (page 33) for more information.

If you want to use LVM's alternate link features, importing the volume group has several implications:

- You must omit the `-N` option to the `vgimport` command.
- The `vgimport` sets the first link found as the primary link for all physical volumes. If the links are not in the desired order after the import, use `vgreduce` and `vgextend` on the primary link for each physical volume for which you want to change the primary.
- The tunable `maxfiles` must be more than double the number of disks free.

Modifying Volume Group Parameters

NOTE: The `vgmodify` command does not support Version 2.x volume groups.

When you create a volume group, you set certain characteristics of the volume group, such as the maximum number of physical extents per physical volume, the maximum number of physical volumes, and the maximum number of logical volumes. Using the `vgmodify` command, you can adjust these parameters without removing and re-creating the volume group or having to move your data.

Use the following procedure to adjust these volume group parameters:

1. Run `vgmodify` to collect information about the volume group.

   Save the output from these three commands:

   ```
   # vgmodify -o -r vgnn
   # vgmodify -v -t vgnn
   # vgmodify -v -n -t vgnn
   ```

   The `-o` option attempts to optimize the values by making full use of the existing LVM metadata space. The `-t` option reports the optimized range of settings without renumbering physical extents; the `-n` option enables renumbering of physical extents.

2. Based on the information collected in the previous step, choose new values for the volume group parameters.
3. The new values may increase the size of the volume group reserved area (VGRA) on each physical volume. The VGRA resides in the LVM header, so increasing its size may require moving the first physical extent of any user data on physical volume. Use the `pvmove` command to move the first physical extent to another location.
4. Review the values by running `vgmodify` with the new settings and the `-r` option.
5. Deactivate the volume group.
6. Commit the new values by running `vgmodify` without the `-r` option.
7. Activate the volume group. Run the `vgdisplay` command to verify the settings have been applied.

As an example, you expect to add larger disks to the volume group `vg32`. You want to increase the maximum number of physical extents per physical volume (`max_pe`) and the maximum number of physical volumes (`max_pv`). Here are the steps involved:

1. Run `vgmodify` to collect information about the volume group.

   Save the output from these three commands:

```
# vgmodify -o -r vg32
Current Volume Group settings:
                                           Max LV      255
                                           Max PV       16
                                    Max PE per PV     1016
                                 PE Size (Mbytes)       32
                               VGRA Size (Kbytes)      176
New configuration requires "max_pes" are increased from 1016 to 6652
The current and new Volume Group parameters differ.
An update to the Volume Group IS required

New Volume Group settings:
                                           Max LV      255
                                           Max PV       16
                                    Max PE per PV     6652
                                 PE Size (Mbytes)       32
                               VGRA Size (Kbytes)      896
Review complete. Volume group not modified

# vgmodify -v -t vg32
Current Volume Group settings:
                                           Max LV      255
                                           Max PV       16
                                    Max PE per PV     1016
                                 PE Size (Mbytes)       32
                               VGRA Size (Kbytes)      176

VGRA space (Kbytes) on Physical Volumes with extents in use:
PV                       current     -n
/dev/rdisk/disk6          896       32768
/dev/rdisk/disk5          896       32768
Summary                   896       32768

Volume Group optimized settings (no PEs renumbered):
max_pv(-p) max_pe(-e) Disk size (Mb)
   2          53756     1720193
   3          35836     1146753
...
   213         296         9473
   255         252         8065

# vgmodify -v -n -t vg32
Volume Group configuration for /dev/vg32 has been saved in
 /etc/lvmconf/vg32.conf

Current Volume Group settings:
                                           Max LV      255
                                           Max PV       16
                                    Max PE per PV     1016
                                 PE Size (Mbytes)       32
                               VGRA Size (Kbytes)      176

VGRA space (Kbytes) on Physical Volumes with extents in use:
PV                       current     -n
/dev/rdisk/disk6          896       32768
/dev/rdisk/disk5          896       32768
Summary                   896       32768
Physical Extent zero is not free on all PVs. You will not achieve these
values until the first extent is made free (see pvmove(1M)) on all the
following disks:
/dev/rdisk/disk6
/dev/rdisk/disk5

Volume Group optimized settings (PEs renumbered lower):
max_pv(-p) max_pe(-e) Disk size (Mb)
61          65535      2097152
62          65532      2097056
...
```

```
252         16048       513568
255         15868       507808
```

2. Based on the output of `vgmodify -n -t`, choose 255 for `max_pv` and 15868 for `max_pe`.
3. Since the new values require physical extent 0 to be free, use `pvmove` to move it to another location:

```
# pvmove /dev/disk/disk5:0 /dev/disk/disk5
Transferring logical extents of logical volume "/dev/vg32/lvol2"...
Physical volume "/dev/disk/disk5" has been successfully moved.
Volume Group configuration for /dev/vg32 has been saved in
 /etc/lvmconf/vg32.conf

# pvmove /dev/disk/disk6:0 /dev/disk/disk6
Transferring logical extents of logical volume "/dev/vg32/lvol1"...
Physical volume "/dev/disk/disk6" has been successfully moved.
Volume Group configuration for /dev/vg32 has been saved in
 /etc/lvmconf/vg32.conf
```

4. Preview the changes by using the `-r` option to `vgmodify`:

```
# vgmodify -p 255 -e 15868 -r -n vg32
Current Volume Group settings:
                                          Max LV      255
                                          Max PV       16
                                   Max PE per PV     1016
                                PE Size (Mbytes)       32
                              VGRA Size (Kbytes)      176
The current and new Volume Group parameters differ.
An update to the Volume Group IS required

New Volume Group settings:
                                          Max LV      255
                                          Max PV      255
                                   Max PE per PV    15868
                                PE Size (Mbytes)       32
                              VGRA Size (Kbytes)    32640
Review complete. Volume group not modified
```

5. Deactivate the volume group:

```
# vgchange -a n vg32
Volume group "vg32" has been successfully changed.
```

6. Commit the new values:

```
# vgmodify -p 255 -e 15868 -n vg32
Current Volume Group settings:
                                          Max LV      255
                                          Max PV       16
                                   Max PE per PV     1016
                                PE Size (Mbytes)       32
                              VGRA Size (Kbytes)      176
The current and new Volume Group parameters differ.
An update to the Volume Group IS required

New Volume Group settings:
                                          Max LV      255
                                          Max PV      255
                                   Max PE per PV    15868
                                PE Size (Mbytes)       32
                              VGRA Size (Kbytes)    32640
New Volume Group configuration for "vg32" has been saved in
 "/etc/lvmconf/vg32.conf"
Old Volume Group configuration for "vg32" has been saved in
 "/etc/lvmconf/vg32.conf.old"
Starting the modification by writing to all Physical Volumes
Applying the configuration to all Physical Volumes from
 "/etc/lvmconf/vg32.conf"
Completed the modification process.
New Volume Group configuration for "vg32" has been saved in
 "/etc/lvmconf/vg32.conf.old"
Volume group "vg32" has been successfully changed.
```

7. Activate the volume group and verify the changes:

```
# vgchange -a y vg32
Activated volume group
Volume group "vg32" has been successfully changed.

# vgdisplay vg32
--- Volume groups ---
VG Name                     /dev/vg32
VG Write Access             read/write
VG Status                   available
Max LV                      255
Cur LV                      0
```

```
Open LV                      0
Max PV                       255
Cur PV                       2
Act PV                       2
Max PE per PV                15868
VGDA                         4
PE Size (Mbytes)             32
Total PE                     1084
Alloc PE                     0
Free PE                      1084
Total PVG                    0
Total Spare PVs              0
Total Spare PVs in use       0
VG Version                   1.0
```

Quiescing and Resuming a Volume Group

If you plan to use a disk management utility to create a backup image or "snapshot" of all the disks in a volume group, you must make sure that LVM is not writing to any of the disks when the snapshot is being taken; otherwise, some disks can contain partially written or inconsistent LVM metadata. To keep the volume group disk image in a consistent state, you must either deactivate the volume group or quiesce it.

Deactivating the volume group requires you to close all the logical volumes in the volume group, which can be disruptive. For example, you must unmount any file system using a logical volume in the volume group. However, temporarily quiescing the volume group enables you to keep the volume group activated and the logical volumes open during the snapshot operation, minimizing the impact to your system.

You can quiesce both read and write operations to the volume group, or just write operations. While a volume group is quiesced, the `vgdisplay` command reports the volume group access mode as `quiesced`. The indicated I/O operations queue until the volume group is resumed, and commands that modify the volume group configuration fail immediately.

NOTE: Individual physical volumes or logical volumes cannot be quiesced using this feature. To temporarily quiesce a physical volume to disable or replace it, see "Disabling a Path to a Physical Volume" (page 75). To quiesce a logical volume, quiesce or deactivate the volume group. To provide a stable image of a logical volume without deactivating the volume group, mirror the logical volume, then split off one of the mirrors, as described in "Backing Up a Mirrored Logical Volume" (page 62).

Quiescing a volume group is not persistent across reboots.

To quiesce a volume group, use the `vgchange` command with the `-Q` option as follows:

`#` **`vgchange -Q`** ***`mode`*** **`vg`*****`nn`***

The *`mode`* parameter can be either `rw`, which blocks both read and write operations, or `w`, which permits read operations but blocks write operations.

By default, the volume group remains quiesced until it is explicitly resumed. You can specify a maximum quiesce time in seconds using the `-t` option. If the quiesce time expires, the volume group is resumed automatically. For example, to quiesce volume group `vg08` for a maximum of ten minutes (600 seconds) but permitting read operations, enter the following command:

`#` **`vgchange -Q w -t 600 vg08`**

To resume a quiesced volume group, use the `vgchange` command with the `-R` option as follows:

`#` **`vgchange -R vg`*****`nn`***

Renaming a Volume Group

To change the name of a volume group, export it, then import it using the new name. For more detailed information on how to export and import a volume group, see "Exporting a Volume Group" (page 54) and "Importing a Volume Group" (page 54).

To rename the volume group `vg01` to `vgdb`, follow these steps:

1. Deactivate the volume group as follows:

   ```
   # vgchange -a n vg01
   ```

2. If you want to retain the same minor number for the volume group, examine the volume group's `group` file as follows:

   ```
   # ls -l /dev/vg01/group
   crw-r--r-- 1 root sys 64 0x010000 Mar 28  2004 /dev/vg01/group
   ```

   For this example, the volume group major number is 64, and the minor number is 0x010000.

3. Export the volume group as follows:

   ```
   # vgexport -m vg01.map vg01
   ```

4. If you are using an HP-UX release before March 2008, or if you want to specify the minor number of the volume group device file, create it for the volume group's new name, using the procedure in "Creating the Volume Group Device File" (page 46).

   Since the `group` file in this example has a major number of 64 and a minor number of 0x010000, enter the following commands:

   ```
   # mkdir /dev/vgdb
   # mknod /dev/vgdb/group c 64 0x010000
   ```

5. Import the volume group under its new name as follows:

   ```
   # vgimport -m vg01.map /dev/vgdb
   ```

6. Back up the volume group configuration information as follows:

   ```
   # vgcfgbackup /dev/vgdb
   ```

7. Activate the volume group as follows:

   ```
   # vgchange -a y /dev/vgdb
   ```

8. Remove saved configuration information based on the old volume group name as follows:

   ```
   # rm /etc/lvmconf/vg01.conf
   ```

9. Update all references to the old name in any other files on the system. These include `/etc/fstab` for mounted file systems or swap devices, and existing mapfiles from a `vgexport` command.

Splitting a Volume Group

You can use the `vgchgid` to split an existing volume group into two or more volume groups, provided that the physical volumes to be split are self-contained; that is, any logical volumes on the physical volumes must be wholly contained on those physical volumes. For example, a splittable volume group can have logical volumes 1, 2, and 3 on physical volumes 0 and 1, and logical volumes 4, 5, and 6 on physical volumes 2, 3, 4, and 5.

In this example, volume group `vgold` contains physical volumes `/dev/disk/disk0` through `/dev/disk/disk5`. Logical volumes `lvol1`, `lvol2`, and `lvol3` are on physical volumes `/dev/disk/disk0` and `/dev/disk/disk1`, and logical volumes `lvol4`, `lvol5`, and `lvol6` are on the remaining physical volumes.

To keep `/dev/disk/disk0` and `/dev/disk/disk1` in `vgold` and split the remaining physical volumes into a new volume group named `vgnew`, follow these steps:

1. Deactivate the volume group as follows:

   ```
   # vgchange -a n vgold
   ```

2. Export the volume group as follows:

```
# vgexport vgold
```

3. Change the VGID on the physical volumes to be assigned to the new volume group as follows:

```
# vgchgid -f /dev/rdisk/disk2 /dev/rdisk/disk3 \
    /dev/rdisk/disk4 /dev/rdisk/disk5
```

4. If you are using an HP-UX release before March 2008, or if you want to specify the minor number of the `vgold` group file, create it using the procedure in “Creating the Volume Group Device File” (page 46).
5. If you are using an HP-UX release before March 2008, or if you want to specify the minor number of the `vgnew` group file, create it using the procedure in “Creating the Volume Group Device File” (page 46).
6. Import the physical volumes in the old volume group as follows:

```
# vgimport /dev/vgold /dev/rdisk/disk0 /dev/rdisk/disk1
```

7. Import the physical volumes in the new volume group as follows:

```
# vgimport /dev/vgnew /dev/rdisk/disk2 /dev/rdisk/disk3 \
    /dev/rdisk/disk4 /dev/rdisk/disk5
```

8. Activate the volume groups. Disable quorum checks for the old volume group (it is missing over half its disks) as follows:

```
# vgchange -a y -q n /dev/vgold
# vgchange -a y /dev/vgnew
```

9. The logical volumes are currently defined in both volume groups. Remove the duplicate logical volumes from the volume group that no longer contains them as follows:

```
# lvremove -f vgold/lvol4 vgold/lvol5 vgold/lvol6
# lvremove -f vgnew/lvol1 vgnew/lvol2 vgnew/lvol3
```

10. The physical volumes are currently defined in both volume groups. Remove the missing physical volumes from both volume groups as follows:

```
# vgreduce -f vgold
# vgreduce -f vgnew
```

11. Enable quorum checks for the old volume group as follows:

```
# vgchange -a y -q y /dev/vgold
```

On completion, the original volume group contains three logical volumes (`lvol1`, `lvol2`, and `lvol3`) with physical volumes `/dev/disk/disk0` and `/dev/disk/disk1`. The new volume group `vgnew` contains three logical volumes (`lvol4`, `lvol5`, and `lvol6`) across physical volumes `/dev/disk/disk2`, `/dev/disk/disk3`, `/dev/disk/disk4`, and `/dev/disk/disk5`.

Removing a Volume Group

TIP: It is easier to export a volume group than to remove it, because removing the volume group requires that you remove all the logical volumes and physical volumes from the volume group before running `vgremove`. In addition, exporting the volume group leaves the LVM information on the disks untouched, which is an advantage if you want to reimport the volume group later. For the procedure to export a volume group, see “Exporting a Volume Group” (page 54).

To remove a volume group, follow these steps:

1. Back up all user data.
2. Find the names of all logical volumes and physical volumes in the volume group. Enter the following command:

```
# vgdisplay -v /dev/vgnn
```

3. Make sure that none of those logical volumes are in use. This may require stopping applications using any logical volumes in the volume group, and unmounting file systems contained in the volume group.

   Use the `fuser` command on each logical volume:

   ```
   # fuser -cu /dev/vgnn/lvoln
   ```

4. Remove each of the logical volumes as follows:

   ```
   # lvremove /dev/vgnn/lvoln
   ```

   For more information, see "Removing a Logical Volume" (page 53).

5. Remove all but one of the physical volumes as follows:

   ```
   # vgreduce /dev/vgnn /dev/disk/diskn
   ```

   For more information, see "Removing a Disk from a Volume Group" (page 48).

6. Use the `vgremove` command to remove the volume group as follows:

   ```
   # vgremove vgnn
   ```

Backing Up a Mirrored Logical Volume

NOTE: Mirroring requires the optional product HP MirrorDisk/UX.

You can split a mirrored logical volume into two logical volumes to perform a backup on an offline copy while the other copy stays online. When you complete the backup of the offline copy, you can merge the two logical volumes back into one. To bring the two copies back in synchronization, LVM updates the physical extents in the offline copy based on changes made to the copy that remained in use.

You can use HP SMH to split and merge logical volumes, or use the `lvsplit` and `lvmerge` commands.

To back up a mirrored logical volume containing a file system, using `lvsplit` and `lvmerge`, follow these steps:

1. Split the logical volume `/dev/vg00/lvol1` into two separate logical volumes as follows:

   ```
   # lvsplit /dev/vg00/lvol1
   ```

   This creates the new logical volume `/dev/vg00/lvol1b`. The original logical volume `/dev/vg00/lvol1` remains online.

2. Perform a file system consistency check on the logical volume to be backed up as follows:

   ```
   # fsck /dev/vg00/lvol1b
   ```

3. Mount the file system as follows:

   ```
   # mkdir /backup_dir
   # mount /dev/vg00/lvol1b /backup_dir
   ```

4. Perform the backup using the utility of your choice.
5. Unmount the file system as follows:

   ```
   # umount /backup_dir
   ```

6. Merge the split logical volume back with the original logical volume as follows:

   ```
   # lvmerge /dev/vg00/lvol1b /dev/vg00/lvol1
   ```

Backing Up and Restoring Volume Group Configuration

It is important that volume group configuration information be saved whenever you make *any* change to the configuration such as:

- Adding or removing disks to a volume group
- Changing the disks in a root volume group
- Creating or removing logical volumes
- Extending or reducing logical volumes

Unlike fixed disk partitions or nonpartitioned disks that begin and end at known locations on a given disk, each volume group configuration is unique, changes, and uses space on several disks.

If you back up your volume group configuration, you can restore a corrupted or lost LVM configuration in the event of a disk failure or corruption of your LVM configuration information.

The `vgcfgbackup` command creates or updates a backup file containing the volume group configuration; it *does not back up the data within your logical volumes*. To simplify the backup process, `vgcfgbackup` is invoked automatically whenever you make a configuration change as a result of using any of the following commands:

lvchange	lvcreate	lvextend	lvlnboot	lvmerge
lvreduce	lvremove	lvrmboot	lvsplit	vgcreate
pvchange	pvmove	vgextend	vgmodify	vgreduce

You can display LVM configuration information previously backed up with `vgcfgbackup` or restore it using `vgcfgrestore`.

By default, `vgcfgbackup` saves the configuration of a volume group to the file `/etc/lvmconf/`*`volume_group_name`*`.conf`.

You can also run `vgcfgbackup` at the command line, saving the backup file in any directory you indicate. If you do, first run `vgdisplay` with the `-v` option to ensure that the all logical volumes in the volume group are shown as `available/syncd`. If so, then run the following command:

```
# vgcfgbackup -f pathname/filename volume_group_name
```

If you use a nondefault volume group configuration file, take note of and retain its location. For information on command options, see *vgcfgbackup*(1M). Make sure backups of the root volume group are on the root file system, in case these are required during recovery.

To run `vgcfgrestore`, the physical volume must be detached. If all the data on the physical volume is mirrored and the mirror copies are current and available, you can temporarily detach the physical volume using `pvchange`, perform the `vgcfgrestore`, and reattach the physical volume. For example, to restore volume group configuration data for `/dev/disk/disk5`, a disk in the volume group `/dev/vgsales`, enter the following commands:

```
# pvchange -a n /dev/disk/disk5
# vgcfgrestore -n /dev/vgsales /dev/rdisk/disk5
# pvchange -a y /dev/disk/disk5
```

If the physical volume is not mirrored or the mirror copies are not current and available, you must deactivate the volume group with `vgchange`, perform the `vgcfgrestore`, and activate the volume group as follows:

```
# vgchange -a n /dev/vgsales
# vgcfgrestore -n /dev/vgsales /dev/rdisk/disk5
# vgchange -a y /dev/vgsales
```

These examples restore the LVM configuration to the disk from the default backup location in `/etc/lvmconf/vgsales.conf`.

For information on command options, see *vgcfgrestore*(1M).

Moving and Reconfiguring Your Disks

This section addresses the following topics:

You might need to do the following tasks:

- Move the disks in a volume group to different hardware locations on a system.
- Move entire volume groups of disks from one system to another.

CAUTION: Moving a disk that is part of your root volume group is not recommended. For more information, see *Configuring HP-UX for Peripherals*.

The `/etc/lvmtab` and `/etc/lvmtab_p` files contain information about the mapping of LVM disks on a system to volume groups; that is, volume group names and lists of the physical volumes included in volume groups. When you do either of the previous tasks, these configuration files must be changed to reflect the new hardware locations and device files for the disks. However, you cannot edit these files directly because they are not text files. Instead, you must use `vgexport` and `vgimport` to reconfigure the volume groups, which records configuration changes in the LVM configuration files.

Moving Disks Within a System

There a two procedures for moving the disks in a volume group to different hardware locations on a system. Choose a procedure depending on whether you use persistent or legacy device files for your physical volumes; the types of device files are described in "Legacy Device Files versus Persistent Device Files" (page 19).

LVM Configuration with Persistent Device Files

If your LVM configuration uses persistent device files, follow these steps:

1. Be sure that you have an up-to-date backup for both the data within the volume group and the volume group configuration.
2. Deactivate the volume group by entering the following command:

   ```
   # vgchange -a n /dev/vgnn
   ```

3. Physically move your disks to their desired new locations.
4. Activate the volume group as follows:

   ```
   # vgchange -a y /dev/vgnn
   ```

LVM Configuration with Legacy Device Files

The names of legacy device files change when the hardware paths to their physical devices change. Therefore, you must update the LVM configuration by exporting and importing the volume group to use the new legacy device files. Follow these steps:

1. Be sure that you have an up-to-date backup for both the data within the volume group and the volume group configuration.
2. Deactivate the volume group as follows:

```
# vgchange -a n /dev/vgnn
```

3. If you want to retain the same minor number for the volume group, examine the volume group's `group` file as follows:

```
# ls -l /dev/vgnn/group
crw-r--r-- 1 root sys 64 0x010000 Mar 28  2004 /dev/vgnn/group
```

   For this example, the volume group major number is 64, and the minor number is 0x010000.

4. Remove the volume group device files and its entry from the LVM configuration files by entering the following command:

```
# vgexport -v -s -m /tmp/vgnn.map /dev/vgnn
```

5. Physically move your disks to their desired new locations.
6. To view the new locations, enter the following command:

```
# vgscan -v
```

7. If you are using an HP-UX release before March 2008, or if you want to retain the minor number of the volume group device file, create it using the procedure in “Creating the Volume Group Device File” (page 46).

   Since the `group` file in this example has a major number of 64 and a minor number of 0x01000000, enter the following commands:

```
# mkdir /dev/vgnn
# mknod /dev/vgnn/group c 64 0x010000
```

8. Add the volume group entry back to the LVM configuration files using the `vgimport` command as follows:

```
# vgimport -v -s -m /tmp/vgnn.map /dev/vgnn
```

9. Activate the newly imported volume group as follows:

```
# vgchange -a y /dev/vgnn
```

10. Back up the volume group configuration as follows:

```
# vgcfgbackup /dev/vgnn
```

Moving Disks Between Systems

To move the disks in a volume group to different hardware locations on a different system, export the volume group from one system, physically move the disks to the other system, and import the volume group there. The procedures for exporting and importing a volume are described in “Exporting a Volume Group” (page 54) and “Importing a Volume Group” (page 54). They are illustrated in the following example.

NOTE: If the volume group contains any multipathed disks, see the note under “Importing a Volume Group” (page 54).

To move the three disks in the volume group `/dev/vg_planning` to another system, follow these steps:

1. If any of the logical volumes contain a file system, unmount the file system. If any of the logical volumes are used as secondary swap, disable swap and reboot the system; for information on secondary swap, see *HP-UX System Administrator's Guide: Configuration Management*.
2. Make the volume group and its associated logical volumes unavailable to users as follows:

```
# vgchange -a n /dev/vg_planning
```

3. Preview the removal of the volume group information from the LVM configuration files using the following command:

```
# vgexport -p -v -s -m /tmp/vg_planning.map /dev/vg_planning
```

With the `-m` option, you can specify the name of a map file that will hold the information that is removed from the LVM configuration files. The map file contains the names of all logical volumes in the volume group. You use this map file when you set up the volume group on the new system.

4. If the preview is satisfactory, remove the volume group information as follows:

   ```
   # vgexport -v -s -m /tmp/vg_planning.map /dev/vg_planning
   ```

   The `vgexport` command removes the volume group from the system and creates the `/tmp/vg_planning.map` file.

5. Connect the disks to the new system and copy the `/tmp/vg_planning.map` file to the new system.
6. If you are using an HP-UX release before March 2008, create the volume group device file using the procedure in "Creating the Volume Group Device File" (page 46).
7. To get device file information about the disks, run the `ioscan` command:

   ```
   # ioscan -funN -C disk
   ```

8. To preview the import operation, run the `vgimport` command with the `-p` option:

   ```
   # vgimport -p -N -v -s -m /tmp/vg_planning.map /dev/vg_planning
   ```

9. To import the volume group, run `vgimport` without the `-p` option as follows:

   ```
   # vgimport -N -v -s -m /tmp/vg_planning.map /dev/vg_planning
   ```

10. Activate the newly imported volume group as follows:

    ```
    # vgchange -a y /dev/vg_planning
    ```

Moving Data to a Different Physical Volume

You can use the `pvmove` command to move data contained in logical volumes from one disk to another disk or to move data between disks within a volume group.

For example, you can move the data from a specific logical volume from one disk to another, to use the vacated space on the first disk for another purpose. To move the data in logical volume `/dev/vg01/markets` from the disk `/dev/disk/disk4` to the disk `/dev/disk/disk7`, enter the following:

```
# pvmove -n /dev/vg01/markets /dev/disk/disk4 /dev/disk/disk7
```

On the other hand, you can move all the data contained on one disk, regardless of which logical volume it is associated with, to another disk within the same volume group. For example, do this to remove a disk from a volume group. You can use `pvmove` to move the data to other specified disks or let LVM move the data to appropriate available space within the volume group, subject to any mirroring allocation policies.

To move all data off disk `/dev/dsk/disk3` and relocate it at the destination disk `/dev/disk/disk5`, enter the following command:

```
# pvmove /dev/disk/disk3 /dev/disk/disk5
```

To move all data off disk `/dev/disk/disk3` and let LVM transfer the data to available space within the volume group, enter the following command:

```
# pvmove /dev/disk/disk3
```

In each of the previous instances, if space does not exist on the destination disk, the `pvmove` command fails.

NOTE: The `pvmove` command is not an atomic operation; it moves data extent by extent. If `pvmove` is abnormally terminated by a system crash or `kill -9`, the volume group can be left in an inconsistent configuration showing an additional pseudomirror copy for the extents being moved. You can remove the extra mirror copy using the `lvreduce` command with the `–m` option on each of the affected logical volumes; there is no need to specify a disk.

Creating a Spare Disk

NOTE: Disk sparing requires the optional product HP MirrorDisk/UX.

Version 2.x volume groups do not support disk sparing.

To configure a spare physical volume into a volume group for which you want protection against disk failure, follow these steps before a disk failure actually occurs:

1. Use the `pvcreate` command to initialize the disk as an LVM disk.

   NOTE: Do not use the `-B` option because spare physical volumes cannot contain boot information.

   ```
   # pvcreate /dev/rdisk/disk3
   ```

2. Ensure the volume group has been activated, as follows:

   ```
   # vgchange -a y /dev/vg01
   ```

3. Use the `vgextend` command with `-z y` to designate one or more physical volumes as spare physical volumes within the volume group. For example:

   ```
   # vgextend -z y /dev/vg01 /dev/disk/disk3
   ```

   Alternately, you can change a physical volume with no extents currently allocated within it into a spare physical volume using the `pvchange` command with the `-z y` option. For example:

   ```
   # pvchange -z y /dev/disk/disk3
   ```

For more information on disk sparing, see “Increasing Disk Redundancy Through Disk Sparing” (page 32).

Reinstating a Spare Disk

NOTE: Disk sparing requires the optional product HP MirrorDisk/UX.

Version 2.x volume groups do not support disk sparing.

After a failed disk has been repaired or a decision has been made to replace it, follow these steps to reinstate it and return the spare disk to its former standby status:

1. Physically connect the new or repaired disk.
2. Restore the LVM configuration to the reconnected disk using `vgcfgrestore` as follows:

   ```
   # vgcfgrestore -n /dev/vg01 /dev/rdisk/disk1
   ```

3. Ensure the volume group has been activated as follows:

   ```
   # vgchange -a y /dev/vg01
   ```

4. Be sure that allocation of extents is now allowed on the replaced disk as follows:

   ```
   # pvchange -x y /dev/disk/disk1
   ```

5. Use the pvmove command to move the data from the spare to the replaced physical volume. For example:

   ```
   # pvmove /dev/disk/disk3 /dev/disk/disk1
   ```

   The data from the spare disk is now back on the original disk or its replacement, and the spare disk is returned to its role as a standby empty disk.

For more information on disk sparing, see "Increasing Disk Redundancy Through Disk Sparing" (page 32).

Modifying Physical Volume Characteristics

NOTE: Version 2.x volume groups do not support bootable physical volumes.

The vgmodify command enables you to modify a volume group to adapt to changes in physical volumes. In particular, you can adjust the volume group to recognize size changes in physical volumes, and you can change a physical volume type between bootable and nonbootable.

Recognizing Size Changes

Disk arrays typically allow a LUN to be resized. If you increase the size of a LUN, follow these steps to incorporate the additional space into the volume group:

1. Increase the LUN size using the instructions for the array.
2. Run vgmodify to detect any physical volume size changes. It also reports whether all of the space can be made available to the volume group.
3. If vgmodify reports that the maximum number of physical extents per physical volume (max_pe) is too small to accommodate the new size, use vgmodify with the -t and -n options to determine a new value for max_pe, as described in "Modifying Volume Group Parameters" (page 55).
4. Review the values by running vgmodify with the new settings and the -r option.
5. Deactivate the volume group.
6. Commit any new value of max_pe and update the physical volume information by running vgmodify without the -r option.
7. Activate the volume group. To verify that the increased space is available, run the vgdisplay and pvdisplay commands.

TIP: Starting with the September 2008 release of HP-UX 11i Version 3, you can use the -E and -a options to vgmodify to recognize and accommodate size changes *without* deactivating the volume group. For more information, see *vgmodify*(1M).

For example, to increase the size of the physical volume /dev/rdisk/disk6 from 4 GB to 100000000 KB, follow these steps:

1. Increase the LUN size using the instructions for the disk array.
2. Run vgmodify with the -v and -r options to check whether any disks have changed in size and whether all of the space on the physical volumes can be used.

   ```
   # vgmodify -v -r vg32
   Current Volume Group settings:
                                             Max LV      255
                                             Max PV       16
                                      Max PE per PV     1016
                                   PE Size (Mbytes)       32
                                 VGRA Size (Kbytes)      176
   ```

```
/dev/rdisk/disk6 Warning: Max_PE_per_PV for the volume group
      (1016) too small for this PV (3051).
Using only 1016 PEs from this physical volume.

"/dev/rdisk/disk6" size changed from 4194304 to 100000000kb

An update to the Volume Group IS required

New Volume Group settings:
                                          Max LV      255
                                          Max PV       16
                                   Max PE per PV     1016
                                PE Size (Mbytes)       32
                              VGRA Size (Kbytes)      176
Review complete. Volume group not modified
```

The expanded physical volume requires 3051 physical extents to use all its space, but the current `max_pe` value limits this to 1016.

3. To determine the optimal values for `max_pv` and `max_pe`, run `vgmodify -t`, with and without the `-n` as follows:

```
# vgmodify -t vg32
Current Volume Group settings:
                                          Max LV      255
                                          Max PV       16
                                   Max PE per PV     1016
                                PE Size (Mbytes)       32
                              VGRA Size (Kbytes)      176
      VGRA space (Kbytes) without PE renumbering      896
        VGRA space (Kbytes) PE renumbering lower    32768

Volume Group optimized settings (no PEs renumbered):
        max_pv(-p) max_pe(-e) Disk size (Mb)
            2         53756     1720193
            3         35836     1146753
            4         26876     860033
            ...
            28         3836     122753
            30         3580     114561
            32         3324     106369
            35         3068     98177
            38         2812     89985
            ...
            255         252     8065
```

The table shows that without renumbering physical extents, a `max_pv` of 35 or lower permits a `max_pe` sufficient to accommodate the increased physical volume size.

```
# vgmodify -v -t -n vg32
Volume Group configuration for /dev/vg32 has been saved in
 /etc/lvmconf/vg32.conf

Current Volume Group settings:
                                          Max LV      255
                                          Max PV       16
                                   Max PE per PV     1016
                                PE Size (Mbytes)       32
                              VGRA Size (Kbytes)      176

VGRA space (Kbytes) on all Physical Volumes:
                PV                 current         -n
        /dev/rdisk/disk6            896             32768
        /dev/rdisk/disk5            896             32768
        Summary                     896             32768

Volume Group optimized settings (PEs renumbered lower):
```

```
max_pv(-p) max_pe(-e) Disk size (Mb)
    61       65535     2097152
    62       65532     2097056
    63       64252     2056096
    ...
    251      16124     516000
    252      16048     513568
    255      15868     507808
```

The table shows that if physical extents are renumbered, all values of `max_pv` permit a `max_pe` large enough to accommodate the increased physical volume size.

For this example, select a `max_pv` of 10, which permits a `max_pe` value of 10748.

4. Preview the changes by using the `-r` option to `vgmodify` as follows:

```
# vgmodify -p 10 -e 10748 -r vg32
Current Volume Group settings:
                                          Max LV      255
                                          Max PV       16
                                   Max PE per PV     1016
                                PE Size (Mbytes)       32
                              VGRA Size (Kbytes)      176
The current and new Volume Group parameters differ.
"/dev/rdisk/disk6" size changed from 4194304 to 100000000kb
An update to the Volume Group IS required

New Volume Group settings:
                                          Max LV      255
                                          Max PV       10
                                   Max PE per PV    10748
                                PE Size (Mbytes)       32
                              VGRA Size (Kbytes)      896
Review complete. Volume group not modified
```

5. Deactivate the volume group as follows:

```
# vgchange -a n vg32
Volume group "vg32" has been successfully changed.
```

6. Commit the new values as follows:

```
# vgmodify -p 10 -e 10748 vg32
Current Volume Group settings:
                                          Max LV      255
                                          Max PV       16
                                   Max PE per PV     1016
                                PE Size (Mbytes)       32
                              VGRA Size (Kbytes)      176
The current and new Volume Group parameters differ.
"/dev/rdisk/disk6" size changed from 4194304 to 100000000kb
An update to the Volume Group IS required

New Volume Group settings:
                                          Max LV      255
                                          Max PV       10
                                   Max PE per PV    10748
                                PE Size (Mbytes)       32
                              VGRA Size (Kbytes)      896
New Volume Group configuration for "vg32" has been saved in
 "/etc/lvmconf/vg32.conf"
Old Volume Group configuration for "vg32" has been saved in
 "/etc/lvmconf/vg32.conf.old"
Starting the modification by writing to all Physical Volumes
Applying the configuration to all Physical Volumes from
 "/etc/lvmconf/vg32.conf"
Completed the modification process.
New Volume Group configuration for "vg32" has been saved in
```

```
 "/etc/lvmconf/vg32.conf.old"
Volume group "vg32" has been successfully changed.
```

7. Activate the volume group and verify the changes by entering the following commands:

```
# vgchange -a y vg32
Activated volume group
Volume group "vg32" has been successfully changed.

# vgdisplay vg32
--- Volume groups ---
VG Name                     /dev/vg32
VG Write Access             read/write
VG Status                   available
Max LV                      255
Cur LV                      0
Open LV                     0
Max PV                      10
Cur PV                      2
Act PV                      2
Max PE per PV               10748
VGDA                        4
PE Size (Mbytes)            32
Total PE                    3119
Alloc PE                    0
Free PE                     3119
Total PVG                   0
Total Spare PVs             0
Total Spare PVs in use      0
VG Version                  1.0
```

CAUTION: This procedure can also be used when the size of a physical volume is decreased. However, there are limitations:

- To avoid data corruption, the size of the LUN (in the disk array) must be reduced only *after* `vgmodify` has successfully changed the volume group.
- The volume group must be deactivated before attempting any reduction. If you reduce the size of the LUN while the volume group is activated, LVM marks the physical volume as unavailable.
- If the physical volume has any allocated physical extents beyond the target size, `vgmodify` prints an error message and exits without changing the physical volume. In this case, you must be prepared to restore the LUN to its original size (ensuring the same disk space is allocated).

Changing Physical Volume Boot Types

NOTE: The `vgmodify` command does not support Version 2.x volume groups.

When a physical volume is initialized for LVM use, it can be made bootable or nonbootable. Bootable physical volumes require additional space in their LVM metadata for boot utilities and information. If a physical volume was accidentally initialized as bootable, you can convert the disk to a nonbootable disk and reclaim LVM metadata space.

CAUTION: The boot volume group requires at least one bootable physical volume. Do not convert all of the physical volumes in the boot volume group to nonbootable, or your system will not boot.

To change a disk type from bootable to nonbootable, follow these steps:

1. Use `vgcfgrestore` to determine if the volume group contains any bootable disks.

2. Run `vgmodify` twice, once with the `-B n` and once without it. Compare the available values for `max_pe` and `max_pv`.
3. Choose new values for `max_pe` and `max_pv`. Review the values by running `vgmodify` with the new settings and the `-r` option.
4. Deactivate the volume group.
5. Commit the changes by running `vgmodify` without the `-r` option.
6. Activate the volume group. Run the `vgcfgrestore` or `pvdisplay` commands to verify that the disk type has changed.

For example, to convert any bootable disks in volume group `vg`, follow these steps:

1. Check if any physical volumes in `vg01` are bootable as follows:

```
# vgcfgrestore -l -v -n vg01
Volume Group Configuration information in "/etc/lvmconf/vg01.conf"
VG Name /dev/vg01
---- Physical volumes : 1 ----
PV      Type     Size (kb) Start (kb) PVkey
c2t1d0 Bootable 35566480   2912       0

max_pv 16 max_pe 1085 max_lv 255
```

2. To determine which values of `max_pe` and `max_pv` are available, run the following command:

```
# vgmodify -t -B n vg01 /dev/rdsk/c2t1d0
Current Volume Group settings:
                                          Max LV      255
                                          Max PV       16
                                   Max PE per PV     1085
                                PE Size (Mbytes)       32
                              VGRA Size (Kbytes)      208
      VGRA space (Kbytes) without PE renumbering     2784
        VGRA space (Kbytes) PE renumbering lower    32768

Volume Group optimized settings (no PEs renumbered):
max_pv(-p) max_pe(-e) Disk size (Mb)
5          65535         2097122
6          56828         1818498
...
255         1276           40834
```

Compare the values if the disk is made non-bootable, and if it is not. Enter the following command:

```
# vgmodify -t vg01
Current Volume Group settings:
                                          Max LV      255
                                          Max PV       16
                                   Max PE per PV     1085
                                PE Size (Mbytes)       32
                              VGRA Size (Kbytes)      208
      VGRA space (Kbytes) without PE renumbering      768
        VGRA space (Kbytes) PE renumbering lower      768

Volume Group optimized settings (no PEs renumbered):
max_pv(-p) max_pe(-e) Disk size (Mb)
1          65535       2097120
2          45820       1466240
...
255          252          8064
```

If you change the disk type, the VGRA space available increases from 768 KB to 2784KB (if physical extents are not renumbered) or 32768 KB (if physical extents are renumbered). Changing the disk type also permits a larger range of `max_pv` and `max_pe`. For example, if `max_pv` is 255, the bootable disk can only accommodate a disk size of 8064 MB, but after conversion to nonbootable, it can accommodate a disk size of 40834 MB.

3. For this example, select a `max_pv` value of 6, which permits a `max_pe` value of 56828. Preview the changes by entering the following command:

```
# vgmodify -r -p 6 -e 56828 -B n vg01 /dev/rdsk/c2t1d0
Current Volume Group settings:
                                          Max LV      255
                                          Max PV       16
                                   Max PE per PV     1085
                                PE Size (Mbytes)       32
                              VGRA Size (Kbytes)      208
The current and new Volume Group parameters differ.
An update to the Volume Group IS required

New Volume Group settings:
Current Volume Group settings:
                                          Max LV      255
                                          Max PV        6
                                   Max PE per PV    56828
                                PE Size (Mbytes)       32
                              VGRA Size (Kbytes)     2784
Review complete. Volume group not modified
```

4. Deactivate the volume group as follows:

```
# vgchange -a n vg01
Volume group "vg01" has been successfully changed.
```

5. Commit the new values as follows:

```
# vgmodify -p 6 -e 56828 -B n vg01 /dev/rdsk/c2t1d0
Current Volume Group settings:
                                          Max LV      255
                                          Max PV       16
                                   Max PE per PV     1085
                                PE Size (Mbytes)       32
                              VGRA Size (Kbytes)      208
The current and new Volume Group parameters differ.
An update to the Volume Group IS required

New Volume Group settings:
Current Volume Group settings:
                                          Max LV      255
                                          Max PV        6
                                   Max PE per PV    56828
                                PE Size (Mbytes)       32
                              VGRA Size (Kbytes)     2784
New Volume Group configuration for "vg01" has been saved in
 "/etc/lvmconf/vg01.conf"
Old Volume Group configuration for "vg01" has been saved in
 "/etc/lvmconf/vg01.conf.old"

Starting the modification by writing to all Physical Volumes
Applying the configuration to all Physical Volumes from
 "/etc/lvmconf/vg01.conf"
Completed the modification process.
New Volume Group configuration for "vg01" has been saved in
 "/etc/lvmconf/vg01.conf.old"
Volume group "vg01" has been successfully changed.
```

6. Activate the volume group and verify the changes as follows:

```
# vgchange -a y vg01
Activated volume group
Volume group "vg01" has been successfully changed.

# vgcfgbackup vg01
Volume Group configuration for /dev/vg01 has been saved in
 /etc/lvmconf/vg01.conf

# vgcfgrestore -l -v -n vg01
Volume Group Configuration information in "/etc/lvmconf/vg01.conf"
VG Name /dev/vg01

---- Physical volumes : 1 ----
PV      Type     Size (kb) Start (kb) PVkey
c2t1d0 Non-Boot 35566480   2912       0

max_pv 6 max_pe 56828 max_lv 255
```

Disabling a Path to a Physical Volume

IMPORTANT: This procedure only disables *LVM's* use of the link. The `pvchange` command does not prevent diagnostics or an application from accessing the physical volume.

By default, the mass storage stack uses all the paths available to access a physical volume, independently of the paths configured in LVM. Disabling a path in LVM does not prevent the native multipathing from using that path. Use the `scsimgr` command to disable I/O along a path or to disable the native multipathing.

You can temporarily disable LVM's use of one or all of the physical paths to a physical volume using the `pvchange` command. Disabling a path, also known as **detaching the link**, causes LVM to close that path to the device and stop using it. This can be useful if you want to guarantee that a link is idle, such as when you are running diagnostics on an I/O card, replacing an I/O card, or replacing the disk containing the physical volume.

Detaching a link to a physical volume is a temporary operation, not a permanent one. If you want to permanently remove a link or physical volume from the volume group, use `vgreduce` instead, as described in "Removing a Disk from a Volume Group" (page 48).

To detach a link to a physical volume, use the `-a` option to `pvchange`. For example, to disable the link through the device `/dev/disk/disk33`, enter the following command:

```
# pvchange -a n /dev/disk/disk33
```

If you are using LVM's alternate links for multipathed disks, each link uses a different legacy device files. In that situation, to detach all links to a physical volume, use `N` as the argument to the `-a` option:

```
# pvchange -a N /dev/dsk/c5t0d0
```

Detaching one or more links to a physical volume does not necessarily cause LVM to stop using that physical volume entirely. If the detached link is the primary path to the device, LVM begins using any available alternate link to it. LVM stops using the physical volume only when all the links to it are detached.

If all the links to a device are detached, the associated physical volume becomes unavailable to the volume group. The links remain associated with the volume group but LVM does not send any I/O requests to the physical volume until it is reattached. This means that the data on that physical volume becomes temporarily unavailable; consequently, you must make sure that any availability requirements for that data can be satisfied by mirroring before you make the device unavailable by detaching it.

Detaching a link does not disable sparing. That is, if all links to a physical volume are detached and a suitable spare physical volume is available in the volume group, LVM uses it to reconstruct the detached disk. For more information on sparing, see "Increasing Disk Redundancy Through Disk Sparing" (page 32).

You can view the LVM status of all links to a physical volume using `vgdisplay` with the `-v` option.

Restoring a detached link to a physical volume, or **reattaching** it, makes that link available to the volume group. LVM can begin using that link as necessary to access the disk.

To reattach a specific path to a physical volume, use the `pvchange` command with the `-a` option. For example, enter the following command:

```
# pvchange -a y /dev/dsk/c5t0d0
```

Because detaching a link to a physical volume is temporary, all detached links in a volume group are reattached when the volume group is activated, either at boot time or with an explicit `vgchange` command, such as the following:

```
# vgchange -a y /dev/vg02
```

Creating an Alternate Boot Disk

NOTE: Version 2.x volume groups do not support bootable physical volumes. You cannot create an alternate boot disk in a Version 2.x volume group.

With non-LVM disks, a single root disk contains all the attributes needed for boot, system files, primary swap, and dump. Using LVM, a single root disk is replaced by a pool of disks, a **root volume group**, which contains all of the same elements but allowing a root logical volume, a boot logical volume, a swap logical volume, and one or more dump logical volumes. Each of these logical volumes must be contiguous, that is, contained on a single disk, and they must have bad block relocation disabled. (Other noncontiguous logical volumes can be used for user data.) For more information on the swap and dump devices and their configuration, see *HP-UX System Administrator's Guide: Configuration Management*.

The root logical volume contains the operating system software and the root file system (`/`).. The boot logical volume contains the boot file system (`/stand`). You can combine the root and boot logical volumes into a single logical volume or keep them separate. Whether you use a single combined root-boot logical volume, or separate root and boot logical volumes, the logical volume used to boot the system must be the first logical volume on its physical volume. It must begin at physical extent 0000 to boot the system in maintenance mode.

If you newly install your HP-UX system and choose the LVM configuration, a root volume group is automatically configured (`/dev/vg00`), as are separate root (`/dev/vg00/lvol3`) and boot (`/dev/vg00/lvol1`) logical volumes. If you currently have a combined root and boot logical volume and you want to reconfigure to separate them after creating the boot logical volume, use the `lvlnboot` command with the `-b` option to define the boot logical volume to the system, taking effect the next time the system is booted.

The swap logical volume is the system's primary swap area and is typically used for dump. The swap logical volume is often on the same physical disk as the root logical volume. However, you can configure it (and the dump logical volumes) on a different physical disk than the root logical volume.

If you create your root volume group with multiple disks, use the `lvextend` command to place the boot, root, and primary swap logical volumes on the boot disk.

TIP: You can use pvmove to move the data from an existing logical volume to another disk if necessary to make room for the root logical volume. For more information, see "Moving Data to a Different Physical Volume" (page 66).

To create a new root volume group with an alternate boot disk, follow these steps:

1. Create a bootable physical volume.
   a. On an HP Integrity server, partition the disk using the idisk command and a partition description file, then run insf as described in "Mirroring the Boot Disk on HP Integrity Servers" (page 81).
   b. Run pvcreate with the -B option. On an HP Integrity server, use the device file denoting the HP-UX partition:

      ```
      # pvcreate -B /dev/rdisk/disk6_p2
      ```

      On an HP 9000 server, use the device file for the entire disk:

      ```
      # pvcreate -B /dev/rdisk/disk6
      ```

2. Create a directory for the volume group. For example:

   ```
   # mkdir /dev/vgroot
   ```

3. Create a device file named group in the previously described directory as follows:

   ```
   # mknod /dev/vgroot/group c 64 0xnn0000
   ```

4. Create the root volume group, specifying each physical volume to be included, as follows:

   ```
   # vgcreate /dev/vgroot /dev/disk/disk6
   ```

5. Place boot utilities in the boot area as follows:

   ```
   # mkboot /dev/rdisk/disk6
   ```

6. Add an autoboot file to the disk boot area as follows:

   ```
   # mkboot -a "hpux" /dev/rdisk/disk6
   ```

7. Create the boot logical volume. To create a boot logical volume named bootlv with size 512 MB, enter the following commands:

   ```
   # lvcreate -C y -r n -n bootlv /dev/vgroot
   # lvextend –L 512 /dev/vgroot/bootlv /dev/disk/disk6
   ```

8. Create the primary swap logical volume. For example, to create a primary swap logical volume named swaplv with size 2 GB on the same disks as the boot logical volume, enter the following commands:

   ```
   # lvcreate -C y –r n -n swaplv /dev/vgroot
   # lvextend –L 2048 /dev/vgroot/swaplv /dev/disk/disk6
   ```

9. Create the root logical volume. For example, to create a root logical volume named rootlv with size 1 GB, enter the following commands:

   ```
   # lvcreate -C y –r n -n rootlv /dev/vgroot
   # lvextend –L 1024 /dev/vgroot/rootlv /dev/disk/disk6
   ```

10. Specify that bootlv is the boot logical volume as follows:

    ```
    # lvlnboot -b /dev/vgroot/bootlv
    ```

11. Specify that rootlv is the root logical volume as follows:

    ```
    # lvlnboot -r /dev/vgroot/rootlv
    ```

12. Specify that swaplv is the primary swap logical volume as follows:

    ```
    # lvlnboot -s /dev/vgroot/swaplv
    ```

13. Specify that swaplv is also to be used for dump as follows:

    ```
    # lvlnboot -d /dev/vgroot/swaplv
    ```

14. Verify the configuration as follows:

```
# lvlnboot -v /dev/vgroot
Boot Definitions for Volume Group /dev/vgroot:
Physical Volumes belonging in Root Volume Group:
                /dev/disk/disk6 -- Boot Disk
Boot: bootlv    on:     /dev/disk/disk6
Root: rootlv    on:     /dev/disk/disk6
Swap: swaplv    on:     /dev/disk/disk6
Dump: swaplv    on:     /dev/disk/disk6, 0
```

15. Once the boot and root logical volumes are created, create file systems for them. For example:

```
# mkfs -F hfs /dev/vgroot/rbootlv
# mkfs -F vxfs /dev/vgroot/rrootlv
```

NOTE: On HP Integrity servers, the boot file system can be VxFS. Enter the following command:

```
# mkfs -F vxfs /dev/vgroot/rbootlv
```

Mirroring the Boot Disk

NOTE: Mirroring requires the optional product, HP MirrorDisk/UX.

Version 2.x volume groups do not support boot disks, so you cannot mirror the boot disk in a Version 2.x volume group.

After you create mirror copies of the root, boot, and primary swap logical volumes, if any of the underlying physical volumes fail, the system can use the mirror copy on the other disk and continue. When the failed disk comes back online, it is automatically recovered, provided the system has not been rebooted.

If the system reboots before the disk is back online, reactivate the volume group to update the LVM data structures that track the disks within the volume group. You can use `vgchange -a y` even though the volume group is already active.

For example, you can reactivate volume group `vg00` by entering the following command:

`#` **`vgchange -a y /dev/vg00`**

As a result, LVM scans and activates all available disks in the volume group `vg00`, including the disk that came online after the system rebooted.

The procedure for creating a mirror of the boot disk is different for HP 9000 and HP Integrity servers. HP Integrity servers use partitioned boot disks.

NOTE: These examples include creating a mirror copy of the primary swap logical volume. The primary swap mirror does not need to be on a specific disk or at a specific location, but it must be allocated on contiguous disk space. The recommended mirror policy for primary swap is to have the mirror write cache and the mirror consistency recovery mechanisms disabled.

When primary swap is mirrored and your primary swap device also serves as a dump area, be sure that mirror write cache and mirror consistency recovery are set to off at boot time to avoid losing your dump. To reset these options, reboot your system in maintenance mode and use the `lvchange` command with the `-M n` and `-c n` options.

Mirroring the Boot Disk on HP 9000 Servers

To set up a mirrored root configuration, you must add a disk to the root volume group, mirror all the root logical volumes onto it, and make it bootable. For this example, the disk to be added is at path 0/1/1/0.0x1.0x0 and has device special files named `/dev/rdisk/disk4` and `/dev/disk/disk4`. Follow these steps:

1. Make sure the device files are in place. For example:

   `#` **`insf -e -H 0/1/1/0.0x1.0x0`**

   The following device files now exist for this disk:

   `/dev/[r]disk/disk4`

2. Create a bootable physical volume as follows:

   `#` **`pvcreate -B /dev/rdisk/disk4`**

3. Add the physical volume to your existing root volume group as follows:

   `#` **`vgextend /dev/vg00 /dev/disk/disk4`**

4. Place boot utilities in the boot area as follows:

   `#` **`mkboot /dev/rdisk/disk4`**

5. Add an autoboot file to the disk boot area as follows:

   `#` **`mkboot -a "hpux" /dev/rdisk/disk4`**

NOTE: If you expect to boot from this disk only when you lose quorum, you can use the alternate string `hpux -lq` to disable quorum checking. However, HP recommends configuring your root volume group to minimize quorum losses, by using at least three physical volumes and no single points of failure, as described in "Planning for Recovery" (page 38).

6. The logical volumes on the mirror boot disk must be extended in the same order that they are configured on the original boot disk. Determine the list of logical volumes in the root volume group and their order. For example:

```
# pvdisplay -v /dev/disk/disk0 | grep 'current.*0000 $'
   00000 current /dev/vg00/lvol1 00000
   00038 current /dev/vg00/lvol2 00000
   00550 current /dev/vg00/lvol3 00000
   00583 current /dev/vg00/lvol4 00000
   00608 current /dev/vg00/lvol5 00000
   00611 current /dev/vg00/lvol6 00000
   00923 current /dev/vg00/lvol7 00000
   01252 current /dev/vg00/lvol8 00000
```

7. Mirror each logical volume in `vg00` (the root volume group) onto the specified physical volume. For example:

```
# lvextend -m 1 /dev/vg00/lvol1 /dev/disk/disk4
The newly allocated mirrors are now being synchronized.
This operation will take some time. Please wait ....
# lvextend -m 1 /dev/vg00/lvol2 /dev/disk/disk4
The newly allocated mirrors are now being synchronized.
This operation will take some time. Please wait ....
# lvextend -m 1 /dev/vg00/lvol3 /dev/disk/disk4
The newly allocated mirrors are now being synchronized.
This operation will take some time. Please wait ....
# lvextend -m 1 /dev/vg00/lvol4 /dev/disk/disk4
The newly allocated mirrors are now being synchronized.
This operation will take some time. Please wait ....
# lvextend -m 1 /dev/vg00/lvol5 /dev/disk/disk4
The newly allocated mirrors are now being synchronized.
This operation will take some time. Please wait ....
# lvextend -m 1 /dev/vg00/lvol6 /dev/disk/disk4
The newly allocated mirrors are now being synchronized.
This operation will take some time. Please wait ....
# lvextend -m 1 /dev/vg00/lvol7 /dev/disk/disk4
The newly allocated mirrors are now being synchronized.
This operation will take some time. Please wait ....
# lvextend -m 1 /dev/vg00/lvol8 /dev/disk/disk4
The newly allocated mirrors are now being synchronized.
This operation will take some time. Please wait ....
```

TIP: To shorten the time required to synchronize the mirror copies, use the `lvextend` and `lvsync` command options introduced in the September 2007 release of HP-UX 11i Version 3. These options enable you to resynchronize logical volumes in parallel rather than serially. For example:

```
# lvextend -s -m 1 /dev/vg00/lvol1 /dev/disk/disk4
# lvextend -s -m 1 /dev/vg00/lvol2 /dev/disk/disk4
# lvextend -s -m 1 /dev/vg00/lvol3 /dev/disk/disk4
# lvextend -s -m 1 /dev/vg00/lvol4 /dev/disk/disk4
# lvextend -s -m 1 /dev/vg00/lvol5 /dev/disk/disk4
# lvextend -s -m 1 /dev/vg00/lvol6 /dev/disk/disk4
# lvextend -s -m 1 /dev/vg00/lvol7 /dev/disk/disk4
# lvextend -s -m 1 /dev/vg00/lvol8 /dev/disk/disk4
# lvsync -T /dev/vg00/lvol*
```

8. Update the root volume group information as follows:

   ```
   # lvlnboot -R /dev/vg00
   ```

9. Verify that the mirrored disk is displayed as a boot disk and that the boot, root, and swap logical volumes appear to be on both disks as follows:

   ```
   # lvlnboot -v
   ```

10. Specify the mirror disk as the alternate boot path in nonvolatile memory as follows:

    ```
    # setboot -a 0/1/1/0.0x1.0x0
    ```

11. Add a line to `/stand/bootconf` for the new boot disk using `vi` or another text editor as follows:

    ```
    # vi /stand/bootconf
    l /dev/disk/disk4
    ```

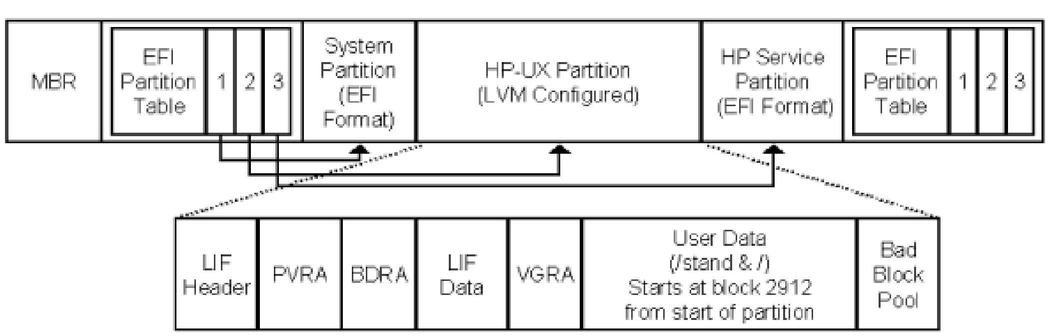
    Where the literal "`l`" (lower case L) represents LVM.

Mirroring the Boot Disk on HP Integrity Servers

The procedure to mirror the root disk on Integrity servers is similar to the procedure for HP 9000 servers. The difference is that Integrity server boot disks are partitioned; you must set up the partitions, copy utilities to the EFI partition, and use the HP-UX partition device files for LVM commands.

Figure 3-1shows the disk layout of a boot disk. The disk contains a Master Boot Record (MBR) and EFI partition tables that point to each of the partitions. The `idisk` command creates the partitions (see *idisk*(1M)).

Figure 3-1 Sample LVM Disk Layout on an HP Integrity Server



For this example, the disk to be added is at hardware path 0/1/1/0.0x1.0x0, with device special files named `/dev/disk/disk2` and `/dev/rdisk/disk2`. Follow these steps:

1. Partition the disk using the `idisk` command and a partition description file.
   a. Create a partition description file. For example:

      ```
      # vi /tmp/idf
      ```

      In this example, the partition description file contains the following information:

      ```
      3
      EFI 500MB
      HPUX 100%
      HPSP 400MB
      ```

      The values in the example represent a boot disk with three partitions: an EFI partition, an HP-UX partition, and an HPSP. Boot disks of earlier HP Integrity servers might have an EFI partition of only 100 MB and might not contain the HPSP partition.

   b. Partition the disk using `idisk` and your partition description file as follows:

      ```
      # idisk -f /tmp/idf -w /dev/rdisk/disk2
      ```

   c. To verify that your partitions are correctly laid out, enter the following command:

      ```
      # idisk /dev/rdisk/disk2
      ```

2. Create the device files for all the partitions. For example:

   ```
   # insf -e -H 0/1/1/0.0x1.0x0
   ```

   The following device files now exist for this disk:

   ```
   /dev/[r]disk/disk2  (this refers to the entire disk)
   /dev/[r]disk/disk2_p1(this refers to the efi partition)
   /dev/[r]disk/disk2_p2(this will be the hp-ux partition)
   /dev/[r]disk/disk2_p3(this refers to the service partition)
   ```

3. Create a bootable physical volume using the device file denoting the HP-UX partition. For example:

   ```
   # pvcreate -B /dev/rdisk/disk2_p2
   ```

4. Add the physical volume to your existing root volume group as follows:

   ```
   # vgextend vg00 /dev/disk/disk2_p2
   ```

5. Place boot utilities in the boot area. Copy EFI utilities to the EFI partition, and use the device special file for the entire disk as follows:

   ```
   # mkboot -e -l /dev/rdisk/disk2
   ```

6. Add an autoboot file to the disk boot area as follows:

   ```
   # mkboot -a "hpux" /dev/rdisk/disk2
   ```

NOTE: If you expect to boot from this disk only when you lose quorum, you can use the alternate string `hpux –lq` to disable quorum checking. However, HP recommends configuring your root volume group to minimize quorum losses, by using at least three physical volumes and no single points of failure, as described in “Planning for Recovery” (page 38).

7. The logical volumes on the mirror boot disk must be extended in the same order that they are configured on the original boot disk. Determine the list of logical volumes in the root volume group and their order. For example:

   ```
   # pvdisplay -v /dev/disk/disk0_p2 | grep 'current.*0000 $'
      00000 current /dev/vg00/lvol1 00000
      00010 current /dev/vg00/lvol2 00000
      00138 current /dev/vg00/lvol3 00000
      00151 current /dev/vg00/lvol4 00000
      00158 current /dev/vg00/lvol5 00000
   ```

```
00159 current /dev/vg00/lvol6 00000
00271 current /dev/vg00/lvol7 00000
00408 current /dev/vg00/lvol8 00000
```

8. Mirror each logical volume in `vg00` (the root volume group) onto the specified physical volume. For example:

```
# lvextend -m 1 /dev/vg00/lvol1 /dev/disk/disk2_p2
The newly allocated mirrors are now being synchronized.
This operation will take some time. Please wait ....
# lvextend -m 1 /dev/vg00/lvol2 /dev/disk/disk2_p2
The newly allocated mirrors are now being synchronized.
This operation will take some time. Please wait ....
# lvextend -m 1 /dev/vg00/lvol3 /dev/disk/disk2_p2
The newly allocated mirrors are now being synchronized.
This operation will take some time. Please wait ....
# lvextend -m 1 /dev/vg00/lvol4 /dev/disk/disk2_p2
The newly allocated mirrors are now being synchronized.
This operation will take some time. Please wait ....
# lvextend -m 1 /dev/vg00/lvol5 /dev/disk/disk2_p2
The newly allocated mirrors are now being synchronized.
This operation will take some time. Please wait ....
# lvextend -m 1 /dev/vg00/lvol6 /dev/disk/disk2_p2
The newly allocated mirrors are now being synchronized.
This operation will take some time. Please wait ....
# lvextend -m 1 /dev/vg00/lvol7 /dev/disk/disk2_p2
The newly allocated mirrors are now being synchronized.
This operation will take some time. Please wait ....
# lvextend -m 1 /dev/vg00/lvol8 /dev/disk/disk2_p2
The newly allocated mirrors are now being synchronized.
This operation will take some time. Please wait ....
```

NOTE: If `lvextend` fails with following message:

```
"m": illegal option
```

HP MirrorDisk/UX is not installed.

TIP: To shorten the time required to synchronize the mirror copies, use the `lvextend` and `lvsync` command options introduced in the September 2007 release of HP-UX 11i Version 3. These options enable you to resynchronize logical volumes in parallel rather than serially. For example:

```
# lvextend -s -m 1 /dev/vg00/lvol1 /dev/disk/disk2_p2
# lvextend -s -m 1 /dev/vg00/lvol2 /dev/disk/disk2_p2
# lvextend -s -m 1 /dev/vg00/lvol3 /dev/disk/disk2_p2
# lvextend -s -m 1 /dev/vg00/lvol4 /dev/disk/disk2_p2
# lvextend -s -m 1 /dev/vg00/lvol5 /dev/disk/disk2_p2
# lvextend -s -m 1 /dev/vg00/lvol6 /dev/disk/disk2_p2
# lvextend -s -m 1 /dev/vg00/lvol7 /dev/disk/disk2_p2
# lvextend -s -m 1 /dev/vg00/lvol8 /dev/disk/disk2_p2
# lvsync -T /dev/vg00/lvol*
```

9. Update the root volume group information as follows:

```
# lvlnboot -R /dev/vg00
```

10. Verify that the mirrored disk is displayed as a boot disk and that the boot, root, and swap logical volumes appear to be on both disks as follows:

```
# lvlnboot -v
```

11. Specify the mirror disk as the alternate boot path in nonvolatile memory as follows:

```
# setboot -a 0/1/1/0.0x1.0x0
```

12. Add a line to `/stand/bootconf` for the new boot disk using `vi` or another text editor as follows:

    ```
    # vi /stand/bootconf
    l /dev/disk/disk2_p2
    ```

    Where the literal "`l`" (lower case L) represents LVM.

Administering File System Logical Volumes

This section describes special actions you must take when working with file systems inside logical volumes. It addresses the following topics:

- "Creating a File System" (page 84)
- "Extending a File System" (page 85)
- "Reducing the Size of a File System" (page 86)
- "Backing Up a VxFS Snapshot File System" (page 88)

TIP: When dealing with file systems, you can use HP SMH or a sequence of HP-UX commands. For most tasks, using HP SMH is quicker and simpler. You do not have to explicitly perform each of the following distinct tasks; rather, proceed from the HP SMH Disk and File Systems area. HP SMH performs all the necessary steps for you.

Creating a File System

When creating either an HFS or VxFS file system in a logical volume, you can use HP SMH or a sequence of HP-UX commands. If you choose to use HP-UX commands directly, the following list describes the subtasks for creating a file system.

1. If you create a new file system of a type other than HFS, you might need to reconfigure the new type into the kernel. Normally, VxFS has been configured into the kernel as part of the default configuration. See *HP-UX System Administrator's Guide: Configuration Management* for information on how to add a file system type.
2. Estimate the size required for the logical volume. To estimate the size requirements for a logical volume containing a file system, see "Setting Up Logical Volumes for File Systems" (page 26).
3. Determine if there is sufficient disk space available in the volume group. Use the `vgdisplay` command to calculate this information, as described in "Creating a Logical Volume" (page 49).

   If there is not enough space within the volume group, you can add a disk to the volume group, as described in "Adding a Disk to a Volume Group" (page 48).
4. Create the logical volume. Use `lvcreate`, as described in "Creating a Logical Volume" (page 49).

5. Create the file system using the character device file. For example:

   ```
   # newfs -F fstype /dev/vg02/rlvol1
   ```

   If you do not use the `-F fstype` option, then `newfs` creates a file system based on the content of your `/etc/fstab` file. If there is no entry for the file system in `/etc/fstab`, then the file system type is determined from the file `/etc/default/fs`. For information on additional options, see *newfs*(1M).

   When creating a VxFS file system, file names will be long automatically.

   For HFS, use the `-s` or `-l` option to specify a file system with short or long file names, respectively. By default, the length of file system names are consistent with those of the root file system. Short file names are 14 characters maximum. Long file names allow up to 255 characters. HP recommends using long file names for flexibility; files created on other systems that use long file names can be moved to your system without being renamed.

6. After you have created a file system, mount it for users to access it, and add it to `/etc/fstab` so that it is automatically mounted at boot time.

Extending a File System

Extending a file system inside a logical volume is a two-step task: extending the logical volume, then extending the file system. The first step is described in "Extending a Logical Volume" (page 50). The second step, extending the file system itself, depends on the following factors:

- What type of file system is involved? If it is HFS or VxFS? HFS requires the file system to be unmounted to be extended.

  Check the type of file system. For example:

  ```
  # /usr/sbin/fstyp /dev/vg01/lvol2
  vxfs
  ```

- If the file system is VxFS, do you have the base VxFS product or the OnlineJFS product? If you have only the base VxFS product, you must unmount the file system before extending it.

  To see if the OnlineJFS product is installed, enter the following command:

  ```
  # swlist -l product | grep -i OnlineJFS
    OnlineJFS  B.11.31   Online features of the VxFS File System
  ```

- Can you unmount the file system? To unmount system directories such as`/var` and `/usr`, you must be in single-user mode.
- Is the file system the root file system (`/`)? If so, there are two complications:
  — The logical volume containing the root file system is created with the contiguous allocation policy, so it might not be possible to extend it in place.
  — The root file system cannot ever be unmounted, even if you change to single-user state.

  If you are using VxFS as your root file system and have the OnlineJFS product, you can extend the original root file system without unmounting, provided contiguous disk space is available.

  Otherwise, to extend the current root file system, you must create and mount *another* root disk which enables you to work with the unmounted original root disk, extending it *if* contiguous disk space is still available. If the original disk does not have contiguous disk space available, instead of expanding the original root disk, you can create a new root file system on another larger disk.

CAUTION: This procedure can fail on a VxFS file system already at 100% capacity (Error 28). You must remove some files before you attempt this operation.

Once you have the answers to these questions, follow these steos:

1. If the file system must be unmounted, unmount it.
   a. Be sure no one has files open in any file system mounted to this logical volume and that it is no user's current working directory. For example:

      `# fuser -cu /work/project5`

      If the logical volume is in use, confirm that the underlying applications no longer need it. If necessary, stop the applications.

      NOTE: If the file system is exported using NFS to other systems, verify that no one is using those other systems, then unmount it on those systems.

      ---

   b. If you cannot stop the applications using the logical volume, or it is a system directory such as `/var` or `/usr`, change to single-user state as follows:

      `# /sbin/shutdown`

   c. Unmount the file system as follows:

      `# /sbin/umount /dev/vg01/lvol2`

2. Extend the logical volume. For example:

   `# /sbin/lvextend -L 332 /dev/vg01/lvol2`

   This increases the size of this volume to 332 MB.

3. Extend the file system size to the logical volume size. If the file system is unmounted, use the `extendfs` command as follows:

   `# /sbin/extendfs /dev/vg01/rlvol2`

   If you did not have to unmount the file system, use the `fsadm` command instead. The new size is specified in terms of the block size of the file system. In this example, the block size of the file system `/work/project5` is 1 KB. To extend the file system to 332 MB, the number of blocks is 339968 (332 times 1024). For example:

   `# fsadm -b 339968 /work/project5`

4. If you unmounted the file system, mount it again.
   a. If you had to change to single-user state, reboot the system.

      `# /sbin/reboot -r`

      You can skip any additional steps to mount the file system and export it, since the boot process mounts and exports any file systems.

   b. Remount the file system as follows:

      `# /sbin/mount /dev/vg01/rlvol2 /`*`mount_point`*

      NOTE: If the file system will continue to be used by NFS clients, export it on the server (`exportfs -a`) and remount it on the clients (`mount -a`).

      ---

5. Verify that the file system reflects the expansion by entering `bdf`, `df`, or `fsadm -E`.

Reducing the Size of a File System

You might want to shrink a file system that has been allocated more disk space than it needs, allowing that disk space to be freed for other uses.

Reducing the size of a file system is more complicated than extending it. Because of file system block allocation strategies, data can be scattered throughout the logical volume. Reducing the logical volume reclaims space at the end of the logical volume, requiring file system drivers to coalesce and rearrange data blocks ahead of time. Most types of file system are unable to do such coalescence, so you must back up the data in the file system, reduce the logical volume, create a new file system in the smaller logical volume, and restore the data from your backup.

The only current file system type able to do online coalescence and size reduction is OnlineJFS, and it can fail in some cases.

Reducing a File System Created with OnlineJFS

Using the `fsadm` command shrinks the file system, provided the blocks it attempts to deallocate are not currently in use; otherwise, it fails. If sufficient free space is currently unavailable, file system defragmentation of both directories and extents might consolidate free space toward the end of the file system, allowing the contraction process to succeed when retried.

For example, suppose your VxFS file system is currently 6 GB. However, you decide you only need 2 GB with an additional 1 GB for reserve space. Thus, you want to resize the file system to 3 GB. Use `fsadm` with the `-b` option to specify the new size of the file system in sectors, then reduce the size of the logical volume to match. Assuming the file system sector size is 1K, use the following commands:

```
# fsadm -b 3145728 /home
# lvreduce -L 3072 /dev/vg01/lvol5
```

Reducing a File System Created with HFS or VxFS

To reduce the size of a file system created with HFS or VxFS, follow these steps:

1. Be sure no one has files open in any file system on the logical volume and that the logical volume is no user's current working directory. For example::

   ```
   # fuser -cu /dev/vg01/lvol5
   ```

   NOTE: If the file system is exported using NFS to other systems, verify that no one is using those other systems, then unmount the file system on those systems before unmounting it on the server.

   ---

2. Back up the data in the logical volume.

   For example, to back up `/work/project5` to the system default tape device, enter the following command:

   ```
   # tar cv /work/project5
   ```

3. Remove the data in the file system the logical volume is mounted to, as follows:

   ```
   # rm -r /work/project5
   ```

   Because `/work/project5` is a mount point, `rm -r` does not remove the directory itself.

4. Unmount the file system to which the logical volume is mounted as follows:

   ```
   # umount /work/project5
   ```

5. Reduce the size of the logical volume as follows:

   ```
   # lvreduce -L 500 /dev/vg01/lvol5
   ```

   This command reduces the logical volume`/dev/vg01/lvol5` to 500 MB.

6. Create a new file system in the reduced logical volume, using the character device file. For example:

   ```
   # newfs -f fstype /dev/vg01/rlvol5
   ```

7. Mount the logical volume as follows:

   ```
   # mount /dev/vg01/lvol5 /work/project5
   ```

8. Recover the data from the backup. For example:

   ```
   # tar xv
   ```

   This recovers all the contents of a tape in the system default drive.

9. If `/work/project5` continues to be used by NFS clients, re-export it on the server (`exportfs -a`) and remount it on the clients (`mount -a`).

Backing Up a VxFS Snapshot File System

NOTE: Creating and backing up a VxFS snapshot file system requires that you have the optional HP OnlineJFS product installed on your system. For more information, see *HP-UX System Administrator's Guide: Configuration Management*.

VxFS enables you to perform backups without taking the file system offline by making a snapshot of the file system, a read-only image of the file system at a moment in time. The primary file system remains online and continues to change. After you create the snapshot, you can back it up with any backup utility except the `dump` command.

To create and back up a VxFS snapshot file system, follow these steps:

1. Determine how large the snapshot file system must be, and create a logical volume to contain it. Use `bdf` to assess the primary file system size and consider the following:
   - Block size of the file system (1,024 bytes per block by default)
   - How much the data in this file system is likely to change (HP recommends 15 to 20% of total file system size)

   For example, to determine how large to make a snapshot of `lvol4`, mounted on the `/home` directory, examine its `bdf` output:

   ```
   # bdf /home
   filesystem          kbytes    used   avail %used mounted on
   /dev/vg00/lvol4      40960   38121    2400   94% /home
   ```

   Allowing for 20% change to this 40 MB file system, create a logical volume of eight blocks (8 MB).

2. Use `lvcreate` to create a logical volume to contain the snapshot file system. For example:

   ```
   # lvcreate -L 8 -n snap /dev/vg02
   ```

   This creates an 8 MB logical volume called `/dev/vg02/snap`, which can contain a snapshot file system of `lvol4`.

3. Create a directory for the mount point of the snapshot file system. For example:

   ```
   # mkdir /tmp/house
   ```

4. Create and mount the snapshot file system. For example:

   ```
   # mount -f vxfs -o snapof=/dev/vg00/lvol4 /dev/vg02/snap /tmp/house
   ```

   In this example, a snapshot is taken of logical volume `/dev/vg00/lvol4`, contained in logical volume `/dev/vg02/snap`, and mounted on `/tmp/house`.

5. Back up the snapshot file system with any backup utility except `dump`.

   For example, use `tar` to archive the snapshot file system `/tmp/house`, ensuring that the files on the tape have relative path names, as follows:

   ```
   # cd tmp; tar cf /dev/rtape/tape0BEST house
   ```

   Alternatively, the following `vxdump` command backs up a snapshot file system `/tmp/house`, which has extent attributes:

   ```
   # vxdump -0 -f /dev/rtape/tape0BEST /tmp/house
   ```

Administering Swap Logical Volumes

NOTE: Version 2.x volume groups do not support swap logical volumes.

When you enable a swap area within a logical volume, HP-UX determines how large the area is, and it uses no more space than that. If your disk has enough remaining contiguous space, you can increase the size of your primary swap area by using the `lvextend` command (or HP SMH) to extend the logical volume, then reboot the system. This procedure allows HP-UX to use the extra space.

If you plan device swap areas in addition to primary swap, you get the best performance when the device swap areas are on different physical volumes. This configuration allows for the interleaving of I/O to the physical volumes when swapping occurs.

To create interleaved swap, create multiple logical volumes for swap, with each logical volume on a separate disk. You must use HP-UX commands to help you obtain this configuration. HP SMH does not allow you to create a logical volume on a specific disk. See "Extending a Logical Volume to a Specific Disk" (page 51).

You can configure your swap space as described in *HP-UX System Administrator's Guide: Overview*.

NOTE: You must reboot the system for the system to recognize changes to the swap configuration.

Creating a Swap Logical Volume

To create a swap logical volume, use the `lvcreate` command. You must set a contiguous allocation policy using the `-C y` option. For example:

```
# lvcreate -C y -n swap_lvol /dev/vgnn
```

For more information, see *lvcreate*(1M).

If you create a logical volume to use as primary swap, use the `lvlnboot` command with the `-s` option to update the swap information used by LVM at boot. For example:

```
# lvlnboot -s /dev/vgnn/swap_lvol
```

Extending a Swap Device

If you are using a logical volume for swap, you must increase the logical volume size *before* increasing the swap size. You can extend the logical volume using `lvextend` or HP SMH.

Swap logical volumes must be contiguous, so extending the logical volume succeeds only if there are physical extents available at the end of the existing logical volume. If contiguous disk space is not available, create a new contiguous logical volume for primary swap within the root volume group. You do not need to designate a specific disk. For example:

```
# lvcreate -C y -L 48 -n pswap /dev/vgroot
```

After creating a logical volume to use as primary swap, use `lvlnboot` to update the boot information:

```
# lvlnboot -s /dev/vgroot/pswap
```

Reducing the Size of a Swap Device

If you are using a logical volume for swap, you must reduce the swap size *before* reducing the size of the logical volume. You can reduce the size of the logical volume using `lvreduce` or HP SMH.

Administering Dump Logical Volumes

NOTE: Version 2.x volume groups do not support dump logical volumes.

This section describes LVM information about using logical volumes as dump devices. For information on configuring and managing dump devices, see *HP-UX System Administrator's Guide: Overview*.

Creating a Dump Logical Volume

To create a dump logical volume, use the `lvcreate` command. You must set a contiguous allocation policy using the `-C y` option and disable bad block relocation using the `-r n` option. For example:

`# lvcreate -C y -r n -n dump_lvol /dev/vgnn`

For more information, see *lvcreate*(1M).

After creating a logical volume to be used as a dump device, use the `lvlnboot` command with the `-d` option to update the dump information used by LVM. If you created a logical volume `/dev/vg00/lvol2` for use as a dump area, update the boot information by entering the following:

`# lvlnboot -d /dev/vg00/lvol2`

Removing a Dump Logical Volume

To discontinue the use of a currently configured logical volume as a dump device, use the `lvrmboot` command with the `-d` option. For example:

`# lvrmboot -d /dev/vg00/lvol2`

You can then use the logical volume for other purposes. To remove the logical volume completely, see "Removing a Logical Volume" (page 53).

Hardware Issues

This section describes hardware-specific issues dealing with LVM.

Integrating Cloned LUNs

Certain disk arrays can create clones of their LUNs. For example, the HP XP product enables you to split off a set of LUNs, called Business Copies (BCs), which are copies of existing LUNs.

Cloned disks have the same information in their LVM headers as the original disks, which violates LVM's requirement that each disk have a unique identifier. To make the cloned disks usable with LVM, use the `vgchgid` command to change their volume group identifier (VGID).

All of the physical volumes to be changed must belong to the same volume group. Therefore, if you are changing multiple physical volumes, specify all of them in a single invocation of `vgchgid`. Otherwise, they are assigned different VGIDs.

For example, you have a volume group containing four physical volumes and need to create a BC for each physical volume. If you run `vgchgid` on only two BCs, `vgchgid` modifies the VGID on those two BCs. If you then run `vgchgid` again with all four BCs, `vgchgid` reports that they belong to different volume groups. To correct this, you can either run `vgchgid` on the two unmodified BCs and then use the four BCs in two separate volume groups, or you can merge back the two modified BCs and split them off again before finally running `vgchgid` with all four BCs.

After running `vgchgid` on a set of physical volumes, use `vgimport` to import them into a new volume group. For example:

1. Make BC copies and create new device files using the instructions for the array.
2. Change the VGID on the cloned disks as follows:

   `# vgchgid /dev/rdisk/disk49 /dev/rdisk/disk50`

3. If you are using an HP-UX release before March 2008, create the volume group `group` file using the procedure in "Creating the Volume Group Device File" (page 46).
4. Import the physical volumes as follows:

   `# vgimport /dev/vg04 /dev/rdisk/disk49 /dev/rdisk/disk50`

5. Back up the volume group configuration information as follows:

   `# vgcfgbackup /dev/vg04`

6. Activate the volume group as follows:

   ```
   # vgchange -a y /dev/vg04
   ```

4 Troubleshooting LVM

This chapter contains the following information:

Troubleshooting Tools Overview

This section describes the tools available for troubleshooting LVM problems.

Information Collection

You can collect information about your LVM configuration using the `vgdisplay`, `lvdisplay`, `pvdisplay`, and `lvlnboot` commands. As noted in “Planning for Recovery” (page 38), periodically collect the outputs from the commands listed in Table 4-1.

Table 4-1 LVM Information to Collect and Maintain

Command	Scope	Purpose
`ioscan -f`		Prints I/O configuration
`lvlnboot -v`		Prints information on root, boot, swap, and dump logical volumes.
`vgcfgrestore -l`	All volume groups	Prints volume group configuration from the backup file.
`vgdisplay -v`	All volume groups	Prints volume group information, including status of logical volumes and physical volumes.
`lvdisplay -v`	All logical volumes	Prints logical volume information, including mapping and status of logical extents.
`pvdisplay -v`	All physical volumes	Prints physical volume information, including status of physical extents.

In addition, use the `lvmadm` command for two purposes:

- To determine which volume group versions are supported by your release of HP-UX 11i Version 3. For example, if your release supports Version 2.1 volume groups, `lvmadm` displays the following:

```
# lvmadm -t -V 2.1
--- LVM Limits ---
VG Version                  2.1
Max VG Size (Tbytes)        2048
Max LV Size (Tbytes)        256
Max PV Size (Tbytes)        16
Max VGs                     2048
Max LVs                     2047
Max PVs                     2048
Max Mirrors                 5
Max Stripes                 511
Max Stripe Size (Kbytes)    262144
```

```
Max LXs per LV              33554432
Max PXs per PV              16777216
Max Extent Size (Mbytes)    256
```

If your release does *not* support Version 2.1 volume groups, it displays the following:

```
# lvmadm -t -V 2.1
Error: 2.1 is an invalid volume group version.
```

- To display the contents of the `/etc/lvmtab` and `/etc/lvmtab_p` files in a human-readable fashion. For example, the following command displays the contents of the LVM configuration files for all Version 1.0 volume groups on your system:

```
# lvmadm -l -V 1.0
--- Version 1.0 volume groups ---
VG Name /dev/vg00
PV Name /dev/disk/disk34_p2
```

In addition, there are some tools available from your HP support representative:

- `dump_lvmtab`: prints the contents of the `/etc/lvmtab` file in human-readable fashion.
- `vgcfgdisplay`: prints the contents of an LVM volume group configuration backup file (as created by `vgcfgbackup`), such as the volume group information, the logical volume information, physical volume information and logical extent distribution.

Consistency Checks

Most LVM commands perform consistency checking. You can inspect your LVM configuration with the `vgdisplay`, `lvdisplay`, and `pvdisplay` commands, and look for inconsistencies.

In addition, the `pvck` command performs explicit consistency checking on a physical volume. This command detects bad checksums caused by a forward system migration after a backward system migration. Run `pvck` only on deactivated volume groups. For more information, see *pvck*(1M).

NOTE: The `pvck` command does not support Version 2.x volume groups.

Maintenance Mode Boot

LVM **maintenance mode boot** is a special way to boot your system that bypasses the normal LVM structures. Use it only for problems that prevent the system from otherwise booting. It is similar to single-user state in that many of the processes that normally start do not start, and many of the system checks are not performed. LVM maintenance mode is intended to enable you to boot your system long enough to repair damage to the system LVM data structures typically using `vgcfgrestore`, which then enables you to boot your system normally.

Normally, the boot loader uses the `LABEL` file in the LIF volume to determine the location of the boot file system and the kernel `/stand/vmunix`. The `LABEL` file also contains the starting block and size of the root file system.

Under a maintenance mode boot, the boot loader attempts to find the boot file system at the start of the boot disk's user data area rather than using information from the LIF volume. To obtain the root file system's starting block and size, the boot loader reads the file `/stand/rootconf`. Since LVM is not enabled, the root file system must be allocated contiguously.

A maintenance mode boot differs from a standard boot as follows:

- The system is booted in single-user mode.
- No volume groups are activated.
- Primary swap and dump are not available.

- Only the root file system and boot file system are available.
- If the root file system is mirrored, only one copy is used. Changes to the root file system are not propagated to the mirror copies, but those mirror copies are marked stale and will be synchronized when the system boots normally.

To boot in maintenance mode on a system with a root disk configured with LVM, use the `-lm` option to the boot loader. On an HP 9000 server, enter the following command:

```
ISL> hpux -lm
```

On an HP Integrity server, enter the following command:

```
HPUX> boot -lm
```

CAUTION: When you boot your system in maintenance mode, do not activate the root volume group and do not change to multiuser mode (for example, by specifying `/sbin/init 2`). Doing so can corrupt the root file system.

When you have repaired or restored the LVM configuration information, reboot the system using the following command:

```
# /sbin/reboot
```

For more information about LVM maintenance mode boots and troubleshooting problems with LVM structures, see *Disk and File Management Tasks on HP-UX*, published by Prentice Hall PTR, 1997.

Log Files and Trace Files

LVM does not have a dedicated log file or trace file. It logs errors and warnings to `/var/adm/syslog/syslog.log`.

I/O Errors

When a device driver returns an error to LVM on an I/O request, LVM classifies the error as either **recoverable** or **nonrecoverable**.

Recoverable Errors

When LVM encounters a recoverable (correctable) error, it internally retries the failed operation assuming that the error will correct itself or that you can take steps to correct it. Examples of recoverable errors are the following:

- Device power failure
- A disk that goes missing *after the volume group is activated*
- A loose disk cable (which looks like a missing disk)

In these cases, LVM logs an error message to the console, but it does not return an error to the application accessing the logical volume.

If you have a current copy of the data on a separate, functioning mirror, then LVM directs the I/O to a mirror copy, the same as for a nonrecoverable error. Applications accessing the logical volume do not detect any error. (To preserve data synchronization between its mirrors, LVM retries recoverable write requests to a problematic disk, even if a current copy exists elsewhere. However, this process is managed by a daemon internal to LVM and has no impact on user access to the logical volume.)

However, if the device in question holds the only copy of the data, LVM retries the I/O request until it succeeds—that is, until the device responds or the system is rebooted. Any application performing I/O to the logical volume might block, waiting for the device to recover. In this case, your application or file system might appear to be stalled and might be unresponsive.

Temporarily Unavailable Device

By default, LVM retries I/O requests with recoverable errors until they succeed or the system is rebooted. Therefore, if an application or file system stalls, your troubleshooting must include checking the console log for problems with your disk drives and taking action to restore the failing devices to service.

Permanently Unavailable Device

If retrying the I/O request never succeeds (for example, the disk was physically removed), your application or file system might block indefinitely. If your application is not responding, you might need to reboot your system.

As an alternative to rebooting, you can control how long LVM retries a recoverable error before treating it as nonrecoverable by setting a timeout on the logical volume. If the device fails to respond within that time, LVM returns an I/O error to the caller. This timeout value is subject to any underlying physical volume timeout and driver timeout, so LVM can return the I/O error seconds after the logical volume timeout expired.

The timeout value is normally zero, which is interpreted as an infinite timeout. Thus, no I/O request returns to the caller until it completes successfully.

View the timeout value for a logical volume using the `lvdisplay` command, as follows:

```
# lvdisplay /dev/vg00/lvol1 | grep Timeout
IO Timeout (Seconds)        default
```

Set the timeout value using the `-t` option of the `lvchange` command. This sets the timeout value in seconds for a logical volume. For example, to set the timeout for `/dev/vg01/lvol1` to one minute, enter the following command:

```
# lvchange -t 60 /dev/vg01/lvol1
```

CAUTION: Setting a timeout on a logical volume increases the likelihood of transient errors being treated as nonrecoverable errors, so any application that reads or writes to the logical volume can experience I/O errors. If your application is not prepared to handle such errors, keep the default infinite logical volume timeout.

TIP: Set the logical volume timeout to an integral multiple of any timeout assigned to the underlying physical volumes. Otherwise, the actual duration of the I/O request can exceed the logical volume timeout. For details on how to change the I/O timeout value on a physical volume, see *pvchange*(1M).

Nonrecoverable Errors

Nonrecoverable errors are considered fatal; there is no expectation that retrying the operation will work.

If you have a current copy of the data on a separate, functioning mirror, then LVM directs reads and writes to that mirror copy. The I/O operation for the application accessing the logical volume completes successfully.

However, if you have no other copies of the data, then LVM returns an error to the subsystem accessing the logical volume. Thus, any application directly accessing a logical volume must be prepared for I/O requests to fail. File systems such as VxFS and most database applications are designed to recover from error situations; for example, if VxFS encounters an I/O error, it might disable access to a file system or a subset of the files in it.

LVM considers the following two situations nonrecoverable.

Media Errors

If an I/O request fails because of a media error, LVM typically prints a message to the console log file (`/var/adm/syslog/syslog.log`) when the error occurs. In the event of a media error, you must replace the disk (see "Replacing a Bad Disk" (page 102)).

If your disk hardware supports automatic bad block relocation (usually known as hardware sparing), enable it, because it minimizes media errors seen by LVM.

NOTE: LVM does not perform software relocation of bad blocks. It recognizes and honors software relocation entries created by previous releases of LVM but does not create new ones. Enabling or disabling bad block relocation using `lvchange` has no effect.

Missing Device When the Volume Group Was Activated

If the device associated with the I/O was not present *when the volume group was activated*, LVM prints an error message to the user's terminal at activation time. You must either locate the disk and restore it to service, or replace it, then activate the volume group again.

Volume Group Activation Failures

Normally, volume groups are automatically activated during system startup. Unless you intentionally deactivate a volume group using `vgchange`, you do not need to manually activate volume groups. In all cases, LVM requires that a quorum of disks in a volume group be available.

A *quorum* is the required number of physical volumes that must be available in a volume group to activate that volume group or for it to remain activated. To activate a volume group, *more than half* its disks that were available during the last activation must be online and in service. For the volume group to remain fully operational, *at least half* the disks must remain present and available.

During run time, when a volume group is already active, if a disk fails or is taken offline, the quorum might become lost. This condition occurs if less than half of the physical volumes defined for the volume group now remain fully operational. For example, if two disks belong to a volume group, the loss of one does not cause a loss of quorum as is the case when activating the volume group. To lose quorum, both disks must become unavailable. In that case, your volume group remains active, but a message prints to the console, indicating that the volume group has lost quorum. Until the quorum is restored (at least one of the LVM disks in the volume group in the previous example is again available), LVM does not allow you to complete most commands that affect the volume group configuration. Further, some of the I/O accesses to the logical volumes for that volume group might hang because the underlying disks are not accessible. Also, until quorum is restored, the MWC will not be updated because LVM cannot guarantee the consistency (integrity) of the LVM information.

Use the `vgchange -q n` option to override the system's quorum check when the volume group is activated. This option has no effect on the runtime quorum check. Overriding quorum can result in a volume group with an inaccurate configuration (for example, missing recently creating logical volumes). This configuration change might not be reversible.

There are ways to override quorum requirements at volume group activation time or boot time. Even when allowed by LVM, HP recommends that you do not make changes to the LVM configuration for active volume groups that do not have a quorum of disks present. To correct quorum issues, HP recommends returning the unavailable disks to service.

Quorum Problems with a Nonroot Volume Group

If you attempt to activate a nonroot volume group when not enough disks are present to establish a quorum, `vgchange` displays error messages similar to the following:

```
# vgchange -a y /dev/vg01
vgchange: Warning: Couldn't attach to the volume group
                   physical volume "/dev/dsk/c1t0d2":
```

```
The path of the physical volume refers to a device that does not exist,
      or is not configured into the kernel.
vgchange: Couldn't activate volume group "/dev/vg01":
Either no physical volumes are attached or no valid VGDAs were found on
    the physical volumes.
```

If a nonroot volume group does not activate because of a failure to meet quorum, follow these steps:

1. Check the power and data connections (including Fibre Channel zoning and security) of all the disks that are part of the volume group that you cannot activate. Return all disks (or at least enough to make a quorum) to service. Then use the `vgchange` command to activate the volume group again.
2. If there is no other way to make a quorum available, use the `-q` option with the `vgchange` command to override the quorum requirement.

   ```
   # vgchange -a y -q n /dev/vg01
   ```

   The volume group activates without a quorum. You might get messages about not being able to access certain logical volumes because part or all of a logical volume might be located on one of the disks that is not present.

   Whenever you override a quorum requirement, you run the risk of using data that is not current. Be sure to check the data on the logical volumes in the activated volume group and the size and locations of the logical volumes to ensure that they are up to date.

   Return the disabled disks to the volume group as soon as possible. When you return a disk to service that was not online when you originally activated the volume group, use the `vgchange` command as follows:

   ```
   # vgchange -a y /dev/vg01
   ```

Quorum Problems with a Root Volume Group

Your root volume group can also report a quorum problem. If not enough disks are present in the root volume group to constitute a quorum, a message indicating that not enough physical volumes are present appears during the boot sequence. This error might occur if you have physically removed a disk from your system because you no longer intended to use it, but did not remove the physical volume from the volume group using `vgreduce`. Do not remove an LVM disk from a system without first removing it from its volume group. However, you can try to recover by booting the system with the quorum override option `-lq`.

On an HP 9000 server, enter the following command:

```
ISL> hpux -lq
```

On an HP Integrity server, enter the following command:

```
HPUX> boot -lq
```

Version 2.x Volume Group Activation Failures

Version 2.x volume groups can fail to activate because of insufficient quorum. For example, `vgchange` could display error messages similar to the following:

```
# vgchange -a y /dev/vgtest
vgchange: Warning: Couldn't attach to the volume group physical
volume "/dev/disk/disk1":
I/O error
vgchange: I/O error
vgchange: Couldn't activate volume group "/dev/vgtest":
Quorum not present, or some physical volume(s) are missing.
```

If a Version 2.x volume group does not activate because of a failure to meet quorum, follow the steps described in "Quorum Problems with a Nonroot Volume Group" (page 97).

A Version 2.x volume group can also fail activation if the necessary commands or kernel drivers are not present. For example, `vgchange` could display the following error message:

```
# vgchange -a y /dev/vgtest
vgchange: Error: The "lvmp" driver is not loaded.
```

Here is another possible error message:

```
# vgchange -a y /dev/vgtest
vgchange: Warning: Couldn't attach to the volume group physical
volume "/dev/disk/disk1":
Illegal byte sequence
vgchange: Couldn't activate volume group "/dev/vgtest":
Quorum not present, or some physical volume(s) are missing.
```

A third possible error message is the following:

```
# vgchange -a y /dev/vgtest
vgchange: Warning: Couldn't attach to the volume group
physical volume "/dev/dsk/c1t0d0":
Cross-device link
vgchange: Warning: couldn't query physical volume "/dev/dsk/c1t0d0":
The specified path does not correspond to physical volume
attached to this volume group
vgchange: Couldn't query the list of physical volumes.
vgchange: Couldn't activate volume group "/dev/vgtest":
Quorum not present, or some physical volume(s) are missing.
```

To troubleshoot these activation failures, run the `lvmadm` command as follows:

```
# lvmadm -t -V 2.0
```

If the necessary commands and drivers are present, `lvmadm` displays the following:

```
--- LVM Limits ---
VG Version                    2.0
Max VG Size (Tbytes)          2048
Max LV Size (Tbytes)          256
Max PV Size (Tbytes)          16
Max VGs                       512
Max LVs                       511
Max PVs                       511
Max Mirrors                   6
Max Stripes                   511
Max Stripe Size (Kbytes)      262144
Max LXs per LV                33554432
Max PXs per PV                16777216
Max Extent Size (Mbytes)      256
```

If `lvmadm` displays no output, your operating system release does not support Version 2.x volumes. You must update your system to the March 2008 release of HP-UX 11i Version 3 or a newer release.

If the kernel driver to support Version 2.x volume groups is not loaded, `lvmadm` displays the following error:

```
# lvmadm -t -V 2.0
lvmadm: Error: The "lvmp" driver is not loaded.
```

Load the `lvmp` module using the `kcmodule` command as follows:

```
# kcmodule lvmp=best
     ==> Update the automatic 'backup' configuration first? n
       * Future operations will ask whether to update the backup.
       * The requested changes have been applied to the currently
         running configuration.
Module              State    Cause      Notes
lvmp     (before)   unused              loadable, unloadable
         (now)      loaded   best
         (next boot) loaded  explicit
```

You do not need to reboot. After you load the `lvmp` module, `lvmadm` succeeds:

```
# lvmadm -t -V 2.0
--- LVM Limits ---
VG Version                   2.0
Max VG Size (Tbytes)         2048
Max LV Size (Tbytes)         256
Max PV Size (Tbytes)         16
Max VGs                      512
Max LVs                      511
Max PVs                      511
Max Mirrors                  6
Max Stripes                  511
Max Stripe Size (Kbytes)     262144
Max LXs per LV               33554432
Max PXs per PV               16777216
Max Extent Size (Mbytes)     256
```

TIP: If your system has no Version 2.x volume groups, you can free up system resources associated with `lvmp` by unloading it from the kernel. Run the `kcmodule` command as follows:

```
# kcmodule lvmp=unused
     ==> Update the automatic 'backup' configuration first? n
       * Future operations will ask whether to update the backup.
       * The requested changes have been applied to the currently
         running configuration.
Module            State   Cause    Notes
lvmp     (before) loaded  explicit loadable, unloadable
         (now)    unused
```

If you later want to create Version 2.x volume groups, load the `lvmp` driver as described previously.

Root Volume Group Scanning

If the LVM subsystem detects that vital information is corrupted on the boot disk, it scans all the attached devices to try to find the physical volumes that are part of the root volume group. You then see the following messages on the system console and in `/var/adm/syslog/syslog.log`:

```
LVM : Failure in attaching PV (dev=0x10000nn) to the root volume group.
The physical volume does not belong to the root volume group
LVM : Failure in attaching PV (dev=0x10000nn) to the root volume group.
The physical volume does not belong to the root volume group
LVM : Activation of root volume group failed
Quorum not present, or some physical volume(s) are missing
LVM: Scanning for Root VG PVs (VGID 0xnnnnnnnn 0xnnnnnnnn)
```

If this root volume group scanning succeeds, messages similar to the following appear:

```
LVM: Rootvgscan detected 10 PV(s).  Will attempt root VG activation
using the following PV(s):
        0x100005f 0x1000060 0x1000061 0x1000062 0x1000063 0x1000064
        0x1000065 0x1000067 0x1000068 0x100006e
LVM: WARNING: Root VG activation required a scan.  The PV information in
        the on-disk BDRA may be out-of-date from the system's current IO
        configuration.  To update the on-disk BDRA, first update /etc/lvmtab
        using vgscan(1M), then update the on-disk BDRA using lvlnboot(1M).
        For example, if the root VG name is /dev/vg00:
                1. vgscan -k -f /dev/vg00
                2. lvlnboot -R /dev/vg00
LVM: Root VG activated
```

If this root volume group scan fails to find all physical volumes, the following message appears:

```
LVM: WARNING: Rootvgscan did not find any PV(s) matching root VGID.
Will attempt root VG activation using the boot device (0x10000nn).
```

Or the following message:

```
LVM: WARNING: BDRA lists the number of PV(s) for the root VG as nn,
but rootvgscan found only nn.  Proceeding with root VG activation.
```

LVM Boot Failures

There are several reasons why an LVM configuration cannot boot. In addition to the problems associated with boots from non-LVM disks, the following problems can cause an LVM-based system not to boot.

Insufficient Quorum

In this scenario, not enough disks are present in the root volume group to meet the *quorum* requirements. At boot time, a message indicating that not enough physical volumes are available appears:

```
panic: LVM: Configuration failure
```

To activate the root volume group and successfully boot the system, the number of available LVM disks must be more than half the number of LVM disks that were attached when the volume group was last active. Thus, if during the last activation there were two disks attached in the root volume group, the "more than half" requirement means that both must be available. For information on how to deal with quorum failures, see"Volume Group Activation Failures" (page 97).

Corrupted LVM Data Structures on Disk

The LVM bootable disks contain vital boot information in the BDRA. This information can become corrupted, not current, or just no longer present. Because of the importance of maintaining up-to-date information within the BDRA, use the `lvrmboot` or `lvlnboot` commands whenever you make a change that affects the location of the root, boot, primary swap, or dump logical volumes.

To correct this problem, boot the system in maintenance mode as described in "Maintenance Mode Boot" (page 94), then repair the damage to the system LVM data structures. Use `vgcfgrestore` on the boot disk.

Corrupted LVM Configuration File

Another problem with activation of a volume group is a missing or corrupted `/etc/lvmtab` or `/etc/lvmtab_p` file. After booting in maintenance mode, you can use the `vgscan` command to re-create the `/etc/lvmtab` and `/etc/lvmtab_p` files. For more information, see *vgscan*(1M).

Problems After Reducing the Size of a Logical Volume

When a file system is first created within a logical volume, it is made as large as the logical volume permits.

If you extend the logical volume without extending its file system, you can subsequently safely reduce the logical volume size, as long as it remains as big as its file system. (Use the `bdf` command to determine the size of your file system.) After you expand the file system, you can no longer safely reduce the size of the associated logical volume.

If you reduce the size of a logical volume containing a file system to a size smaller than that of a file system within it using the `lvreduce` command, you corrupt the file system. If you subsequently attempt to mount the corrupt file system, you might crash the system. If this occurs, follow these steps:

1. Reboot your system in single-user mode.
2. If you already have a good current backup of the data in the now corrupt file system, skip this step. If you do not have backup data and if that data is critical, try to recover whatever part of the data that might remain intact by attempting to back up the files on the file system.

   Before you attempt any current backup, consider the following:

   - When your backup program accesses the corrupt part of the file system, your system will crash again. You must reboot your system again to continue with the next step.
   - There is no guarantee that all (or any) of your data on that file system will be intact or recoverable. This step is an attempt to save as much as possible. That is, any data successfully backed up in this step will be recoverable, but some or all of your data might not be successfully backed up because of file corruption.
3. Immediately unmount the corrupted file system if it is mounted.
4. Use the logical volume for swap space or raw data storage, or use HP SMH or the `newfs` command to create a new file system in the logical volume. This new file system now matches the current reduced size of the logical volume.
5. If you have created a new file system on the logical volume, do one of the following:
   - If you have a good prior backup (not the backup from step 2), restore its contents. Because the new file system in the smaller logical volume is smaller than the original file system, you might not have enough space to restore all your original files.
   - If you do not have a good prior backup, attempt to restore as many files as possible from any backup you made in step 2.
   - Use the new file system for creating and storing a new set of files (not for trying to restore the original files).

Replacing a Bad Disk

Because disks are physical devices, their hardware can fail, necessitating their replacement. After a failing disk is replaced with a new one (retaining the hardware address of the original disk to avoid confusion), the data must be restored to that disk from a backup.

Since the disk was under LVM control, it can have physical extents for several logical volumes on it. The layout of those logical volumes must first be restored and the data for each of those logical volumes restored from backup.

This section provides a step-by-step guide to replacing a faulty LVM disk.

Review "Preparing for LVM System Recovery" (page 39) for steps to perform before a disk fails. Read this section carefully, and implement the required procedures as soon as possible. Your system recovery might rely on these steps.

If you have any questions about the recovery process, contact your local HP Customer Response Center for assistance.

TIP: For an in-depth discussion of disk failures, see the white paper *When Good Disks Go Bad: Dealing with Disk Failures under LVM*, available at http://docs.hp.com. It covers additional topics such as recognizing a disk failure, identifying the failing disk, and choosing the appropriate resolution, such as removing the disk instead of replacing it. The paper also covers releases prior to HP-UX 11i Version 3.

Disk Replacement Prerequisites

Once you have isolated a failed disk, the replacement process depends on answers to the following questions:

- **Is the disk hot-swappable?**

You can remove or add an inactive hot-swappable hard disk drive module to a system while power is still on and the SCSI bus is still active. That is, you can replace or remove a hot-swappable disk from a system without turning off the power to the entire system.

Consult your system hardware manuals for information about which disks in your system are hot-swappable. Specifications for other hard disks are available in their installation manuals at http://docs.hp.com.

If your disk is not hot-swappable, you must schedule system down time to replace the disk.

- **Is the disk the root disk or part of the root volume group?**

  If the root disk is failing, the replacement process includes steps to set up the boot area; in addition, you might have to boot from its mirror if the primary root disk has failed. If a failing root disk is not mirrored, you must reinstall to the replacement disk or recover it from an Ignite-UX backup.

  To determine whether the disk is in the root volume group, use the `lvlnboot` command with the `–v` option. It lists the disks in the root volume group, and any special volumes configured on them. For example:

```
# lvlnboot –v
Boot Definitions for Volume Group /dev/vg00:
Physical Volumes belonging in Root Volume Group:
        /dev/disk/disk47_p2 -- Boot Disk
Boot: lvol1     on:     /dev/disk/disk47_p2
Root: lvol3     on:     /dev/disk/disk47_p2
Swap: lvol2     on:     /dev/disk/disk47_p2
Dump: lvol2     on:     /dev/disk/disk47_p2, 0
```

- **What logical volumes are on the disk, and are they mirrored?**

  After you replace the disk, you might need to restore data from backups. However, you must only recover data for a subset of the logical volumes in the volume group. Only the logical volumes that actually have physical extents on the disk are affected. In addition, if a logical volume is mirrored, there is probably a current copy of the data on the mirror, so it does not need to be recovered from backup.

  You can find the list of logical volumes using the disk with the `pvdisplay` command. With the `–v` option, `pvdisplay` shows a listing of all the physical extents on a physical volume and to what logical volume they belong. This list is long; pipe it to `more` or send it to a file. For example:

```
# pvdisplay -v /dev/disk/disk3 | more
...
   --- Distribution of physical volume ---
   LV Name            LE of LV  PE for LV
   /dev/vg00/lvol5    50        50
   /dev/vg00/lvol6    245       245
 ...
```

  In this example, logical volumes `/dev/vg00/lvol5` and `/dev/vg00/lvol6` have physical extents on this disk, so you must restore `lvol5` and `lvol6` only.

  If `pvdisplay` fails, you can see any configuration documentation you created in advance, or use the `vgcfgdisplay` command, available from your HP support representative.

  For each of the logical volumes affected, use `lvdisplay` to determine if the number of mirror copies is greater than zero. This verifies that the logical volume is mirrored. For example:

```
# lvdisplay /dev/vg00/lvol1
--- Logical volumes ---
LV Name                 /dev/vg00/lvol1
VG Name                 /dev/vg00
LV Permission           read/write
LV Status               available/syncd
```

```
Mirror copies              1
Consistency Recovery       MWC
Schedule                   parallel
LV Size (Mbytes)           300
Current LE                 75
Allocated PE               150
Stripes                    0
Stripe Size (Kbytes)       0
Bad block                  off
Allocation                 strict/contiguous
IO Timeout (Seconds)       default
```

The number of mirror copies is not zero; therefore, the logical volume is mirrored.

Use `lvdisplay` again to determine which logical extents are mapped onto the suspect disk, and whether there is a current copy of that data *on another disk*. With the `-v` option, `lvdisplay` shows every logical extent, its mapping to any physical extents, and the status of those physical extents (stale or current).

This listing can be quite long, so use `grep` to confine the listing to the disk that is being replaced. For example:

```
# lvdisplay -v /dev/vg00/lvol1 | grep -e /dev/disk/disk3 -e '???'
   00000 /dev/disk/disk3  00000 current  /dev/disk/disk6  00000 current
   00001 /dev/disk/disk3  00001 current  /dev/disk/disk6  00001 current
   00002 /dev/disk/disk3  00002 current  /dev/disk/disk6  00002 current
   00003 /dev/disk/disk3  00003 current  /dev/disk/disk6  00003 current
   00004 /dev/disk/disk3  00004 current  /dev/disk/disk6  00004 current
   00005 /dev/disk/disk3  00005 current  /dev/disk/disk6  00005 current
...
```

In this example, all `lvol1` physical extents on `/dev/disk/disk3` have a current copy elsewhere on the system, specifically on `/dev/disk/disk6`. If `/dev/disk/disk3` was unavailable when the volume group was activated, its column contains a '`???`' instead of the disk name.

Based on the gathered information, choose the appropriate procedure.

Replacing a Mirrored Nonboot Disk

Use this procedure if *all* the physical extents on the disk have copies on another disk, and the disk is *not* a boot disk. If the disk contains any unmirrored logical volumes or any mirrored logical volumes without an available and current mirror copy, see "Replacing an Unmirrored Nonboot Disk" (page 106).

For this example, the disk to be replaced is at lunpath hardware path 0/1/1/1.0x3.0x0, with device special files named `/dev/disk/disk14` and `/dev/rdisk/disk14`. Follow these steps:

1. Save the hardware paths to the disk.

   Run the `ioscan` command and note the hardware paths of the failed disk.

   ```
   # ioscan -m lun /dev/disk/disk14
   Class I Lun H/W Path      Driver  S/W State H/W Type Health  Description
   ========================================================================
   disk 14 64000/0xfa00/0x0  esdisk  CLAIMED   DEVICE   offline HP MSA Vol
           0/1/1/1.0x3.0x0
                     /dev/disk/disk14      /dev/rdisk/disk14
   ```

   In this example, the LUN instance number is 14, the LUN hardware path is 64000/0xfa00/0x0, and the lunpath hardware path is 0/1/1/1.0x3.0x0.

   When the failed disk is replaced, a new LUN instance and LUN hardware path are created. To identify the disk after it is replaced, you must use the lunpath hardware path (0/1/1/1.0x3.0x0).

2. Halt LVM access to the disk.

   If the disk is not hot-swappable, power off the system to replace it. By shutting down the system, you halt LVM access to the disk, so you can skip this step.

   If the disk is hot-swappable, detach it using the `-a` option of the `pvchange` command:

   ```
   # pvchange -a N /dev/disk/disk14
   ```

3. Replace the disk.

   For the hardware details on how to replace the disk, see the hardware administrator's guide for the system or disk array.

   If the disk is hot-swappable, replace it.

   If the disk is not hot-swappable, shut down the system, turn off the power, and replace the disk. Reboot the system.

4. Notify the mass storage subsystem that the disk has been replaced.

   If the system was not rebooted to replace the failed disk, then run `scsimgr` before using the new disk as a replacement for the old disk. For example:

   ```
   # scsimgr replace_wwid -D /dev/rdisk/disk14
   ```

   This command allows the storage subsystem to replace the old disk's LUN World-Wide-Identifier (WWID) with the new disk's LUN WWID. The storage subsystem creates a new LUN instance and new device special files for the replacement disk.

5. Determine the new LUN instance number for the replacement disk.

   For example:

   ```
   # ioscan -m lun
   Class I Lun H/W Path      Driver  S/W State H/W Type Health  Description
   ========================================================================
   disk 14 64000/0xfa00/0x0  esdisk  NO_HW     DEVICE   offline HP MSA Vol
                     /dev/disk/disk14      /dev/rdisk/disk14
   ...
   disk 28 64000/0xfa00/0x1c esdisk  CLAIMED   DEVICE   online  HP MSA Vol
           0/1/1/1.0x3.0x0
                     /dev/disk/disk28      /dev/rdisk/disk28
   ```

   In this example, LUN instance 28 was created for the new disk, with LUN hardware path 64000/0xfa00/0x1c, device special files `/dev/disk/disk28` and `/dev/rdisk/disk28`, at the same lunpath hardware path as the old disk, 0/1/1/1.0x3.0x0. The old LUN instance 14 for the old disk now has no lunpath associated with it.

NOTE: If the system was rebooted to replace the failed disk, then `ioscan -m lun` does not display the old disk.

6. Assign the old instance number to the replacement disk.

   For example:

   ```
   # io_redirect_dsf -d /dev/disk/disk14 -n /dev/disk/disk28
   ```

   This assigns the old LUN instance number (14) to the replacement disk. In addition, the device special files for the new disk are renamed to be consistent with the old LUN instance number. The following `ioscan -m lun` output shows the result:

   ```
   # ioscan -m lun /dev/disk/disk14
   Class I Lun H/W Path      Driver  S/W State H/W Type Health  Description
   ========================================================================
   disk 14 64000/0xfa00/0x1c esdisk  CLAIMED   DEVICE   online  HP MSA Vol
           0/1/1/1.0x3.0x0
                     /dev/disk/disk14      /dev/rdisk/disk14
   ```

   The LUN representation of the old disk with LUN hardware path 64000/0xfa00/0x0 was removed. The LUN representation of the new disk with LUN hardware path

64000/0xfa00/0x1c was reassigned from LUN instance 28 to LUN instance 14 and its device special files were renamed as `/dev/disk/disk14` and `/dev/rdisk/disk14`.

7. Restore LVM configuration information to the new disk.

   For example:

   ```
   # vgcfgrestore -n /dev/vgnn /dev/rdisk/disk14
   ```

8. Restore LVM access to the disk.

   If you did *not* reboot the system in Step 2, reattach the disk as follows:

   ```
   # pvchange -a y /dev/disk/disk14
   ```

   If you did reboot the system, reattach the disk by reactivating the volume group as follows:

   ```
   # vgchange -a y /dev/vgnn
   ```

   NOTE: The `vgchange` command with the `-a y` option can be run on a volume group that is deactivated or already activated. It attaches all paths for all disks in the volume group and resumes automatically recovering any disks in the volume group that had been offline or any disks in the volume group that were replaced. Therefore, run `vgchange` only after all work has been completed on all disks and paths in the volume group, and it is necessary to attach them all.

Because all the data on the replaced disk was mirrored, you do not need to do anything else; LVM automatically synchronizes the data on the disk with the other mirror copies of the data.

Replacing an Unmirrored Nonboot Disk

Use this procedure if *any* of the physical extents on the disk do not have mirror copies elsewhere, and your disk is not a boot disk.

In this example, the disk to be replaced is at lunpath hardware path 0/1/1/1.0x3.0x0, with device special files named `/dev/disk/disk14` and `/dev/rdisk/disk14`. Follow these steps:

1. Save the hardware paths to the disk.

   Run the `ioscan` command and note the hardware paths of the failed disk.

   ```
   # ioscan -m lun /dev/disk/disk14
   Class I Lun H/W Path      Driver  S/W State H/W Type Health  Description
   ========================================================================
   disk 14 64000/0xfa00/0x0  esdisk  CLAIMED   DEVICE   offline HP MSA Vol
           0/1/1/1.0x3.0x0
                     /dev/disk/disk14      /dev/rdisk/disk14
   ```

   In this example, the LUN instance number is 14, the LUN hardware path is 64000/0xfa00/0x0, and the lunpath hardware path is 0/1/1/1.0x3.0x0.

   When the failed disk is replaced, a new LUN instance and LUN hardware path are created. To identify the disk after it is replaced, you must use the lunpath hardware path (0/1/1/1.0x3.0x0).

2. Halt LVM access to the disk.

   If the disk is not hot-swappable, power off the system to replace it. By shutting down the system, you halt LVM access to the disk, so you can skip this step.

   If the disk is hot-swappable, disable user and LVM access to all unmirrored logical volumes.

   First, disable *user* access to all unmirrored logical volumes. Halt any applications and unmount any file systems using these logical volumes. This prevents the applications or file systems from writing inconsistent data over the newly restored replacement disk.

   For each unmirrored logical volume using the disk:

a. Use the `fuser` command to make sure no one is accessing the logical volume, either as a raw device or as a file system. If users have files open in the file system or it is their current working directory, `fuser` reports their process IDs.

   For example, if the logical volume was `/dev/vg01/lvol1`, enter the following command:

```
# fuser -cu dev/vg01/lvol1
/dev/vg01/lvol1:    27815c(root)   27184c(root)
```

b. If `fuser` reports process IDs using the logical volume, use the `ps` command to map the list of process IDs to processes, and then determine whether you can halt those processes.

   For example, look up processes 27815 and 27184 as follows:

```
# ps -fp27815 -p27184
     UID   PID  PPID  C    STIME TTY       TIME COMMAND
    root 27815 27184  0 09:04:05 pts/0     0:00 vi test.c
    root 27184 27182  0 08:26:24 pts/0     0:00 -sh
```

c. If so, use `fuser` with the `–k` option to kill all processes accessing the logical volume.

   The example processes are noncritical, so kill them as follows:

```
# fuser -ku dev/vg01/lvol1
/dev/vg01/lvol1:    27815c(root)   27184c(root)
```

d. If the logical volume is being used as a file system, unmount it as follows:

```
# umount /dev/vg01/lvol1
```

NOTE: If you cannot stop the applications using the logical volume, or you cannot unmount the file system, you must shut down the system.

After disabling user access to the unmirrored logical volumes, disable LVM access to the disk:

```
# pvchange -a N /dev/disk/disk14
```

3. Replace the disk.

   For the hardware details on how to replace the disk, see the hardware administrator's guide for the system or disk array.

   If the disk is hot-swappable, replace it.

   If the disk is not hot-swappable, shut down the system, turn off the power, and replace the disk. Reboot the system.

4. Notify the mass storage subsystem that the disk has been replaced.

   If the system was not rebooted to replace the failed disk, then run `scsimgr` before using the new disk as a replacement for the old disk. For example:

```
# scsimgr replace_wwid –D /dev/rdisk/disk14
```

   This command allows the storage subsystem to replace the old disk's LUN World-Wide-Identifier (WWID) with the new disk's LUN WWID. The storage subsystem creates a new LUN instance and new device special files for the replacement disk.

5. Determine the new LUN instance number for the replacement disk.

   For example:

```
# ioscan -m lun
Class I Lun H/W Path      Driver  S/W State H/W Type Health  Description
========================================================================
disk 14 64000/0xfa00/0x0  esdisk  NO_HW     DEVICE   offline HP MSA Vol
                  /dev/disk/disk14      /dev/rdisk/disk14
...
disk 28 64000/0xfa00/0x1c esdisk  CLAIMED   DEVICE   online  HP MSA Vol
```

```
        0/1/1/1.0x3.0x0
                  /dev/disk/disk28      /dev/rdisk/disk28
```

In this example, LUN instance 28 was created for the new disk, with LUN hardware path 64000/0xfa00/0x1c, device special files `/dev/disk/disk28` and `/dev/rdisk/disk28`, at the same lunpath hardware path as the old disk, 0/1/1/1.0x3.0x0. The old LUN instance 14 for the old disk now has no lunpath associated with it.

NOTE: If the system was rebooted to replace the failed disk, then `ioscan -m lun` does not display the old disk.

---

6. Assign the old instance number to the replacement disk.

   For example:

   ```
   # io_redirect_dsf -d /dev/disk/disk14 -n /dev/disk/disk28
   ```

   This assigns the old LUN instance number (14) to the replacement disk. In addition, the device special files for the new disk are renamed to be consistent with the old LUN instance number. The following `ioscan -m lun` output shows the result:

   ```
   # ioscan -m lun /dev/disk/disk14
   Class I Lun H/W Path      Driver  S/W State H/W Type Health  Description
   ========================================================================
   disk 14 64000/0xfa00/0x1c esdisk  CLAIMED   DEVICE   online  HP MSA Vol
           0/1/1/1.0x3.0x0
                     /dev/disk/disk14      /dev/rdisk/disk14
   ```

   The LUN representation of the old disk with LUN hardware path 64000/0xfa00/0x0 was removed. The LUN representation of the new disk with LUN hardware path 64000/0xfa00/0x1c was reassigned from LUN instance 28 to LUN instance 14 and its device special files were renamed as `/dev/disk/disk14` and `/dev/rdisk/disk14`.

7. Restore LVM configuration information to the new disk.

   For example:

   ```
   # vgcfgrestore -n /dev/vgnn /dev/rdisk/disk14
   ```

8. Restore LVM access to the disk.

   If you did *not* reboot the system in Step 2, reattach the disk as follows:

   ```
   # pvchange -a y /dev/disk/disk14
   ```

   If you did reboot the system, reattach the disk by reactivating the volume group as follows:

   ```
   # vgchange -a y /dev/vgnn
   ```

NOTE: The `vgchange` command with the `-a y` option can be run on a volume group that is deactivated or already activated. It attaches all paths for all disks in the volume group and resumes automatically recovering any disks in the volume group that had been offline or any disks in the volume group that were replaced. Therefore, run `vgchange` only after all work has been completed on all disks and paths in the volume group, and it is necessary to attach them all.

9. Recover any lost data.

   LVM recovers all the mirrored logical volumes on the disk, and starts that recovery when the volume group is activated.

   For all the unmirrored logical volumes that you identified in Step 2, restore the data from backup and reenable user access as follows:

   - For raw volumes, restore the full raw volume using the utility that was used to create your backup. Then restart the application.
   - For file systems, you must re-create the file systems first. For example:

     ```
     # newfs -F fstype /dev/vgnn/rlvolnn
     ```

     Use the logical volume's character device file for the `newfs` command. For file systems that had nondefault configurations, see *newfs*(1M) for the correct options.

     After creating the file system, mount it under the mount point that it previously occupied. Then restore the data for that file system from your full backups.

TIP: To make the file system re-creation step easier, record how they were originally created. You can change other file system parameters, such as those used to tune file system performance. The file system must be at least as large as before the disk failure.

Replacing a Mirrored Boot Disk

There are two additional operations you must perform when replacing a mirrored boot disk:

1. You must initialize boot information on the replacement disk.
2. If the replacement requires rebooting the system, and the primary boot disk is being replaced, you must boot from the alternate boot disk.

In this example, the disk to be replaced is at lunpath hardware path 0/1/1/1.0x3.0x0, with device special files named `/dev/disk/disk14` and `/dev/rdisk/disk14`. The system is an HP Integrity server, so the physical volume names must specify the HP-UX partition on the boot disk (`/dev/disk/disk14_p2` and `/dev/disk/disk14_p2`).

1. Save the hardware paths to the disk.

   Run the `ioscan` command and note the hardware paths of the failed disk as follows:

   ```
   # ioscan -m lun /dev/disk/disk14
   Class I Lun H/W Path      Driver  S/W State H/W Type Health  Description
   ========================================================================
   disk 14 64000/0xfa00/0x0  esdisk  CLAIMED   DEVICE   offline HP MSA Vol
           0/1/1/1.0x3.0x0
                     /dev/disk/disk14      /dev/rdisk/disk14
                     /dev/disk/disk14_p1   /dev/rdisk/disk14_p1
                     /dev/disk/disk14_p2   /dev/rdisk/disk14_p2
                     /dev/disk/disk14_p3   /dev/rdisk/disk14_p3
   ```

   In this example, the LUN instance number is 14, the LUN hardware path is 64000/0xfa00/0x0, and the lunpath hardware path is 0/1/1/1.0x3.0x0.

   When the failed disk is replaced, a new LUN instance and LUN hardware path are created. To identify the disk after it is replaced, you must use the lunpath hardware path (0/1/1/1.0x3.0x0).

2. Halt LVM access to the disk.

   If the disk is not hot-swappable, power off the system to replace it. By shutting down the system, you halt LVM access to the disk, so you can skip this step.

   If the disk is hot-swappable, detach the device using the `–a` option of the `pvchange` command:

   ```
   # pvchange -a N /dev/disk/disk14_p2
   ```

   NOTE: On an HP 9000 server, the boot disk is not partitioned so the physical volume refers to the entire disk, not the HP-UX partition. Use the following command:

   ```
   # pvchange -a N /dev/disk/disk14
   ```

3. Replace the disk.

   For the hardware details on how to replace the disk, see the hardware administrator’s guide for the system or disk array.

   If the disk is hot-swappable, replace it.

If the disk is not hot-swappable, shut down the system, turn off the power, and replace the disk. Reboot the system. Two problems can occur:

- If you replaced the disk that you normally boot from, the replacement disk does not contain the information needed by the boot loader. In this case, interrupt the boot process and boot from the mirror boot disk, which is configured as the alternate boot path.
- If there are only two disks in the root volume group, the system probably fails its quorum check as described in "Volume Group Activation Failures" (page 97). It can panic early in the boot process with the message:

  ```
  panic: LVM: Configuration failure
  ```

  In this situation, you must override quorum to boot successfully. Do this by interrupting the boot process and adding the `-lq` option to the boot command.

For information on the boot process and how to select boot options, see *HP-UX System Administrator's Guide: Configuration Management*.

4. Notify the mass storage subsystem that the disk has been replaced.

   If the system was not rebooted to replace the failed disk, then run `scsimgr` before using the new disk as a replacement for the old disk. For example:

   ```
   # scsimgr replace_wwid -D /dev/rdisk/disk14
   ```

   This command allows the storage subsystem to replace the old disk's LUN World-Wide-Identifier (WWID) with the new disk's LUN WWID. The storage subsystem creates a new LUN instance and new device special files for the replacement disk.

5. Determine the new LUN instance number for the replacement disk.

   For example:

   ```
   # ioscan -m lun
   Class I Lun H/W Path      Driver  S/W State H/W Type Health  Description
   ========================================================================
   disk 14 64000/0xfa00/0x0  esdisk  NO_HW     DEVICE   offline HP MSA Vol
                     /dev/disk/disk14      /dev/rdisk/disk14
                     /dev/disk/disk14_p1   /dev/rdisk/disk14_p1
                     /dev/disk/disk14_p2   /dev/rdisk/disk14_p2
                     /dev/disk/disk14_p3   /dev/rdisk/disk14_p3
   ...
   disk 28 64000/0xfa00/0x1c esdisk  CLAIMED   DEVICE   online  HP MSA Vol
           0/1/1/1.0x3.0x0
                     /dev/disk/disk28      /dev/rdisk/disk28
   ```

   In this example, LUN instance 28 was created for the new disk, with LUN hardware path 64000/0xfa00/0x1c, device special files `/dev/disk/disk28` and `/dev/rdisk/disk28`, at the same lunpath hardware path as the old disk, 0/1/1/1.0x3.0x0. The old LUN instance 14 for the old disk now has no lunpath associated with it.

NOTE: If the system was rebooted to replace the failed disk, then `ioscan -m lun` does not display the old disk.

6. (HP Integrity servers only) Partition the replacement disk.

   Partition the disk using the `idisk` command and a partition description file, and create the partition device files using `insf`, as described in "Mirroring the Boot Disk on HP Integrity Servers" (page 81).

7. Assign the old instance number to the replacement disk.

   For example:

   ```
   # io_redirect_dsf -d /dev/disk/disk14 -n /dev/disk/disk28
   ```

   This assigns the old LUN instance number (14) to the replacement disk. In addition, the device special files for the new disk are renamed to be consistent with the old LUN instance number. The following `ioscan -m lun` output shows the result:

```
# ioscan -m lun /dev/disk/disk14
Class I Lun H/W Path      Driver  S/W State H/W Type Health  Description
========================================================================
disk 14 64000/0xfa00/0x1c esdisk  CLAIMED   DEVICE   online  HP MSA Vol
        0/1/1/1.0x3.0x0
                  /dev/disk/disk14      /dev/rdisk/disk14
                  /dev/disk/disk14_p1   /dev/rdisk/disk14_p1
                  /dev/disk/disk14_p2   /dev/rdisk/disk14_p2
                  /dev/disk/disk14_p3   /dev/rdisk/disk14_p3
```

The LUN representation of the old disk with LUN hardware path 64000/0xfa00/0x0 was removed. The LUN representation of the new disk with LUN hardware path 64000/0xfa00/0x1c was reassigned from LUN instance 28 to LUN instance 14 and its device special files were renamed as `/dev/disk/disk14` and `/dev/rdisk/disk14`.

8. Restore LVM configuration information to the new disk.

   For example:

   `# vgcfgrestore -n /dev/vg00 /dev/rdisk/disk14_p2`

NOTE: On an HP 9000 server, the boot disk is not partitioned, so the physical volume refers to the entire disk, not the HP-UX partition. Use the following command:

`# vgcfgrestore -n /dev/vg00 /dev/rdisk/disk14`

---

9. Restore LVM access to the disk.

   If you did *not* reboot the system in Step 2, reattach the disk as follows:

   `# pvchange -a y /dev/disk/disk14_p2`

   On an HP 9000 server, use this command:

   `# pvchange -a y /dev/disk/disk14`

   If you did reboot the system, reattach the disk by reactivating the volume group as follows:

   `# vgchange -a y /dev/vg00`

NOTE: The `vgchange` command with the `-a y` option can be run on a volume group that is deactivated or already activated. It attaches all paths for all disks in the volume group and resumes automatically recovering any disks in the volume group that had been offline or any disks in the volume group that were replaced. Therefore, run `vgchange` only after all work has been completed on all disks and paths in the volume group, and it is necessary to attach them all.

---

10. Initialize boot information on the disk.

    For an HP Integrity server, set up the boot area and update the autoboot file in the disk's EFI partition as described in step 5 and step 6 of “Mirroring the Boot Disk on HP Integrity Servers” (page 81).

    For an HP 9000 server, set up the boot area and update the autoboot file as described in step 4 and step 5 of “Mirroring the Boot Disk on HP 9000 Servers” (page 79).

Replacing an Unmirrored Boot Disk

With the failure of an unmirrored boot disk, you have lost the only copy of information that is required to boot the system. You must reinstall to the replacement disk, or recover it from an Ignite-UX backup.

Warning and Error Messages

This section lists some of the warning and error messages reported by LVM. For each message, the cause is described and an action is recommended.

Matching Error Messages to Physical Disks and Volume Groups

Often an error message contains the device number for a device, rather than the device file name. For example, you might see the following message in /var/adm/syslog/syslog.log:

```
SCSI: Request Timeout -- lbolt: 329741615, dev: 1f022000
```

To map this error message to a specific disk, search under the /dev directory for a device file with a device number that matches the printed value. That is, search for a file with a minor number matching the lower six digits of the number following dev:. The device number in this example is 1f022000; its lower six digits are 022000, so search for that value using the following command:

```
# ll /dev/*dsk | grep 022000
brw-r-----   1 bin        sys         31 0x022000 Sep 22  2002 c2t2d0
crw-r-----   1 bin        sys        188 0x022000 Sep 25  2002 c2t2d0
```

Use the pvdisplay command to determine which volume group contains this physical volume as follows:

```
# pvdisplay /dev/dsk/c2t2d0 | grep "VG Name"
VG Name                    /dev/vg03
```

If the pvdisplay command fails, search for the physical volume in the LVM configuration files /etc/lvmtab and /etc/lvmtab_p as follows:

```
# lvmadm -l -V 1.0
--- Version 1.0 volume groups ---
VG Name /dev/vg00
PV Name /dev/disk/disk36_p2

VG Name /dev/vgtest
PV Name /dev/disk/disk43
PV Name /dev/disk/disk44

VG Name /dev/vg03
PV Name /dev/dsk/c2t2d0
```

If your version of lvmadm does not recognize the -l option, use the strings command as follows:

```
# strings /etc/lvmtab | more
/dev/vg00
/dev/disk/disk36_p2
/dev/vgtest
/dev/disk/disk43
/dev/disk/disk44
/dev/vg03
/dev/dsk/c2t2d0
```

Based on the output of these commands, the error message refers to physical volume /dev/dsk/c2t2d0, which belongs to volume group vg03.

Similarly, some LVM error messages in /var/adm/syslog/syslog.log contain the device number for a volume group. For example:

```
LVM: VG 128 0x002000: Lost quorum.
```

The major number 128 indicates that this is a Version 2.x volume group. A Version 1.0 volume group has a major number of 64. To map this error message to a volume group, search under the /dev directory for a volume group device file with a device number that matches the major and minor numbers. In this example, the major number is 128 and the minor number is 0x002000, so search for those value using the following command:

```
 # ll /dev/*/group | grep 128.0x002000
  crw-r-----  1 root  sys   128 0x002000 Jan  7 08:27 /dev/vgtest2/group
```

The example error message refers to the Version 2.x volume group vgtest.

Messages For All LVM Commands

Message Text

```
vgcfgbackup: /etc/lvmtab is out of date with the running kernel:
Kernel indicates # disks for "/dev/vgname"; /etc/lvmtab has # disks.
Cannot proceed with backup.
```

Cause

The number of current physical volumes (`Cur PV`) and active physical volumes (`Act PV`) are not the same. `Cur PV` and `Act PV` must always agree for the volume group. This error also indicates that `/etc/lvmtab` or `/etc/lvmtab_p`, which is used to match physical volumes to a volume group, is out of date with the LVM data structures in memory and on disk.

Recommended Action

Try to locate any missing disks. For each of the disk in the volume group, use `ioscan` and `diskinfo` to confirm that the disk is functioning properly.

lvchange(1M)

Message Text

```
"m": Illegal option.
```

Cause

The system does not have HP MirrorDisk/UX installed.

Recommended Action

Install HP MirrorDisk/UX.

lvextend(1M)

Message Text

```
lvextend: Not enough free physical extents available.
Logical volume "/dev/vgname/lvname" could not be extended.
Failure possibly caused by strict allocation policy
```

Cause

There is not enough space in the volume group to extend the logical volume to the requested size. This is typically caused by one of the following situations:

1. There are not enough free physical extents in the volume group. Run `vgdisplay` to confirm the number of available physical extents, and multiply that number by the extent size to determine the free space in the volume group. For example:

```
# vgdisplay vg00
--- Volume groups ---
VG Name                     /dev/vg00
VG Write Access             read/write
VG Status                   available
Max LV                      255
Cur LV                      10
Open LV                     10
Max PV                      16
Cur PV                      1
Act PV                      1
Max PE per PV               4350
VGDA                        2
PE Size (Mbytes)            4
```

```
Total PE                     4340
Alloc PE                     3740
Free PE                      600
Total PVG                    0
Total Spare PVs              0
Total Spare PVs in use       0
VG Version                   1.0
VG Max Size                  1082g
VG Max Extents               69248
```

In this example, the total free space is 600 physical extents of 4 MB, or 2400 MB.

2. The logical volume is mirrored with a strict allocation policy, and there are not enough extents on a separate disk to comply with the allocation policy. To confirm this, run `lvdisplay` to determine which disks the logical volume occupies, and then check whether there is sufficient space on the other disks in the volume group.
3. In a SAN environment, one of the disks was dynamically increased in size. LVM did not detect the asynchronous change in size.

Recommended Action

1. Choose a smaller size for the logical volume, or add more disk space to the volume group.
2. Choose a smaller size for the logical volume, or add more disk space to the volume group. Alternatively, free up space on an available disk using `pvmove`.
3. Use the `vgmodify` command to detect the disk size change and incorporate the new space into the volume group.

Message Text

```
"m": Illegal option.
```

Cause

The system does not have HP MirrorDisk/UX installed.

Recommended Action

Install HP MirrorDisk/UX.

lvlnboot(1M)

Message Text

```
lvlnboot: Unable to configure swap logical volume.
Swap logical volume size beyond the IODC max address.
```

Cause

The boot disk firmware cannot access the entire range of the swap logical volume. This happens with older host bus adapters when primary swap is configured past 4 GB on the disk.

Recommended Action

Upgrade the system firmware or use a newer host bus adapter that supports block addressing. If neither of these actions succeeds, reduce the size of the primary swap logical volume so that it does not exceed 4 GB.

pvchange(1M)

Message Text

```
Unable to detach the path or physical volume via the pathname provided.
Either use pvchange(1M) -a N to detach the PV using an attached path
or detach each path to the PV individually using pvchange(1M) –a n
```

Cause

The specified path is not part of any volume group, because the path has not been successfully attached to the otherwise active volume group it belongs to.

Recommended Action

Check the specified path name to make sure it is correct. If the error occurred while detaching a physical volume, specify a different path. If it is not clear whether any path was attached before, individually detach each path to the physical volume using `pvchange` with the *–a n* option.

Message Text

```
Warning: Detaching a physical volume reduces the availability
of data within the logical volumes residing on that disk.
Prior to detaching a physical volume or the last available path to it,
verify that there are alternate copies of the data available on other
disks in the volume group. If necessary, use pvchange(1M) to reverse
this operation.
```

Cause

This warning is advisory. It is generated whenever a path or physical volume is detached.

Recommended Action

None.

vgcfgbackup(1M)

Message Text

`Invalid LVMREC on Physical Volume` *`pvname`*

Cause

The LVM header on the disk is incorrect. This can happen when an existing LVM disk is overwritten with a command like `dd` or `pvcreate`. If the disk is shared between two systems, one system might not be aware that the disk is already in a volume group. The corruption can also be caused by running `vgchgid` incorrectly when using BC split volumes.

Recommended Action

Restore a known good configuration to the disk using `vgcfgrestore`. Be sure to use a valid copy dated before the first occurrence of the problem. For example:

```
# vgcfgrestore –n vgname pvname
```

vgcfgrestore(1M)

Message Text

```
Cannot restore Physical Volume pvname
Detach the PV or deactivate the VG, before restoring the PV.
```

Cause

The `vgcfgrestore` command was used to initialize a disk that already belongs to an active volume group.

Recommended Action

Detach the physical volume or deactivate the volume group before attempting to restore the physical volume. If the disk may be corrupted, detach the disk and mark it using `vgcfgrestore`, then attach it again without replacing the disk. This causes LVM to reinitialize the disk and synchronize any mirrored user data mapped there.

vgchange(1M)

Message Text

```
vgchange: WARNING: The "lvmp" driver is not loaded.
```

Cause

You are activating a Version 2.x volume group, and the kernel driver for Version 2.x volume groups is not loaded.

Recommended Action

Load the `lvmp` driver, as follows:

```
# kcmodule lvmp=best
     ==> Update the automatic 'backup' configuration first? n
       * Future operations will ask whether to update the backup.
       * The requested changes have been applied to the currently
         running configuration.
Module              State   Cause     Notes
lvmp    (before)    unused            loadable, unloadable
        (now)       loaded  best
        (next boot) loaded  explicit
```

Message Text

```
vgchange: Warning: Couldn't attach to the volume group
physical volume "pvname":
Illegal byte sequence
vgchange: Couldn't activate volume group "vgname":
Quorum not present, or some physical volume(s) are missing.
```

Cause

You are activating a Version 2.x volume group, and your operating system release does not support Version 2.x volumes.

Recommended Action

Update your system to the March 2008 release of HP-UX 11i Version 3 or a newer release.

Message Text

```
Warning: couldn't query physical volume "pvname":
The specified path does not correspond to physical volume
attached to this volume group
Couldn't query the list of physical volumes.
```

Cause

This error has the following possible causes:

1. The disk was missing when the volume group was activated, but was later restored. This typically occurs when a system is rebooted or the volume group is activated with a disk missing, uncabled, or powered down.
2. The disk LVM header was overwritten with the wrong volume group information. If the disk is shared between two systems, one system might not be aware that the disk was already in a volume group. To confirm, check the volume group information using the `dump_lvmtab` command (available from your HP support representative) and look for inconsistencies. For example:

   ```
   # dump_lvmtab -s | more
   SYSTEM : 0x35c8cf58
   TIME   : 0x3f9acc69 : Sat Oct 25 15:18:01 2003
   FILE   : /etc/lvmtab
   HEADER : version:0x03e8   vgnum:7
   VG[00]  VGID:35c8cf58 3dd13164 (@0x00040c) pvnum:2 state:0 /dev/vg00
     (00)  VGID:35c8cf58 3dd13164 PVID:35c8cf58 3dd13164 /dev/dsk/c0t6d0
     (01)  VGID:35c8cf58 3dd13164 PVID:35c8cf58 3dda4694 /dev/dsk/c4t6d0
   VG[01]  VGID:065f303f 3e63f01a (@0x001032) pvnum:92 state:0 /dev/vg01
     (00) !VGID:35c8cf58 3f8df316 PVID:065f303f 3e63effa /dev/dsk/c40t0d0
     (01) !VGID:35c8cf58 3f8df316 PVID:065f303f 3e63effe /dev/dsk/c40t0d4
     (02) !VGID:35c8cf58 3f8df316 PVID:065f303f 3e63f003 /dev/dsk/c40t1d0
   ...
   ```

   In this example, the VGIDs for the disks in `/dev/vg01` are not consistent; inconsistencies are marked `!VGID`.

Recommended Action

1. Use `ioscan` and `diskinfo` to confirm that the disk is functioning properly. Reactivate the volume group using the following command:

   ```
   # vgchange -a y vgname
   ```

2. There are several methods of recovery from this error. If you are not familiar with the commands outlined in the following procedures, contact your HP support representative for assistance.
   a. Restore a known good configuration to the disks using `vgcfgrestore`. Be sure to use a valid copy dated before the first occurrence of the problem. For example:

      ```
      # vgcfgrestore -n vgname pvname
      ```

   b. Recreate the volume group and its logical volumes, restoring the data from the most current backup. See “Creating a Volume Group” (page 46) and “Creating a Logical Volume” (page 49).
   c. Export and reimport the volume group, as described in “Exporting a Volume Group” (page 54) and “Importing a Volume Group” (page 54). For example:

      ```
      # vgexport -m vgname.map -v -f vgname.file /dev/vgname
      # vgimport -m vgname.map -v -f vgname.file /dev/vgname
      ```

Message Text

`vgchange: Couldn't set the unique id for volume group "/dev/vgname"`

Cause

There are multiple LVM group files with the same minor number.

Recommended Action

List the LVM group files. If there are any duplicate minor numbers, export one of the affected volume groups, optionally create a new group file with a unique minor number, and reimport the volume group. If you are not familiar with this procedure, contact your HP support representative for assistance.

```
# ll /dev/*/group
# vgexport -m vgname.map -v -f vgname.file /dev/vgname
# mkdir /dev/vgname
# mknod /dev/vgname/group c 64 unique_minor_number
# vgimport -m vgname.map -v -f vgname.file /dev/vgname
```

vgcreate(1M)

Message Text

```
vgcreate: "/dev/vgname/group": not a character device.
```

Cause

The volume group device file does not exist, and this version of the `vgcreate` command does not automatically create it.

Recommended Action

Create the directory for the volume group and create a group file, as described in "Creating the Volume Group Device File" (page 46).

Message Text

```
vgcreate: Volume group "/dev/vgname" could not be created:
Error: The physical volume "pvname" contains BDRA.
It cannot be added into volume group "/dev/vgname" because
this version does not support bootable disks.
```

Cause

The physical volume *pvname* is a bootable disk, and *vgname* is a Version 2.x volume group. Version 2.x volume groups do not support bootable physical volumes.

Recommended Action

Use the `pvcreate` command without the `-B` option to reinitialize the disk, as described in "Initializing a Disk for LVM Use" (page 46). For example:

```
# pvcreate -f pvname
```

Use the `vgmodify` command to convert the disk to a nonbootable disk, as described in "Changing Physical Volume Boot Types" (page 72). Alternately, use the `pvcreate` command without the `-B` option to reinitialize the disk, as described in "Initializing a Disk for LVM Use" (page 46). For example:

```
# pvcreate -f pvname
```

Message Text

```
vgcreate: Volume group "/dev/vgname" could not be created:
VGRA for the disk is too big for the specified parameters.
Increase the extent size or decrease max_PVs/max_LVs and try again.
```

Cause

The volume group reserved area (VGRA) at the front of each LVM disk cannot hold all the information about the disks in this volume group. This error typically occurs if you use disks larger than 100 GB.

Recommended Action

Adjust the volume group creation parameters. Use the `-s` option of the `vgcreate` command to select an extent size larger than 4 MB, or use the `-p` option to select a smaller number of physical volumes. For more information on these options, see *vgcreate*(1M). For recommendations on extent sizes, see Appendix C (page 131).

vgdisplay(1M)

Message Text

```
vgdisplay: Couldn't query volume group "/dev/vgname".
Possible error in the Volume Group minor number;
Please check and make sure the group minor number is unique.
vgdisplay: Cannot display volume group "/dev/vgname".
```

Cause

This error has the following possible causes:

1. There are multiple LVM group files with the same minor number.
2. Serviceguard was previously installed on the system and the `/dev/slvmvg` device file still exists.

Recommended Action

1. List the LVM group files. If there are any duplicate minor numbers, export one of the affected volume groups, optionally create a new group file with a unique minor number, and reimport the volume group. If you are not familiar with this procedure, contact your HP support representative for assistance.

   ```
   # ll /dev/*/group
   # vgexport -m vgname.map -v -f vgname.file /dev/vgname
   # mkdir /dev/vgname
   # mknod /dev/vgname/group c 64 unique_minor_number
   # vgimport -m vgname.map -v -f vgname.file /dev/vgname
   ```

2. Remove the `/dev/slvmvg` device file and re-create the `/etc/lvmtab` and `/etc/lvmtab_p` files using the following commands:

   ```
   # rm /dev/slvmvg
   # mv /etc/lvmtab /etc/lvmtab.old
   # mv /etc/lvmtab_p /etc/lvmtab_p.old
   # vgscan -v
   ```

Message Text

```
vgdisplay: /etc/lvmtab: No such file or directory
vgdisplay: No volume group name could be read from "/etc/lvmtab".
vgdisplay: /etc/lvmtab: No such file or directory
vgdisplay: No volume group name could be read from "/etc/lvmtab".
```

Cause

One of the LVM configuration files, either `/etc/lvmtab` or `/etc/lvmtab_p`, is missing.

Recommended Action

Create the `/etc/lvmtab` and `/etc/lvmtab_p` files using the following command:

```
# vgscan -v
```

For additional information on the `vgscan` command and its option, see *vgscan*(1M).

Message Text

```
Warning: couldn't query physical volume "pvname":
The specified path does not correspond to physical volume
attached to this volume group
Couldn't query the list of physical volumes.
```

Cause

The possible causes of this error are described under the "vgchange(1M)" (page 117) error messages.

Recommended Action

See the recommended actions under the "vgchange(1M)" (page 117) error messages.

vgextend(1M)

Message Text

```
vgextend: Not enough physical extents per physical volume.
Need: #, Have: #.
```

Cause

The disk size exceeds the volume group maximum disk size. This limitation is defined when the volume group is created, as a product of the extent size specified with the `-s` option of `vgcreate` and the maximum number of physical extents per disk specified with the `-e` option. Typically, the disk is successfully added to the volume group, but not all of the disk is accessible.

Recommended Action

Use the `vgmodify` command to adjust the maximum number of physical extents per disk. Alternately, you can re-create the volume group with new values for the `-s` and `-e` options.

vgimport(1M)

Message Text

`vgimport: "/dev/`*`vgname`*`/group": not a character device.`

Cause

The volume group device files do not exist, and this version of the `vgimport` command does not automatically create them.

Recommended Action

Create the directory for the volume group, and create a group file, as described in "Creating the Volume Group Device File" (page 46).

Message Text

`Verification of unique LVM disk id on each disk in the volume group`
`/dev/`*`vgname`*` failed.`

Cause

There are two possible causes for this message:

1. The `vgimport` command used the `-s` option and two or more disks on the system have the same LVM identifier; this can happen when disks are created with BC copy or cloned with `dd`.
2. LVM was unable to read the disk header; this can happen when you create new logical units on a SAN array.

Recommended Action

1. Use `vgimport` without the `-s` option. Alternately, use `vgchgid` to change the LVM identifiers on copied or cloned disks.
2. Retry the `vgimport` command.

/var/adm/syslog/syslog.log

Message Text

```
LVM: VG mm 0xnn0000:
 Data in one or more logical volumes on PV nn 0x0nn000
 was lost when the disk was replaced.
This occurred because the disk contained the only copy of the data.
Prior to using these logical volumes, restore the data from backup.
```

Cause

LVM cannot synchronize the data on a replaced disk automatically, for example, when LVM discovers an unmirrored logical volume residing on a disk that was just replaced.

Recommended Action

Restore the contents of the logical volume from backup.

Message Text

```
LVM: VG mm 0xnn0000: PVLink nn 0x0nn000 Detached.
```

Cause:

This message is advisory. It is generated whenever a disk path is detached.

Recommended Action

None.

Message Text

```
LVM: vg[nn] pv[nn] No valid MCR, resyncing all mirrored MWC LVs on the PV
```

Cause

This message appears when you import a volume group from a previous release of HP-UX. The format of the MWC changed at HP-UX 11i Version 3, so if the volume group contains mirrored logical volumes using MWC, LVM converts the MWC at import time. It also performs a complete resynchronization of all mirrored logical volumes, which can take substantial time.

Recommended Action

None.

Message Text

```
LVM: vg 64 0xnnnnnn: Unable to register
 for event notification on device 0xnnnnnnnn (1)
```

Cause

This message can be displayed on the first system boot after upgrading to HP-UX 11i Version 3. It is a transient message caused by updates to the I/O configuration. Later in the boot process, LVM registers for event notification again, and succeeds.

Recommended Action

None.

Reporting Problems

If you are unable to solve a problem with LVM, follow these steps:

1. Read the *HP-UX Logical Volume Manager and MirrorDisk/UX Release Notes* to see if the problem is known. If it is, follow the solution offered to solve the problem.
2. Determine if the product is still under warranty or if your company purchased support services for the product. Your operations manager can supply you with the necessary information.
3. Access http://www.itrc.hp.com and search the technical knowledge databases to determine if the problem you are experiencing has been reported already. The type of documentation and resources you have access to depend on your level of entitlement.

   NOTE: The ITRC resource forums at http://www.itrc.hp.com offer peer-to-peer support to solve problems and are free to users after registration.

   If this is a new problem or if you need additional help, log your problem with the HP Response Center, either online through the support case manager at http://www.itrc.hp.com, or by calling HP Support. If your warranty has expired or if you do not have a valid support contract for your product, you can still obtain support services for a fee, based on the amount of time and material required to solve your problem.
4. If you are asked to supply any information pertaining to the problem, gather the requested information and submit it.

A LVM Specifications and Limitations

This appendix discusses LVM product specifications.

NOTE: Do not infer that a system configured to these limits is automatically usable.

Table A-1 Volume Group Version Maximums

	Version 1.0 Volume Groups	Version 2.0 Volume Groups	Version 2.1 Volume Groups
Maximum data on a single HP-UX system	128 PB	1024 PB	1024 PB
Maximum number of volume groups on a system	256	512[1]	2048[1]
Maximum number of physical volumes in a volume group	255	511	2048
Maximum number of logical volumes in a volume group	255	511	2047
Maximum size of a physical volume	2 TB	16 TB	16 TB
Maximum size of a volume group	510 TB	2048 TB	2048 TB
Maximum size of a logical volume	16 TB	256 TB	256 TB
Maximum size of a physical extent	256 MB	256 MB	256 MB
Maximum size of a stripe	32 MB	256 MB	256 MB
Maximum number of stripes	255	511	511
Maximum number of logical extents per logical volume	65535	33554432	33554432
Maximum number of physical extents per physical volume	65535	16777216	16777216
Maximum number of mirror copies (MirrorDisk/UX product required)	2	5	5

1 The limit of 2048 volume groups is shared between Version 2.0 and Version 2.1 volume groups. Volume groups of both versions can be created with volume group numbers ranging from 0-2047. However, the maximum number of Version 2.0 volume groups that can be created is 512.

Table A-2 Version 1.0 Volume Group Limits

Parameter	Command to Set/Change Parameter	Minimum Value	Default Value	Maximum Value
Number of volume groups on a system	n/a	0	n/a	256
Number of physical volumes in a volume group	`vgcreate -p` *max_pv* `vgmodify -p` *max_pv*	1	16	255
Number of logical volumes in a volume group	`vgcreate -l` *max_lv* `vgmodify -l` *max_lv*	1	255	255
Size of a physical volume	`pvcreate -s` *pv_size*	1 PE	LUN Capacity	2 TB
Size of a logical volume	`lvcreate -L` *lv_size* `lvextend -L` *lv_size*	0	0	16 TB
Size of a physical extent	`vgcreate -s` *pe_size*	1 MB	4 MB	256 MB
Size of a stripe	`lvcreate -l` *stripe_size*	4 KB	8 KB	32 MB
Number of stripes	`lvcreate -i` *stripes*	2	n/a	Number of PVs in VG
Number of logical extents per logical volume	`lvcreate -l` *max_le* `lvextend -l` *max_le*	0	0	65535
Number of physical extents per physical volume	`vgcreate -e` *max_pe* `vgmodify -e` *max_pe*	1	1016[1]	65535
Number of mirror copies (MirrorDisk/UX product required)	`lvcreate -m` *copies* `lvextend -m` *copies*	0	0	2

1 If more than 1016 extents are necessary to access all of the first physical volume in the volume group, `vgcreate` increases the default value to the size of the first physical volume divided by the physical extent size.

Table A-3 Version 2.x Volume Group Limits

Parameter	Command to Set/Change Parameter	Minimum Value	Default Value	Maximum Value
Number of volume groups on a system	n/a	0	n/a	2048[1]
Number of physical volumes in a volume group	n/a	511	511 (2.0) 2048 (2.1)	511 (2.0) 2048 (2.1)
Number of logical volumes in a volume group	n/a	511	511 (2.0) 2047 (2.1)	511 (2.0) 2047 (2.1)
Size of a volume group	`vgcreate -S` *max_vgsize*	1 MB[2]	n/a	2 PB
Size of a physical volume	`pvcreate -s` *pv_size*	1 PE	LUN Capacity	16 TB
Size of a logical volume	`lvcreate -L` *lv_size* `lvextend -L` *lv_size*	0	0	256 TB
Size of a physical extent	`vgcreate -s` *pe_size*	1 MB	n/a	256 MB
Size of a stripe	`lvcreate -I` *stripe_size*	4 KB	n/a	256 MB
Number of stripes	`lvcreate -i` *stripes*	2	n/a	Number of PVs in VG
Number of logical extents per logical volume	`lvcreate -l` *max_le* `lvextend -l` *max_le*	0	0	33554432
Number of physical extents per physical volume	n/a	1	LUN Capacity ÷ PE Size	16384
Number of mirror copies (MirrorDisk/UX product required)	`lvcreate -m` *copies* `lvextend -m` *copies*	0	0	5

1 The limit of 2048 volume groups is shared between Version 2.0 and Version 2.1 volume groups. Volume groups of both versions can be created with volume group numbers ranging from 0-2047. However, the maximum number of Version 2.0 volume groups that can be created is 512.

2 If the total size of the specified physical volumes is larger than *max_vgsize*, `vgcreate` adjusts the minimum volume group size to the total size.

B LVM Command Summary

This appendix contains a summary of the LVM commands and descriptions of their use.

Table B-1 LVM Command Summary

Command	Description and Example
`extendfs`	Extends a file system: `# extendfs /dev/vg00/rlvol3`
`lvmadm`	Displays the limits associated with a volume group version: `# lvmadm -t -V 2.0`
`lvchange`	Changes the characteristics of a logical volume: `# lvchange -t 60 /dev/vg00/lvol3`
`lvcreate`	Creates a logical volume in a volume group: `# lvcreate -L 100 /dev/vg00`
`lvdisplay`	Displays information about logical volumes: `# lvdisplay -v /dev/vg00/lvol1`
`lvextend -m`	Adds a mirror to a logical volume: `# lvextend -m 1 /dev/vg00/lvol3`
`lvextend -L`	Increases the size of a logical volume: `# lvextend -L 120 /dev/vg00/lvol3`
`lvlnboot`	Prepares a logical volume to become a root, swap, or dump area: `# lvlnboot -d /dev/vg00/lvol2`
`lvmerge`	Merges split volumes into one logical volume: `# lvmerge /dev/vg00/lvol4b /dev/vg00/lvol4`
`lvreduce -L`	Decreases the size of a logical volume: `# lvreduce -L 100 /dev/vg00/lvol3`
`lvreduce -m`	Decreases the number of mirror copies of a logical volume: `# lvreduce -m 0 /dev/vg00/lvol3`
`lvremove`	Removes logical volumes from a volume group: `# lvremove /dev/vg00/lvol6`
`lvrmboot`	Removes a logical volume link to root, swap, or dump: `# lvrmboot -d /dev/vg00/lvol2`
`lvsplit`	Splits a mirrored logical volume into two logical volumes: `# lvsplit /dev/vg00/lvol4`
`lvsync`	Synchronizes stale logical volume mirrors: `# lvsync /dev/vg00/lvol1`
`pvchange`	Changes the characteristics of a physical volume: `# pvchange -a n /dev/disk/disk2`
`pvck`	Performs a consistency check on a physical volume: `# pvck /dev/disk/disk47_p2`

Table B-1 LVM Command Summary *(continued)*

Command	Description and Example
`pvcreate`	Creates a physical volume be to used as part of a volume group: `# `**`pvcreate /dev/rdisk/disk2`**
`pvdisplay`	Displays information about a physical volume: `# `**`pvdisplay -v /dev/disk/disk2`**
`pvmove`	Moves extents from one physical volume to another: `# `**`pvmove /dev/disk/disk2 /dev/disk/disk3`**
`pvremove`	Removes LVM data structures from a physical volume: `# `**`pvremove /dev/rdisk/disk2`**
`vgcfgbackup`	Saves the LVM configuration for a volume group: `# `**`vgcfgbackup vg00`**
`vgcfgrestore`	Restores the LVM configuration: `# `**`vgcfgrestore -n /dev/vg00 /dev/rdisk/disk2`**
`vgchange`	Turns a volume group off or on: `# `**`vgchange -a y /dev/vg00`**
`vgchgid`	Changes the volume group ID of a physical volume: `# `**`vgchgid /dev/rdisk/disk3`**
`vgcreate`	Creates a volume group: `# `**`vgcreate /dev/vg01 /dev/disk/disk2 /dev/disk/disk3`**
`vgdisplay`	Displays information about a volume group: `# `**`vgdisplay -v /dev/vg00`**
`vgextend`	Extends a volume group by adding a physical volume: `# `**`vgextend /dev/vg00 /dev/disk/disk2`**
`vgexport`	Removes a volume group from the system: `# `**`vgexport /dev/vg01`**
`vgimport`	Adds an existing volume group to the system: `# `**`vgimport -v /dev/vg04`**
`vgmodify`	Modifies the configuration parameters of a volume group: `# `**`vgmodify -v -t -n -r vg32`**
`vgscan`	Scans the system disks for volume groups: `# `**`vgscan -v`**
`vgreduce`	Reduces a volume group by removing one or more physical volumes from it: `# `**`vgreduce /dev/vg00 /dev/disk/disk2`**
`vgremove`	Removes the definition of a volume group from the system and the disks: `# `**`vgremove /dev/vg00 /dev/disk/disk2`**
`vgsync`	Synchronizes all mirrored logical volumes in the volume group: `# `**`vgsync vg00`**

C Volume Group Provisioning Tips

This appendix contains recommendations for parameters to use when creating your volume groups.

Choosing an Optimal Extent Size for a Version 1.0 Volume Group

When creating a Version 1.0 volume group, the `vgcreate` command may fail and display a message that the extent size is too small or that the VGRA is too big. In this situation, you must choose a larger extent size and run `vgcreate` again.

Increasing the extent size increases the data area marked stale when a write to a mirrored logical volume fails. That can increase the time required to resynchronize stale data. Also, more space may be allocated to each logical volume because the space is allocated in units of extent size. Therefore, the optimal extent size is the smallest value that can be used to successfully create the volume group with the desired configuration parameters.

The minimum extent size for a volume group is calculated using the maximum number of logical volumes (MAXLVs) and physical volumes (MAXPVs) in the volume group and the maximum number of physical extents (MAXPXs) per each physical volume.

For a volume group with bootable physical volumes, the metadata must fit within 768 KB. Therefore, running `vgcreate` with a set of values for MAXLVs, MAXPVs and MAXPXs that succeed on a volume group without bootable physical volumes may fail on a volume group with bootable physical volumes. In this situation, if you must add a bootable physical volume to a volume group, recreate the volume group by giving lesser values for these arguments. By far the biggest factor in the size of the metadata is the values for MAXPVs and MAXPXs. Alternatively, convert the bootable physical volume to a normal physical volume by running `pvcreate` on that physical volume without the `–B` option and then adding it to the volume group. For a physical volume that is already part of a volume group, you can use `vgmodify` to change a physical volume from a bootable to a normal physical volume.

Sample Shell Script

The following shell script creates and compiles a small program that displays the minimum extent size for a given volume group:

```
#!/usr/bin/sh
cat << EOF > vgrasize.c
#include <stdio.h>

#define BS 1024 /* Device block Size */
#define roundup(val, rnd) (((val + rnd - 1) / rnd) * rnd)

main(int argc, char *argv[])
{
    int i, length, lvs, pvs, pxs;
    if (argc != 4) {
        /* Usage example:
         * Maximum LVs in the VG = 255
         * Maximum PVs in the VG = 16
         * Maximum extents per PV = 2500
         *
         * $ vgrasize 255 16 2500
         */
        printf("USAGE: %s MAXLVs MAXPVs MAXPXs\n", argv[0]);
        exit(1);
    }
    lvs = atoi(argv[1]);
    pvs = atoi(argv[2]);
    pxs = atoi(argv[3]);
    length = 16 + 2 * roundup(2 +
                     (roundup(36 + ((3 * roundup(pvs, 32)) / 8) +
                             (roundup(pxs, 8) / 8) * pvs, BS) +
                      roundup(16 * lvs, BS) +
                      roundup(16 + 4 * pxs, BS) * pvs) / BS, 8);

    if (length > 768) {
        printf("Warning: A bootable PV cannot be added to a VG \n"
        "created with the specified argument values. \n"
        "The metadata size %d Kbytes, must be less \n"
        "than 768 Kbytes.\n"
        "If the intention is not to have a boot disk in this \n"
        "VG then do not use '-B' option during pvcreate(1M) \n"
        "for the PVs to be part of this VG. \n", length);
    }

    length = roundup(length, 1024) / 1024;

    if (length > 256 ) {
        printf("Cannot configure a VG with the maximum values"
        " for LVs, PVs and PXs\n");
        exit(1);
    }

    for (i = 1; i < length ; i = i << 1) { }

    printf("\nMinimum extent size for this configuration = %d MB\n", i);

    exit(0);
}
EOF
make vgrasize
```

Glossary

Allocation Policy The LVM allocation policy governing how disk space is distributed to logical volumes and how extents are laid out on an LVM disk. LVM allocates disk space in terms of strict vs. non-strict and contiguous vs. noncontiguous. Strict allocation requires that mirror copies reside on different LVM disks. Contiguous allocation requires that no gaps exist between physical extents on a single disk.

Disk Spanning The allocation of a logical volume across multiple disks, allowing the volume size to exceed the size of a single disk.

I/O Channel Separation A configuration of disks useful for segregating highly I/O-intensive areas. For example, you might have a database on one channel and file systems on another. When mirroring logical volumes using HP MirrorDisk/UX, you can spread the mirrored copies over different I/O channels to increase system and data availability.

Logical Extents Fixed-size addressable areas of space on a logical volume. The basic allocation unit for a logical volume, a logical extent is mapped to a physical extent; thus, if the physical extent size is 4 MB, the logical extent size will also be 4 MB. The size of a logical volume is determined by the number of logical extents configured.

Logical Volume A virtual storage device of flexible size that can hold a file system, raw data, dump area, or swap. Because its data are distributed logically (rather than physically), a single logical volume can be mapped to one LVM disk or span multiple disks. A logical volume appears to the administrator as though it was a single disk.

Logical Volume Manager An operating system software module that implements virtual (logical) disks to extend, mirror, and improve the performance of physical disk access.

Mirroring Simultaneous replication of data, ensuring a greater degree of data availability. LVM can map identical logical volumes to multiple LVM disks, thus providing the means to recover easily from the loss of one copy (or multiple copies in the case of multi-way mirroring) of data. Mirroring can provide faster access to data for applications using more data reads than writes. Mirroring requires the MirrorDisk/UX product.

Physical Extents Fixed-size addressable areas of space on an LVM disk. They are the basic allocation units for a physical volume. Physical extents map to areas on logical volumes called logical extents.

Physical Volume A disk that has been initialized by LVM for inclusion in a volume group; also called an LVM disk. As with standard disks, an LVM disk (physical volume) is accessed via a raw device file (for example, `/dev/rdisk/disk3`). Use the HP SMH or the `pvcreate` command to initialize a disk as a physical volume.

Physical Volume Group A subset of physical volumes within a volume group, each with a separate I/O channel or interface adapter to achieve higher availability of mirrored data.

Quorum The requirement that a certain number of LVM disks be present in order to change or activate a volume group. To activate a volume group, quorum requires the number of available LVM disks to be *more than* half the number of configured LVM disks that were present when the volume group was last active. To make a configuration change, the quorum requirement is *at least* half. If there is no quorum, LVM prevents the operation. Quorum is checked both during configuration changes (for example, when creating a logical volume) and at state changes (for example, if a disk fails). Quorum ensures the consistency and integrity of the volume groups. The `vgchange` command with the `-q n` option can be used to override quorum check, but this should be used with caution.

Synchronization The process of updating stale (non-current) copies of mirrored logical extents by copying data from a fresh (current) copy of the logical volume. Synchronization keeps mirrored logical volumes consistent by ensuring that all copies contain the same data.

Volume Group A collection of one or more LVM disks from which disk space may be allocated to individual logical volumes. A disk can belong to only one volume group. A volume group is accessed through the group file (for example, `/dev/vg01/group`) in that volume group's directory. Use HP SMH or the `vgcreate` command to create a volume group.

Index

G

H

I

L

M

N

P