

Hewlett-Packard Computer Systems  
Training Course

# Logical Volume Manager Basics Self-Paced Training

Course No.: CES2-LVMBASICS



**Technical Planning and Education Center  
Bd Steve Biko, 38090 VILLEFONTAINE**

**P/N: H6167+49A-90001 Printed in France. 12/98  
Copyright © 1998 Hewlett-Packard Company**

◆ Copyright

## NOTICE

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

HEWLETT-PACKARD PROVIDES THIS MATERIAL "AS IS" AND MAKES NO WARRANTY OF ANY KIND, EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. HEWLETT-PACKARD SHALL NOT BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES (INCLUDING LOST PROFITS) IN CONNECTION WITH THE FURNISHING, PERFORMANCE OR USE OF THIS MATERIAL WHETHER BASED ON WARRANTY, CONTRACT OR OTHER LEGAL THEORY.

**Some states do not allow the exclusion of implied warranties or the limitation or exclusion of liability for incidental or consequential damages, so the above limitation and exclusions may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.**

**Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.**

**This document contains proprietary information which is protected by copyright. All rights are reserved. No part of this document may be photocopied, reproduced or translated to another program language without the prior written consent of Hewlett-Packard Company.**

Copyright © 1998 by HEWLETT PACKARD COMPANY

**DAY 1**

<b>INTRODUCTION.....</b>	<b>1</b>
<b>1.0 THE BASICS.....</b>	<b>3</b>
<b>1.1 Concepts .....</b>	<b>4</b>
Logical Volume Manager Basics .....	4
LVM Spans Disks .....	6
LVM's Logical to Physical Extent Mapping .....	8
<b>1.2 Rules .....</b>	<b>16</b>
Logical Volume Manager Rules .....	16
<b>1.3 Terminology .....</b>	<b>20</b>
Glossary of LVM terms .....	20
<b>2.0 STRUCTURES.....</b>	<b>26</b>
<b>2.1 Create'ing .....</b>	<b>28</b>
<b>2.2 Display'ing LVM Information.....</b>	<b>37</b>
For Physical Volumes .....	37
For Volume Groups .....	39
For Logical Volumes .....	42
To display kernel devices on LVM bootable Disks .....	44
The contents of /etc/lvmtab .....	46
For Swap information (all swap including LVM) .....	47
LVM Device Files .....	48
LVM Commands – Quick Reference .....	50
<b>3.0 MODIFYING .....</b>	<b>52</b>
<b>3.1 Volume Groups .....</b>	<b>54</b>
Vgextend Example .....	54
VGREDUCE .....	55
VGREMOVE .....	55
<b>3.2 Logical Volumes .....</b>	<b>58</b>
Increasing the Size of a Logical Volume .....	58
Extending a Logical Volume to a Specific Disk .....	59
Extending the File System .....	60
Reducing the Size of a Logical Volume .....	61
Moving Data in Logical Volumes From Disk to Disk .....	62
Removing Logical Volumes .....	63
<b>DAY 2</b>	
<b>4.0 MIRRORS .....</b>	<b>66</b>
<b>4.1 Concepts .....</b>	<b>68</b>
<b>4.2 Commands .....</b>	<b>70</b>
Splitting and Merging Mirrored Logical Volumes .....	73
Discussion, Splitting and Merging a Mirrored Logical Volume .....	74
<b>5.0 BOOT DISKS .....</b>	<b>77</b>
<b>5.1 Structures .....</b>	<b>79</b>
<b>5.2 Booting up .....</b>	<b>82</b>
<b>5.3 Emergency boot .....</b>	<b>84</b>
<b>5.4 Creating .....</b>	<b>87</b>
Guidelines for configuring the Root Volume Group .....	89
<b>6.0 RECOVERY .....</b>	<b>92</b>
<b>6.1 LVM Data Structure Backup .....</b>	<b>94</b>
<b>6.2 vgscan .....</b>	<b>97</b>
<b>6.3 vgimport .....</b>	<b>99</b>
Using vgimport and vgexport effectively .....	101
<b>LVM Cookbooks .....</b>	<b>106</b>
<b>Add An LVM Disk .....</b>	<b>108</b>
<b>Extend An LVM Logical Volume .....</b>	<b>112</b>
<b>Reduce the Size of an LVM Logical Volume .....</b>	<b>115</b>
<b>Remove an LVM Volume Group .....</b>	<b>117</b>

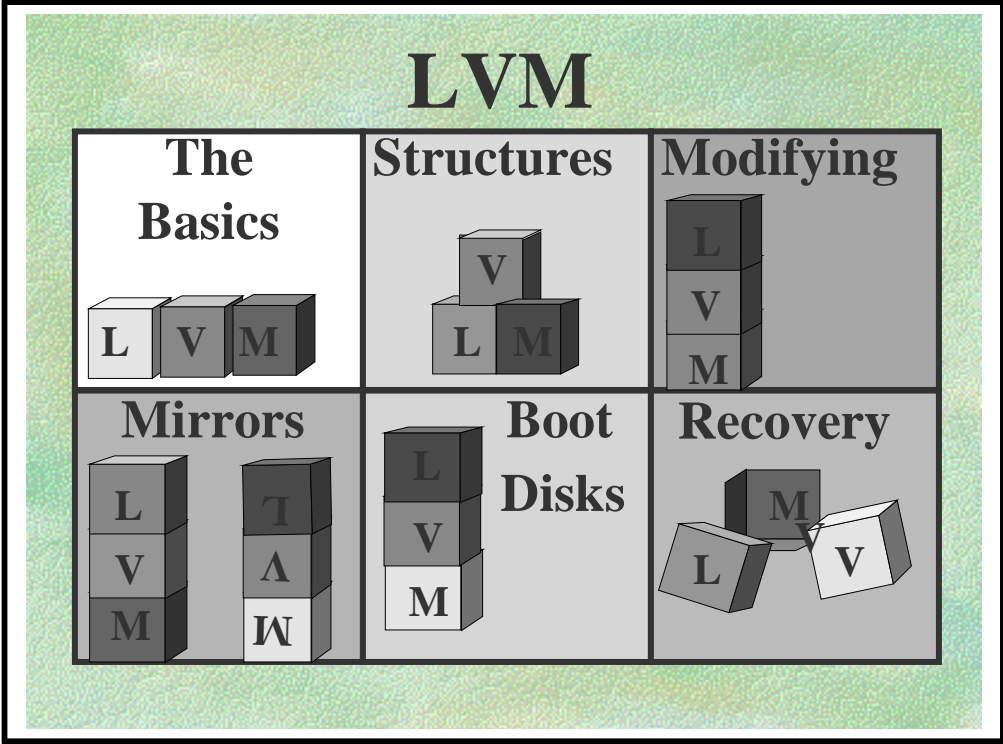
- Moving an LVM Logical Volume ..... 118**
- Exporting and Importing an LVM Volume Group ..... 119**
- Adding Secondary Device Swap on a Logical Volume ..... 122**
- Adding Dumps Devices on a Logical Volume ..... 124**
- Bootng a Damaged LVM Bootable disk..... 126**
- Mirroring a Root Disk using Logical Volumes ..... 128**
- Lvsplit/lvmerge:Backing Up a Mirrored Disk ..... 129**
- Create A Boot Disk ..... 130**
- LOGICAL VOLUME MANAGER LAB EXERCISES..... 132**
  - Objective: ..... 133
  - References: ..... 133
  - Key Commands Used: ..... 133
- Module 1 –Introduction ..... 136**
  - Objective: ..... 136
  - Commands: ..... 136
  - Tasks: ..... 136
- Module 2 – Creating a Logical Volume ..... 138**
  - Objective: ..... 138
  - Commands: ..... 138
  - Tasks: ..... 138
- Module 3 – Creating a Volume Group ..... 140**
  - Objective: ..... 140
  - Commands: ..... 140
  - Tasks: ..... 140
- Module 4 – Extending a Logical Volume ..... 142**
  - Objective: ..... 142
  - Commands: ..... 142
  - Tasks: ..... 142
- Module 5 – Reducing a Logical Volume ..... 144**
  - Objective: ..... 144
  - Commands: ..... 144
  - Tasks: ..... 144
- Module 6 – Adding Secondary Swap Space ..... 145**
  - Objective: ..... 145
  - Commands: ..... 145
  - Tasks: ..... 145
- Module 7 – Removing a Volume Group ..... 147**
  - Objective: ..... 147
  - Commands: ..... 147
  - Tasks: ..... 147
- Module 8 – Extending a Volume Group ..... 148**
  - Objective: ..... 148
  - Commands: ..... 148
  - Tasks: ..... 148
- Module 9 – Creating Dump Logical Volumes ..... 149**
  - Objective: ..... 149
  - Commands: ..... 149
  - Tasks: ..... 149
- Module 10 – Mirroring a Logical Volume ..... 153**
  - Objective: ..... 153
  - Commands: ..... 153
  - Tasks: ..... 153
- Module 11 – Reducing a Volume Group ..... 157**
  - Objective: ..... 157
  - Commands: ..... 157
  - Tasks: ..... 157

<b>Module 12 – Mirroring a Boot Disk.....</b>	<b>158</b>
Objective: .....	158
Commands: .....	158
Tasks: .....	158
<b>Module 13 – Exporting/Importing a Volume Group.....</b>	<b>159</b>
Objective: .....	159
Commands: .....	159
Tasks: .....	159
<b>Module 14 – Rebuilding Volume Group Configuration</b>	<b>162</b>
Objective:.....	162
Commands:.....	162
Tasks:.....	162
<b>Module 15 – Maintenance mode .....</b>	<b>163</b>
Objective: .....	163
Commands: .....	163
Tasks: .....	163
<b>Module 16 – Creating /etc/lvmtab .....</b>	<b>165</b>
Objective: .....	165
Commands: .....	165
Tasks: .....	165
<b>Module 17 – Rebuilding the LVM Root Disk’s Logical Interchange Format (LIF)</b>	<b>166</b>
<b>Utilities .....</b>	
Objective: .....	166
Commands: .....	166
Tasks: .....	166
<b>LOGICAL VOLUME MANAGER LAB ANSWERS</b>	<b>171</b>
Objective: .....	171
References: .....	171
Key Commands Used: .....	171
<b>Module 1 –Introduction .....</b>	<b>174</b>
Objective: .....	174
Commands: .....	174
Tasks: .....	174
<b>Module 2 – Creating a Logical Volume .....</b>	<b>176</b>
Objective: .....	176
Commands: .....	176
Tasks: .....	176
<b>Module 3 – Creating a Volume Group .....</b>	<b>178</b>
Objective: .....	178
Commands: .....	178
Tasks: .....	178
<b>Module 4 – Extending a Logical Volume .....</b>	<b>181</b>
Objective: .....	181
Commands: .....	181
Tasks: .....	181
<b>Module 5 – Reducing a Logical Volume .....</b>	<b>183</b>
Objective: .....	183
Commands: .....	183
Tasks: .....	183
<b>Module 6 – Adding Secondary Swap Space .....</b>	<b>184</b>
Objective: .....	184
Commands: .....	184
Tasks: .....	184
<b>Module 7 – Removing a Volume Group .....</b>	<b>186</b>
Objective: .....	186
Commands: .....	186

Tasks: .....	186
<b>Module 8 – Extending a Volume Group .....</b>	<b>187</b>
Objective: .....	187
Commands: .....	187
Tasks: .....	187
<b>Module 9 – Creating Dump Logical Volumes .....</b>	<b>188</b>
Objective: .....	188
Commands: .....	188
Tasks: .....	188
<b>Module 10 – Mirroring a Logical Volume .....</b>	<b>191</b>
Objective: .....	191
Commands: .....	191
Tasks: .....	191
<b>Module 11 – Reducing a Volume Group .....</b>	<b>195</b>
Objective: .....	195
Commands: .....	195
Tasks: .....	195
<b>Module 12 – Mirroring a Boot Disk .....</b>	<b>196</b>
Objective: .....	196
Commands: .....	196
Tasks: .....	196
<b>Module 13 –Exporting/Importing a Volume Group.....</b>	<b>197</b>
Objective: .....	197
Commands: .....	197
Tasks: .....	197
<b>Module 14 –Rebuilding Volume Group Information.....</b>	<b>200</b>
Objective: .....	200
Commands: .....	200
Tasks: .....	200
<b>Module 15 – Maintenance mode .....</b>	<b>201</b>
Objective: .....	201
Commands: .....	201
Tasks: .....	201
<b>Module 16 – Creating /etc/lvmtab .....</b>	<b>203</b>
Objective: .....	203
Commands: .....	203
Tasks: .....	203
<b>Module 17 – Rebuilding the LVM Root Disk’s Logical Interchange Format (LIF)</b>	<b>204</b>
<b>Utilities .....</b>	
Objective: .....	204
Commands: .....	204
Tasks: .....	204



◆ Introduction



**INTRODUCTION**

The object of this course is to provide you with enough understanding and knowledge of Logical Volume Manager so you feel confident working on a system that uses LVM.

To do this you will be using this workbook and the On-line Training System OLTS. The OLTS system allows you to practice as "root" via the console port on a live HP9000 that you will have to yourself.

**NOTE** This is not a "dummy" login: you really are "**root**"!

See the OLTS Users Guide available from the Online Course Library (Training WEB) for details on how to access these systems in the US. Before being connected to your system console you need to do a telnet on a reference server (IP address is given in the OLTS User Guide). While connected to this server you get a menu with several items. Please select the item README FIRST.



◆ Introduction

.....  
**IMPORTANTIMPORTANTIMPORTANT**  
.....

**Below are the instructions to manage the two days class:**

**DAY1 :     PERFORM     - Theory**  
**from -1.0 The Basics**  
**to     -3.2 Logicals Volumes**  
**(included)**

**AND THEN**

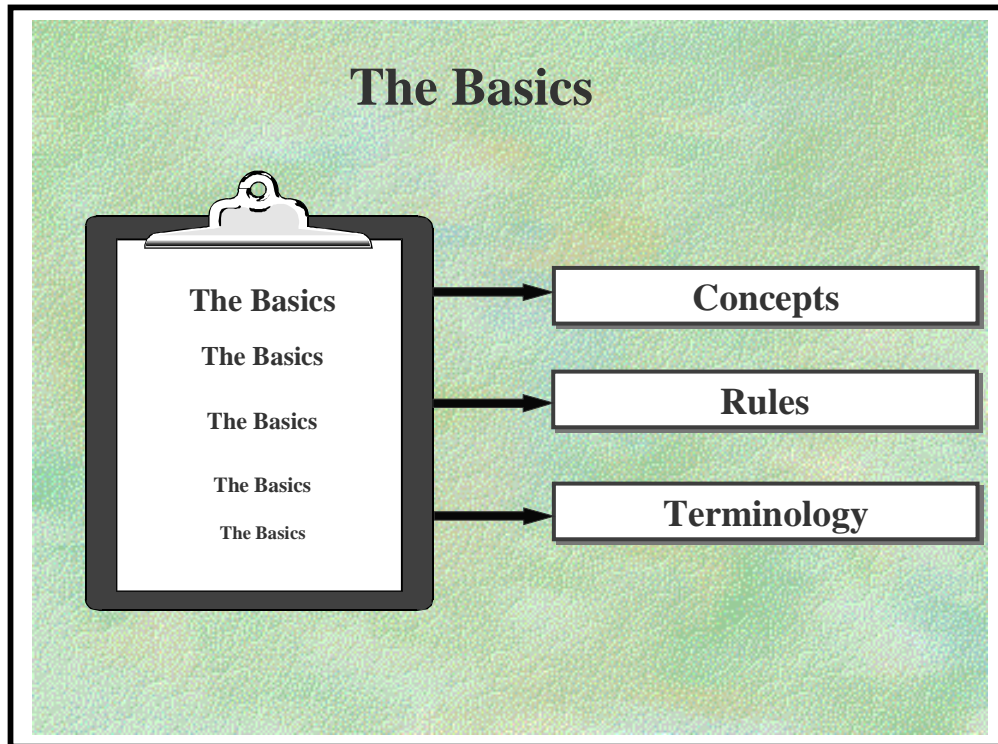
**- Labs**  
**from -Module 1**  
**to     -Module 9**  
**(included).**

**DAY2 :     PERFORM     - Theory**  
**from -4.0 Mirrors**  
**to     -LVM Cookbook**  
**(included)**

**AND THEN**

**- Labs**  
**from -Module 10**  
**to     -Module 17**

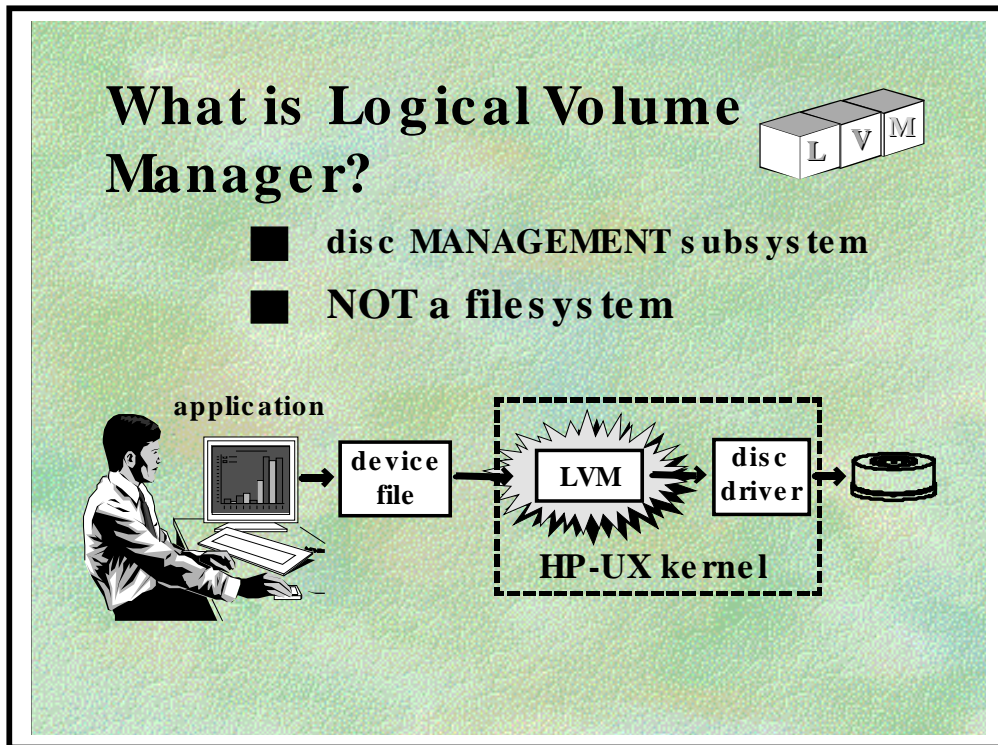
◆ **The Basics**



**This module will look at the following aspects of Logical Volume Manager:**

- \* The basic Concepts of LVM
- \* The Rules of LVM
- \* A glossary of Terms used

## ◆ 1.1 Concepts



### LOGICAL VOLUME MANAGER BASICS

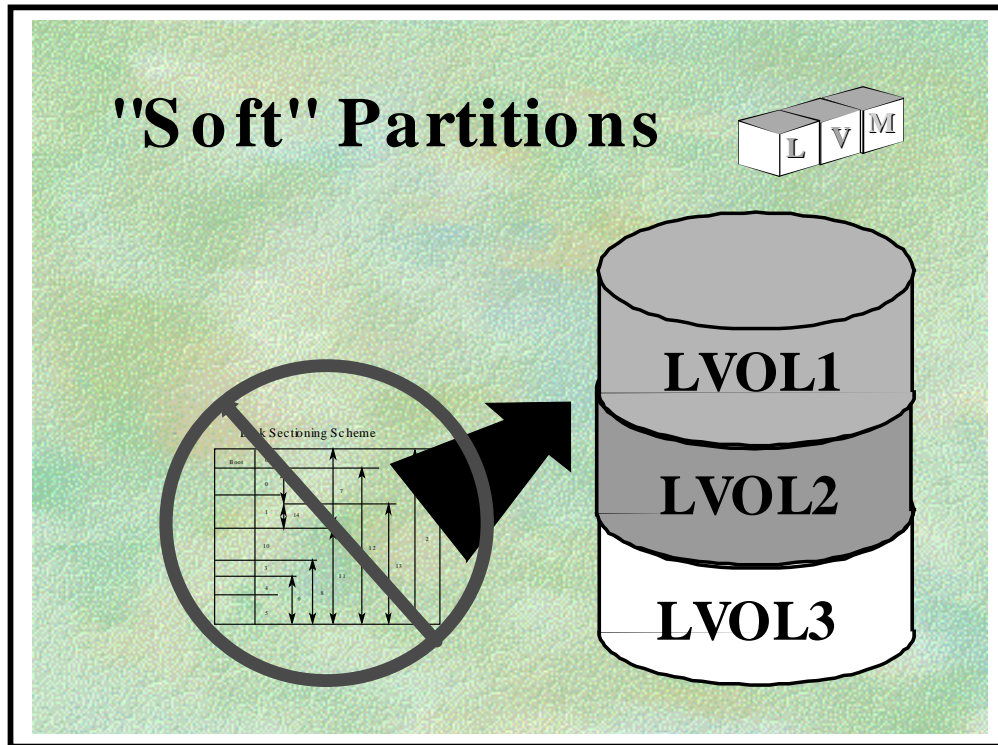
The Logical Volume Manager (LVM) is a disk management subsystem that lets you allocate disk space according to the specific or projected sizes of your file systems or raw data. LVM file systems can exceed the size of a physical disk. This feature is known as disk spanning, because a single file system can span disks.

Using LVM, you can combine one or more disks (physical volumes) into a volume group, which can then be subdivided into one or more logical volumes.

LVM is not a file system. It is a manager that points to the start and end of logical volume data space for each physical disk that the logical volume happens to span.

Logical volumes resemble disk sections, but with some important differences.

## ◆ 1.1 Concepts



- Logical volumes can range in size from 1 MB to 4 GB.

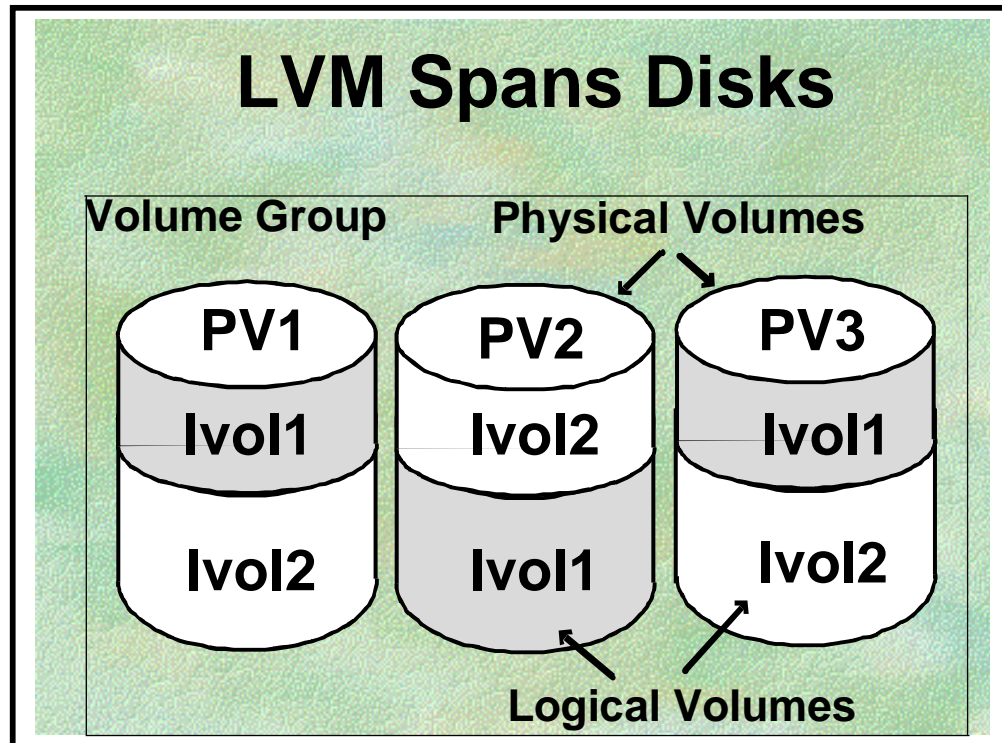
**Maximum size: File system 128GB(The maximum size of files continues to be 2 Gb); Root 2GB; Raw data, dump, swap 2GB**

**NOTE :please refer to /usr/share/doc/10.20RelNotes file**

- Logical volumes can be expanded or reduced in size as needs change.

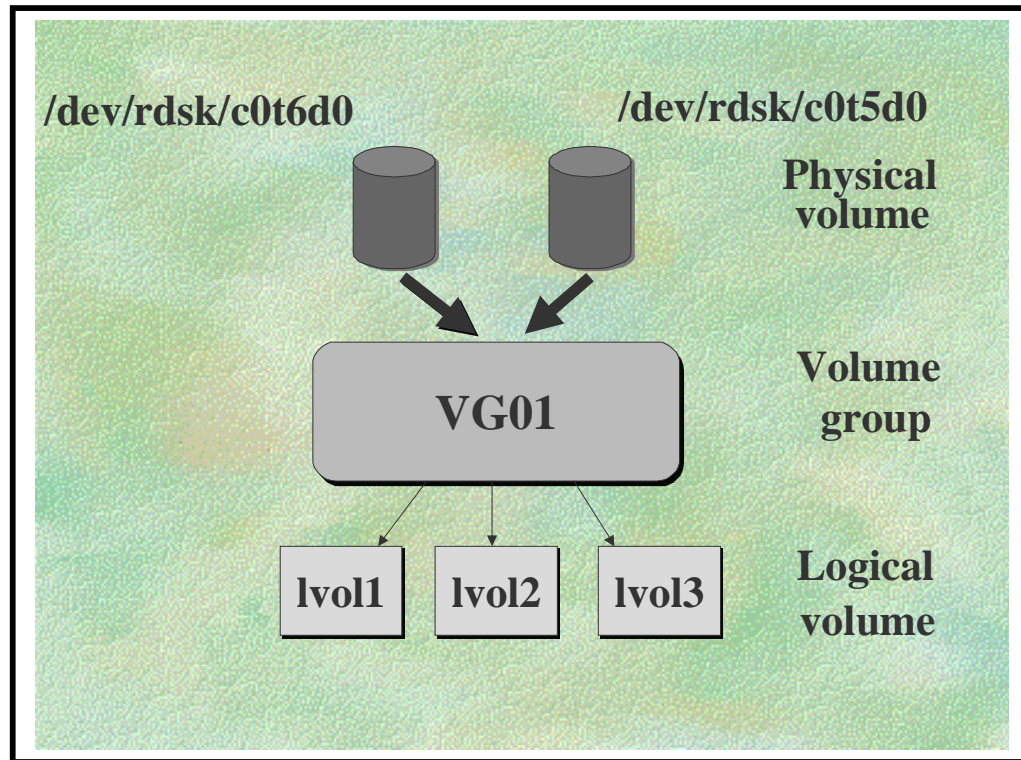
**CAUTION** Reducing a logical volume's size will result in data loss. Always back up logical volume data before reducing logical volume size.

## ◆ 1.1 Concepts



- File systems can be larger than a physical disk's size.
- Logical volumes can grow and shrink allowing more efficient space usage.
- Large-scale applications, such as databases and CAD/CAE systems, whose data requirements often exceed the capacity of a single disk can be expanded across disks.

## ◆ 1.1 Concepts

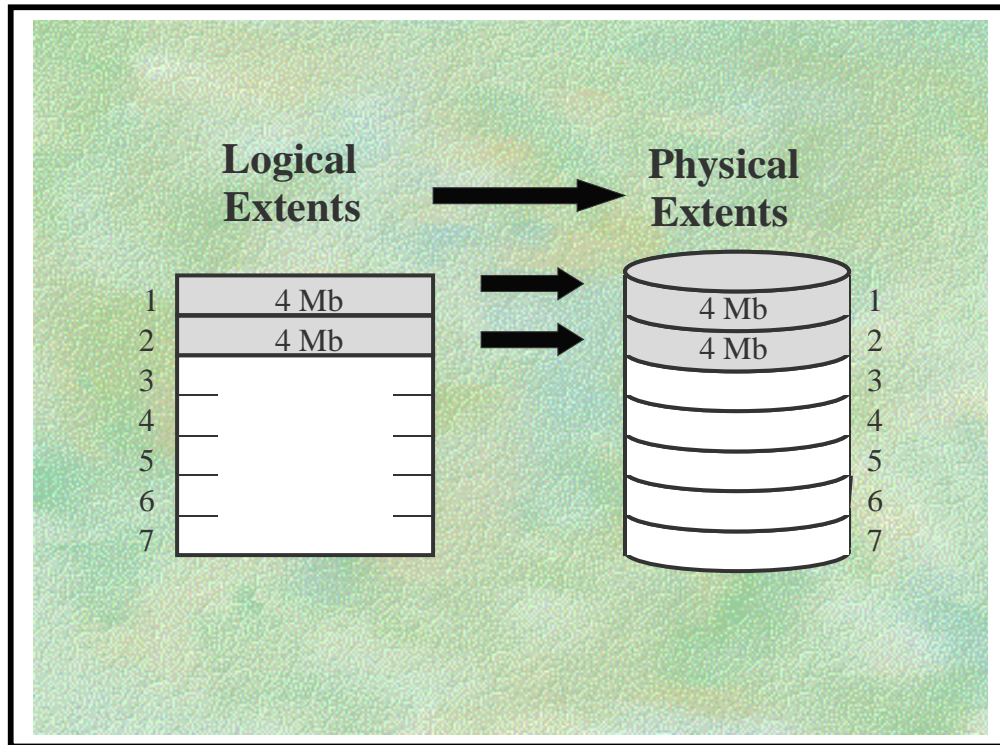


- An LVM system consists of groupings of disks initialized for LVM and organized into volume groups. A volume group might consist of one or many LVM disks; your entire system might consist of one or several volume groups.
- While volume groups represent groupings of one or more LVM disks, logical volumes represent subdivisions of a volume group's total disk space into virtual disks.
- Logical volumes can encompass space on one or more LVM disk and represent only a portion of one or more LVM disks.

So then:

- **Physical Volumes** are combined to form
- **Volume Groups**. These are divided into
- **Logical Volumes**

## ◆ 1.1 Concepts



### LVM's Logical to Physical Extent Mapping

Extent size must be a power of 2 (range: 1 - 256 MB). Set by `vgcreate -s`

Logical Extent size = Physical Extent size

The Physical Extent size is the same for all Physical Volumes in the Volume Group.

The LVM subsystem maps logical extents to physical extents via a translation table that resides on the LVM disk. When the volume group is activated, the table resides in real memory. LVM translates incoming read and write requests to the correct address of the physical extent, then sends the request to the corresponding physical block.

◆ **1.1 Concepts**

---

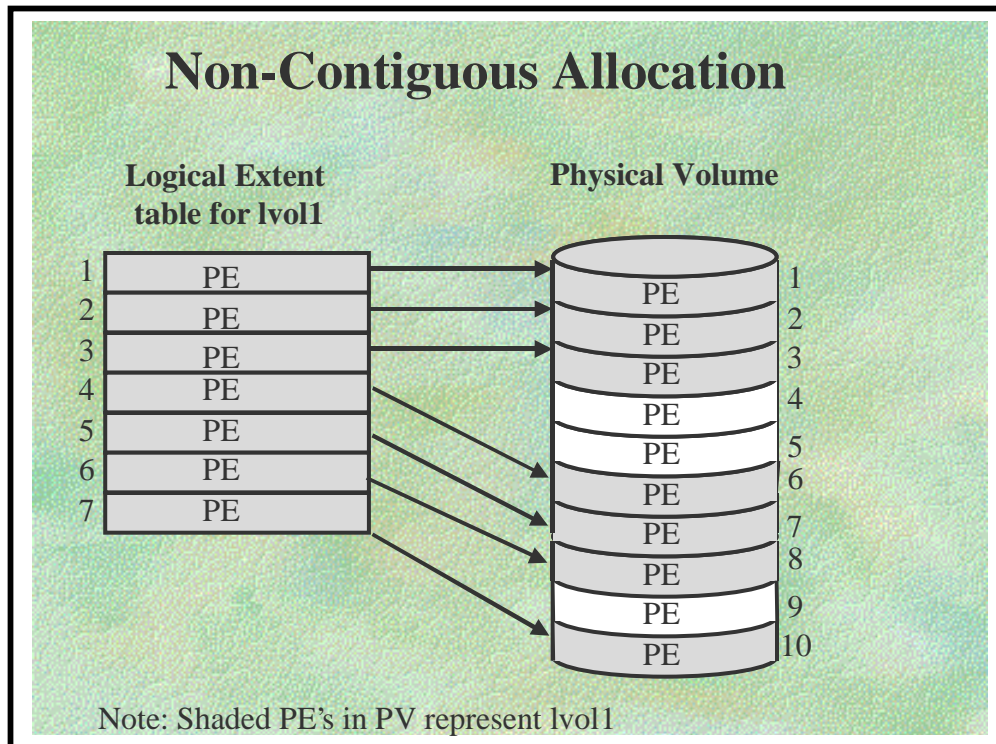
.

Thus, the extent serves as a translation mechanism between the incoming request and underlying device drivers.

By default, LVM selects available physical extents from LVM disks in the order the disks were added to the volume group.



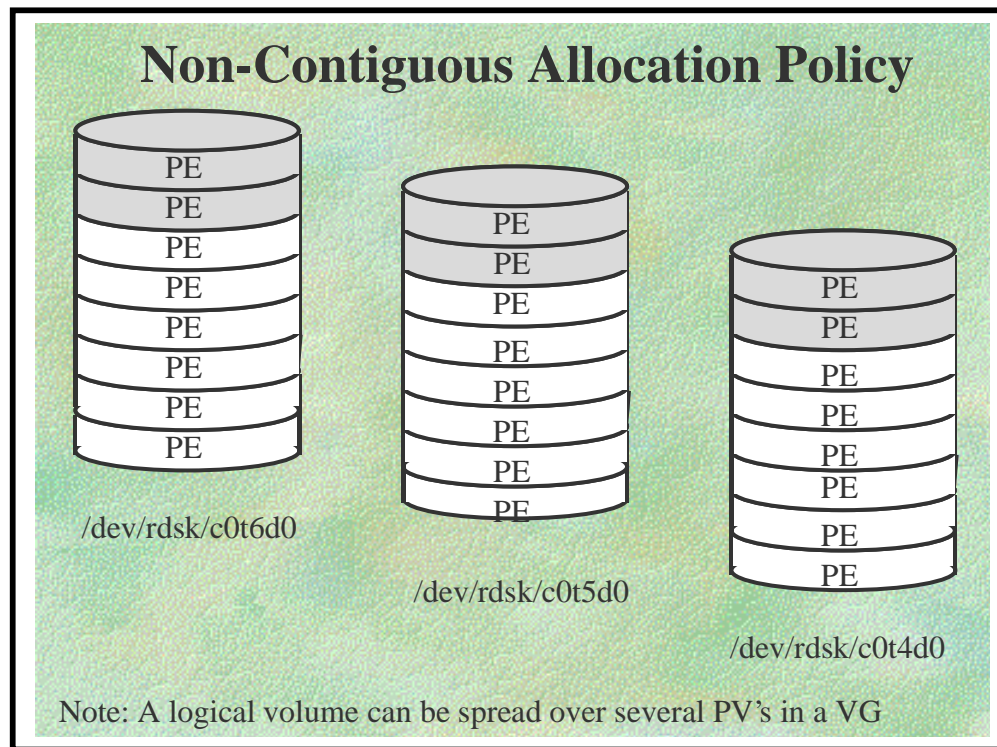
## ◆ 1.1 Concepts



◆

- Non-contiguous disk space allocation means that the logical extents of a logical volume need not be mapped to adjacent physical extents
- This is the default allocation policy.
- Gaps are allowed between allocated Physical Extents (PE's).
- PE's can be taken from other disks in the Volume Group.

## ◆ 1.1 Concepts



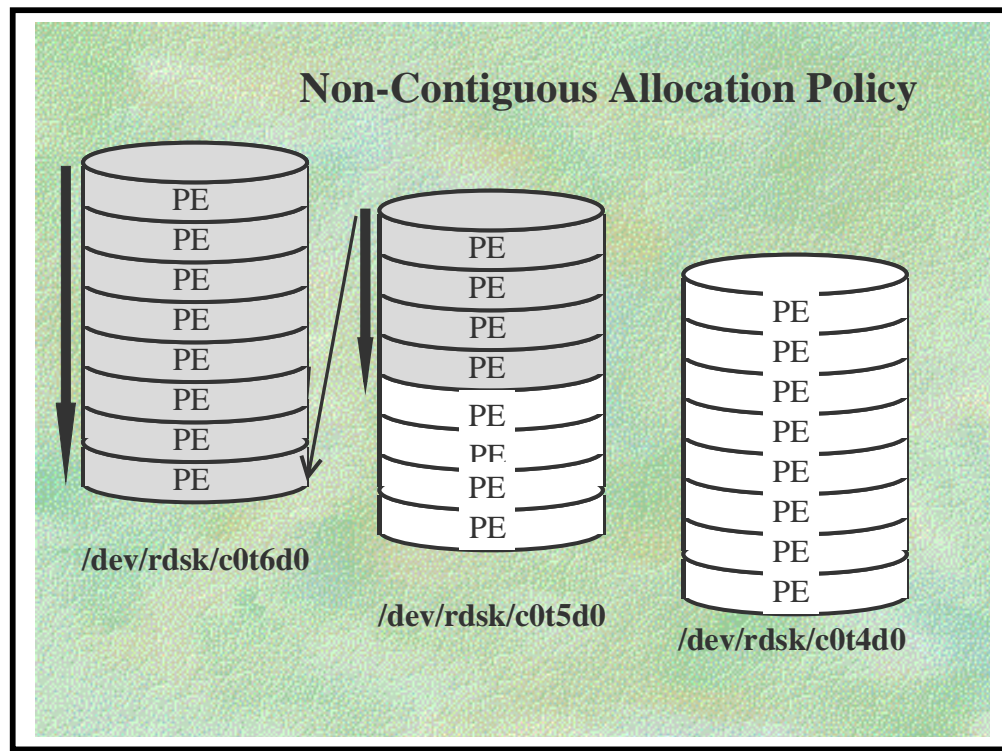
- PE's may be on any disk in the Volume Group.
- PE's may be allocated by chance or selected through use of the *lvextend* command.
- To limit the allocation to specific physical volumes, specify the physical volume names as *pv\_path* arguments or specify the physical volume group names as *pvg\_name* arguments.

### EXAMPLE:

To extend a logical volume onto a particular physical volume:

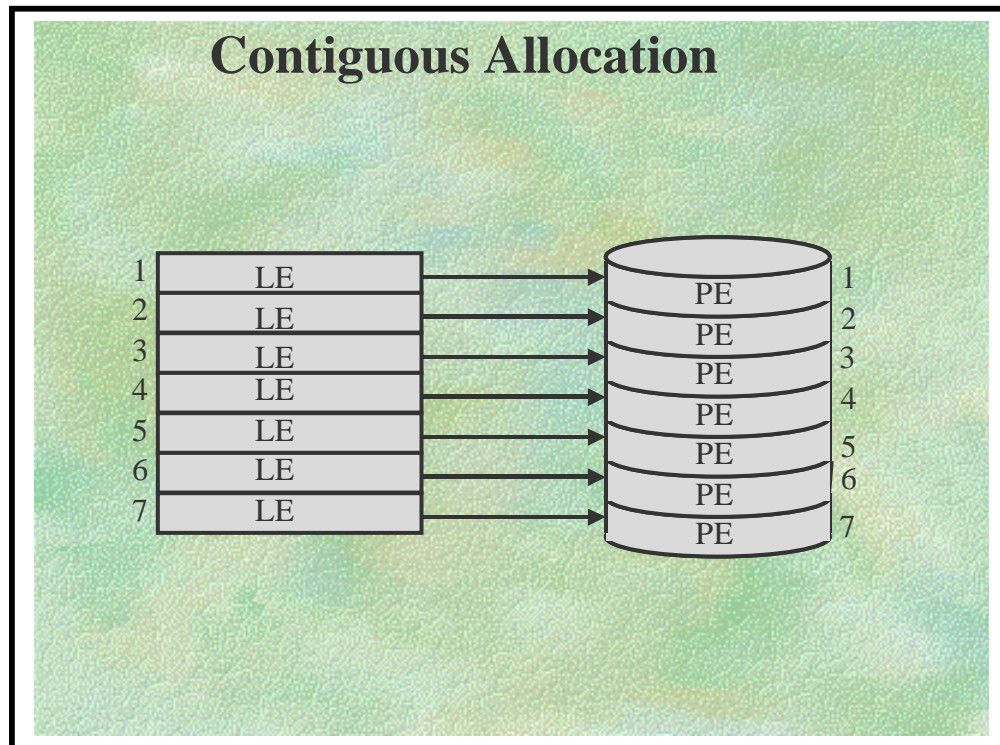
```
# lvextend /dev/vg01/lvol1 /dev/dsk/c0t5d0
```

## ◆ 1.1 Concepts



- LVM allocates the first free PE's from the Volume Group.
  - \* *SAM* and *lvcreate* use this default behavior.
  - \* Some disks may be under-used or even not used at all.
- Use *lvextend* to distribute the Logical Volume more evenly.

## ◆ 1.1 Concepts



- Created using `lvcreate -C y`
- There cannot be any gaps between its allocated PE's.
- The PE's must be allocated in ascending, numerical order.
- A contiguous Logical Volume cannot span Physical volumes.
- Contiguous disk space allocation means that the logical extents (LE) of a logical volume must be mapped to adjacent physical extents (PE) on an LVM disk.

---

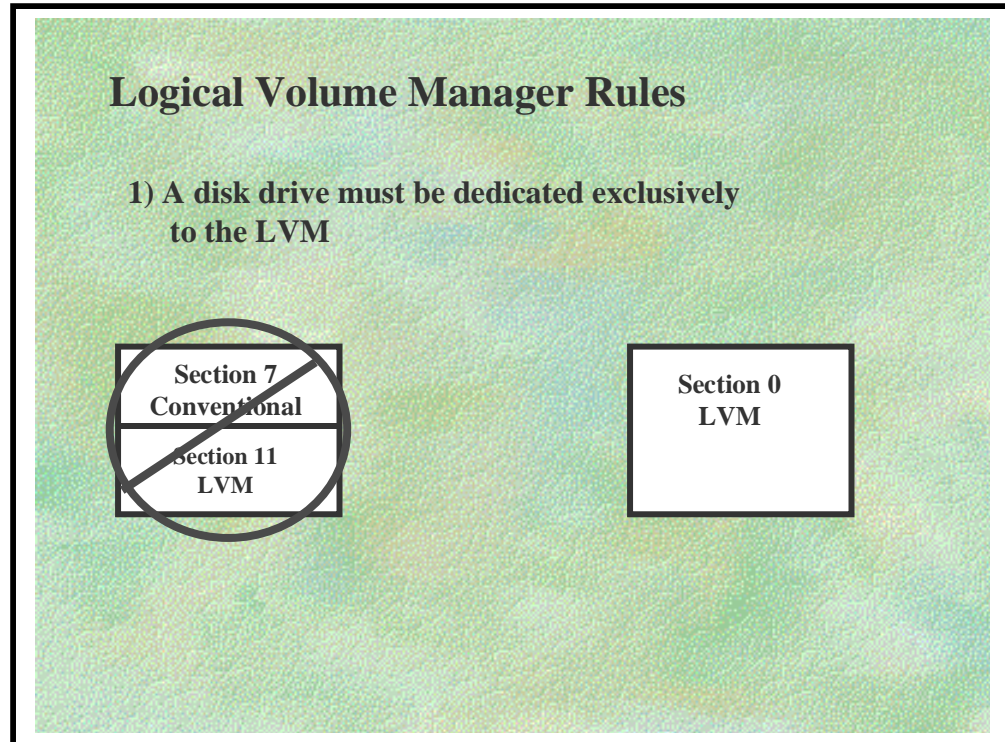
◆ **1.1 Concepts**

---

- Disk space used by the root logical volume must be contiguous. That is, physical extents mapping to the logical volumes of the root file system, primary swap, and dump must each be contiguous. Contiguous extents also adhere to the following requirements:
  - \* Physical extents must be allocated in ascending order.
  - \* No gap may exist between physical extents
  - \* When mirrored, all physical extents of a mirrored copy must reside on the same disk.
- Contiguous allocation is less flexible than non-contiguous allocation and therefore uses disk space less economically. Non-contiguous mapping can result in a logical volume's physical extents being dispersed onto more than one LVM disk, since logical volumes can span multiple disks.

This page is left blank intentionally

◆ 1.2 Rules



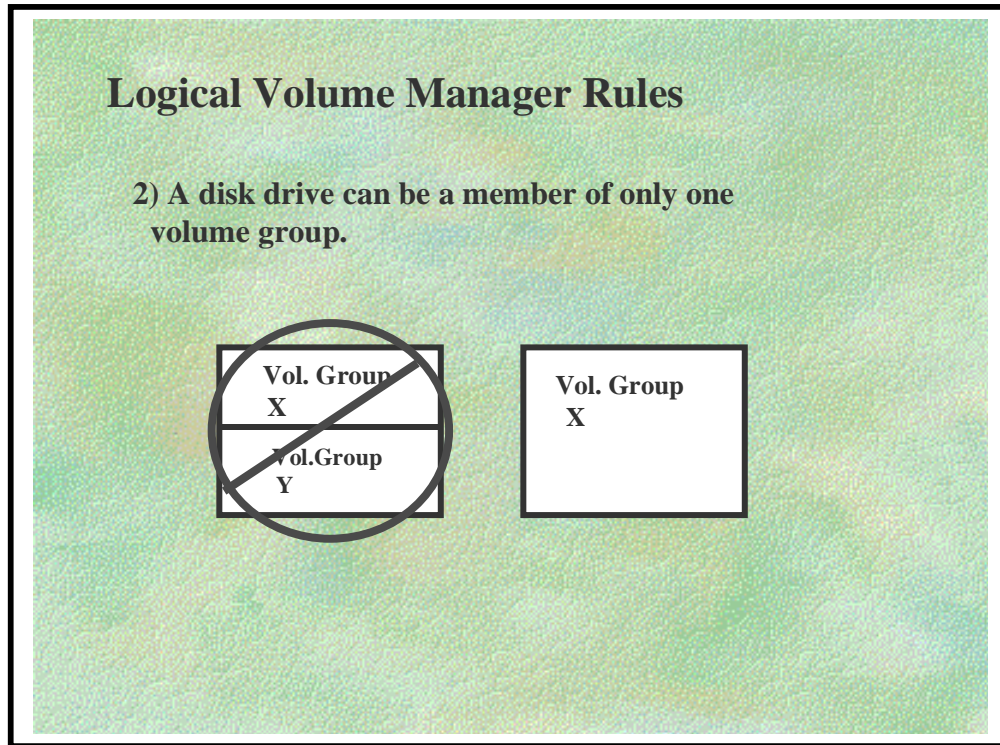
**LOGICAL VOLUME MANAGER RULES**

- A disk drive must be dedicated exclusively to LVM.  
Using any section other than section 0 is not supported.

---

**◆ 1.2 Rules**


---

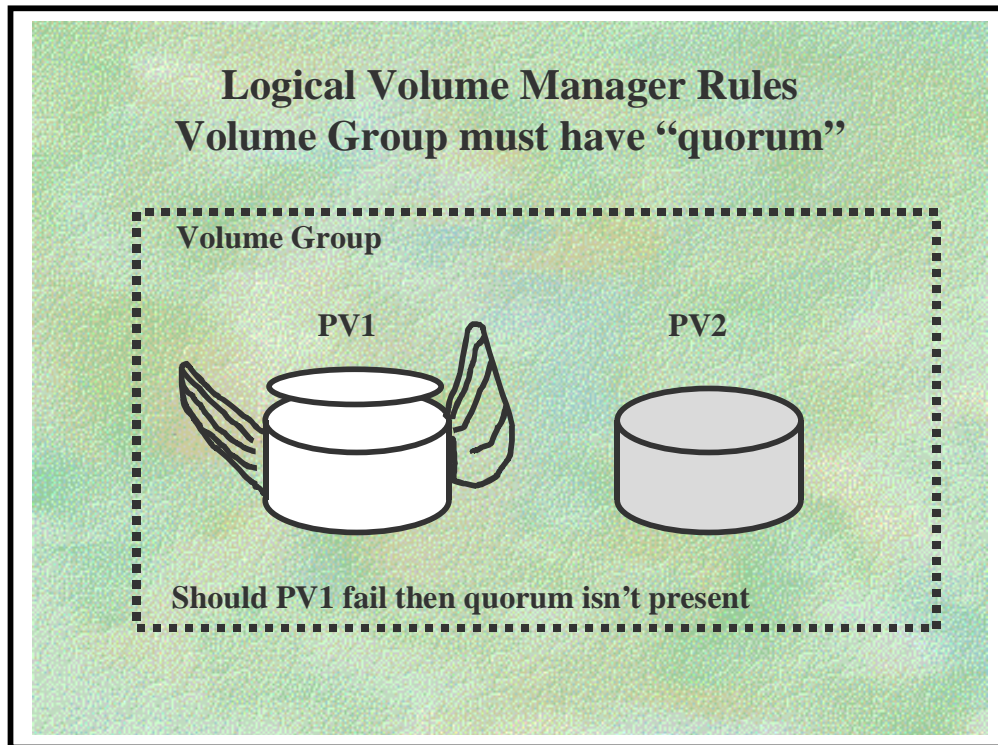

**LOGICAL VOLUME MANAGER RULES (CONTINUED)**

- A disk drive can be a member of only one volume group.
- The maximum number of logical volumes in a volume group is 255.  
(Range is 1 - 255; Default is 255; set using `vgcreate -l`)
- The maximum number of physical extents allowed per physical volume is 65,535.  
(Range is 1 - 65,535; Default is 1016; set using `vgcreate -e`)
- Maximum number of physical volumes per volume group is 255.  
(Range is 1 - 255; Default is 16; set using `vgcreate -p`)

Maximum number of volume groups is set in the kernel by the `maxvgs` parameter to default of 10. In order to change this, a new kernel must be generated with the desired value and can be increased to as high as 256 if necessary.



## ◆ 1.2 Rules

**LOGICAL VOLUME MANAGER RULES (CONTINUED)**

- More than half of the configured LVM disks in a volume group must be present to change or activate that volume group. Quorum is checked both during configuration changes (for example, when creating a logical volume) and at state changes (for example, if a disk fails). If quorum is not maintained, LVM will not acknowledge the change. To override quorum check use *vgchange* with the *-q n* option. If quorum is not met on the root volume group, the system will not boot.

Use "*ISL> hpux -lq*" to boot the system when quorum isn't present.

- Any time a change is made to the root volume group (i.e., root, swap, dump, or file system) the Boot Data Reserved Area (BDRA) must be updated using the *lvlnboot* command. Failure to do this may result in an unbootable system.

This page is left blank intentionally

---

**◆ 1.3 Terminology**


---

**GLOSSARY OF LVM TERMS**

<b>activation</b>	The process by which a volume group, its disks, or its logical volumes are made available for use.
<b>allocation policy</b>	Allocation refers to how data is distributed on a disk. Allocation policy can be contiguous, that is, with no allowable gaps in the set of physical extents on a physical volume, or non-contiguous, with gaps allowed in the set of physical extents.
<b>bad block relocation</b>	An LVM feature, employable at the system administrator's option, that provides for the transparent detection of bad disk sectors and the relocation of data from the bad to good disk sectors. The LVM bad block relocation feature can work in conjunction with disk controllers that perform bad sector relocation at the hardware level. Note that the root logical volume does not support software bad block relocation.
<b>block</b>	A unit of space for data on a disk, typically having a size of 1024-bytes.
<b>device file</b>	Used to identify and communicate with a device (such as a disk) or pseudo device (such as a logical volume). There are block device files (used for transmitting data in multiple-byte blocks) and character device files (used for transmitting data byte-by-byte). Device files are typically stored in the <code>/dev</code> directory.  By default, the full path name of block device files follows a convention for logical volumes: <code>/dev/vg<sub>n</sub>nlvol<sub>n</sub></code> .  By default, the full path name of character device files follows a convention for logical volumes: <code>/dev/vg<sub>n</sub>nr<sub>l</sub>vol<sub>n</sub></code> .
<b>extent</b>	See physical extent, logical extent.

---

**◆ 1.3 Terminology**


---

**GLOSSARY OF LVM TERMS (CONTINUED)**

<b>file system</b>	The organization of files on storage devices. The term file system can refer either to the entire file system tree or to a subsection of that file system, contained within a disk section or a logical volume, that can be mounted or unmounted from the tree.
<b>group file</b>	The character device file for a volume group.
<b>logical extent</b>	A contiguous set of virtual blocks of data contained within a logical volume. A logical extent maps to one (more, if LVM mirroring is used) physical extent. Within a volume group, logical extents are of a fixed size.
<b>logical volume</b>	A logical (rather than physical) structure that is a map of storage areas on physical volumes (disks). A logical volume can be conceptualized as a storage device of flexible size. The data in a logical volume can be mapped to one or more physical volumes.
<b>physical extent</b>	A specific, contiguous set of blocks within a region of a physical volume (disk) where data resides. A physical extent can range, in megabytes, by a power of two from one megabyte to 256 megabytes. Within a volume group, physical extents are of a fixed size.
<b>physical volume</b>	A disk on which LVM has built data structures, making it an LVM disk. The physical device where data is stored. A physical volume is an entire disk, and has, therefore the device files assigned to section two of the disk. It is not possible to use only part of a disk as an LVM disk. The character device file, <code>/dev/rdisk/cxydz</code> , and the block device file, <code>/dev/dsk/cxydz</code> , are the appropriate device files for physical volumes.

---

**◆ 1.3 Terminology**


---

**GLOSSARY OF LVM TERMS (CONTINUED)**

<b>quorum</b>	The number of physical volumes required to be available in order for a volume group to be activated or manipulated after one or more of a volume group's physical volumes becomes unavailable (because of a disk failure or because it contains stale data after a system crash). A volume group without a quorum of physical volumes remains unavailable.
<b>raw disk I/O</b>	The reading and writing of data directly to and from a disk section or a logical volume via the character (raw) device file, independent of file system or swap management.
<b>swap space</b>	Space within a disk section, a logical volume, or a file system that is designated to temporarily store the process image.
<b>volume group</b>	A set of physical volumes (disks) whose space can be combined and logically divided up into logical volumes. Only logical volumes and physical volumes that are a part of a volume group can map together; a physical volume can belong to only one volume group.

---

**◆ 1.3 Terminology**


---

**GLOSSARY OF LVM TERMS (CONTINUED)****Terms used for Mirroring Only**

**NOTE :** The following terms apply to the MirrorDisk/UX product

<b>activation</b>	The process by which a volume group, its disks, or its logical volumes are made available for use.
<b>allocation policy</b>	<p>Allocation refers to how mirrored copies of data are distributed to disks (physical volumes). Allocation policy can be strict; that is, copies cannot reside on the same disk. With non-strict allocation, mirrored copies can reside on the same disk.</p> <p>Allocation policy can be contiguous, that is, with no allowable gaps in the set of physical extents on a physical volume, or non-contiguous, with gaps allowed in the set of physical extents.</p>
<b>logical extent</b>	A set of virtual blocks of data contained within a logical volume. A logical extent maps to one (more, if MirrorDisk/UX is used) physical extent. In the case of MirrorDisk/UX, a logical extent maps to one, two, or three physical extents, depending on whether the stored data is unmirrored, mirrored once, or mirrored twice.
<b>mirrored data</b>	The copies of data stored in physical extents that map to a unique logical extent. Data can be singly mirrored (two physical extents containing replicated data are allocated for each logical extent) or doubly mirrored (three physical extents contain replicated data are allocated for each logical extent).
<b>mirror write cache (MWC)</b>	A MirrorDisk/UX mechanism, whose use is optional, that tracks outstanding mirror write requests and provides a basis for the resynchronization of data blocks after a system crash or power failure.

---

**◆ 1.3 Terminology**


---


**GLOSSARY OF LVM TERMS (CONTINUED)****Terms used for Mirroring Only**

<b>physical volume group</b>	Physical volumes within volume groups can be organized, usually on the basis of the I/O channel or interface adapter to which they are connected, into subgroups known as physical volume groups for the purpose of achieving higher availability of mirrored data. With strict allocation with respect to physical volume groups, mirror copies of a logical volume cannot reside in the same physical volume group.
<b>scheduling policy</b>	Determines whether data is written to mirrors in parallel or sequential manner.
<b>strict allocation</b>	Mirrored copies of data cannot reside on the same disk; strict allocation is only possible when more than one physical volume is available in the volume group. Allocation can be set up to be strict with respect to groupings of disks; physical volumes are usually based on which I/O interfaces card or I/O bus they are connected to.
<b>synchronization</b>	The process of updating "stale" copies of mirrored physical extents.

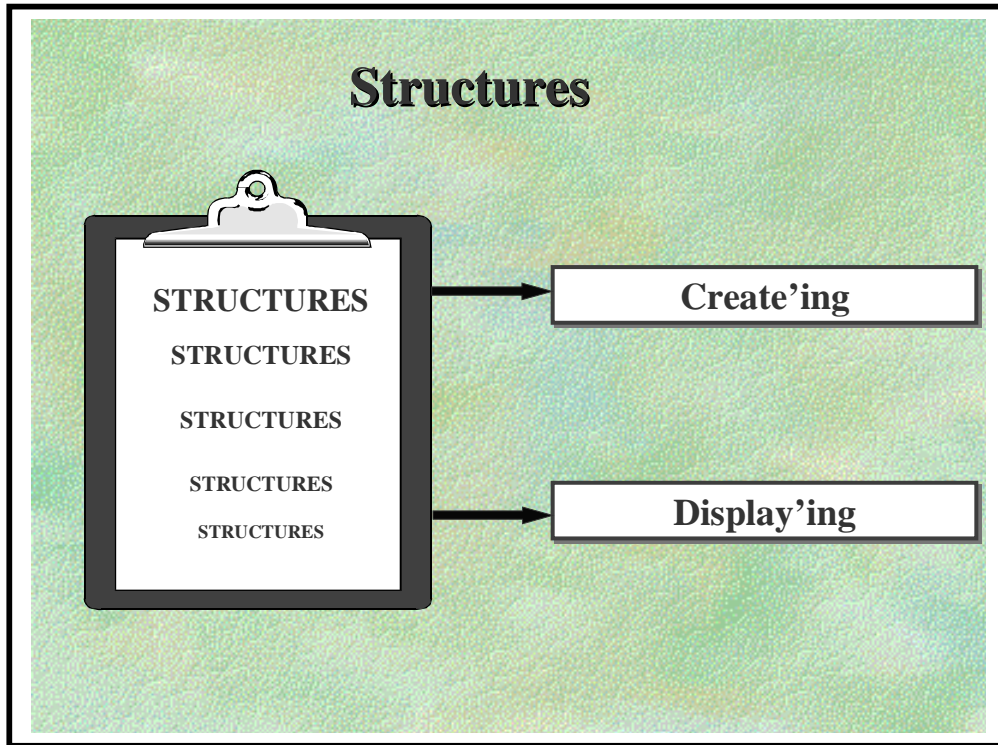
This page is left blank intentionally



◆ 2.0 Structures

<b>The Basics</b>	<b>Structures</b> 	<b>Modifying</b>
<b>Mirrors</b>	<b>Boot Disks</b>	<b>Recovery</b>

◆ 2.0 Structures



**In this module we will look at:**

- Creating LVM structures
- Displaying LVM structures

---

## ◆ 2.1 Create'ing

---

**Creating an HP-UX LVM File System on a new Disk**

1) create physical volume for use as a volume group

```
# pvcreate /dev/rdisk/c0t6d0
```

**NEW LVM DISK**

<b>P V R A</b>	Physical Volume Reserved Area Contents * ID's for volume group and physical volumes * Physical extent size * Physical volume size * Bad block directory * Size and location of the other disk structures
<b>BAD BLOCK POOL</b>	

← Transparent software sparing is not supported on HPIB or the root disk (only used if disk hardware sparing fails)

```
# pvcreate -f /dev/rdisk/cxydz
```

---

**WARNING**            this will destroy any data on the disk

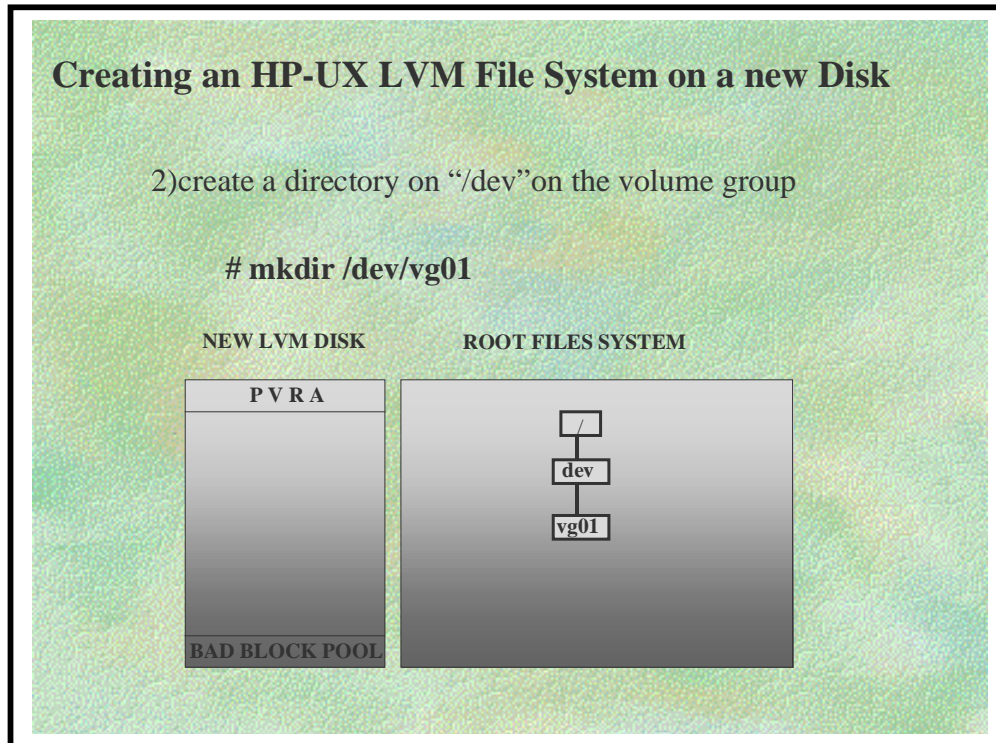
---

- Creates the Physical Volume Reserved Area (PVRA) and the Bad Block Pool
- The PVRA will now contain:
  - \* Unique ID number for the Physical Volume.
  - \* Pointers to the Bad Block Pool
- Use the -f option to force pvcreate to create a new PV on an already existing PV

---

**◆ 2.1 Create'ing**

---

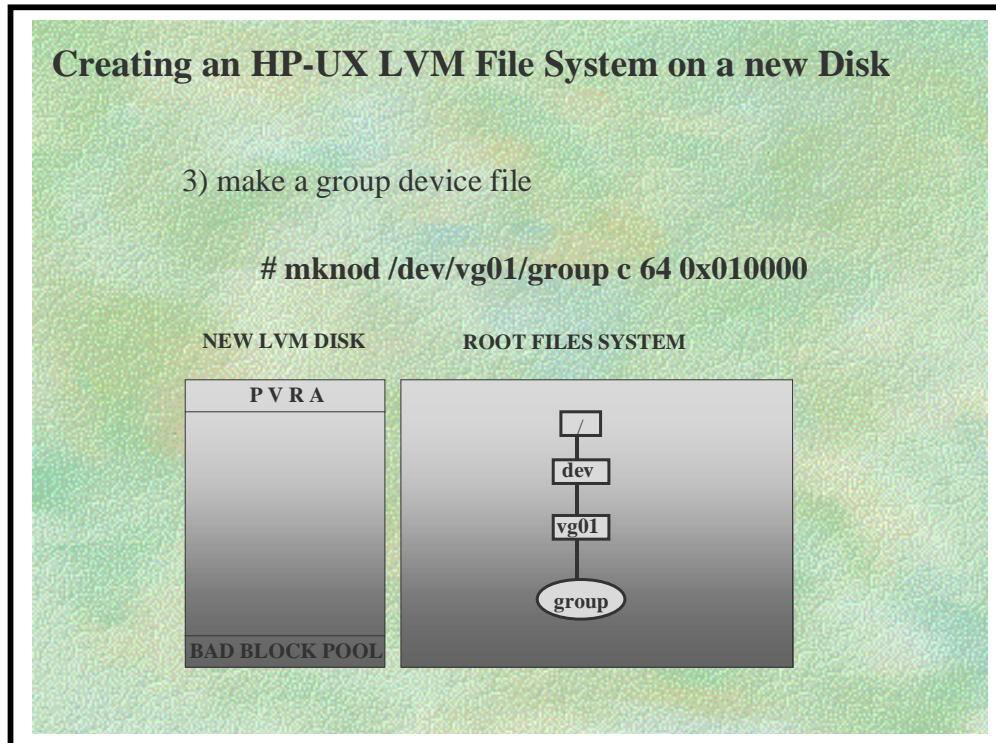


If necessary, create a new directory under /dev for a new volume group to reside on the disk.

```
# mkdir /dev/vg01
```

- The directory name usually begins with vg so it can be easily identified but name can vary.

## ◆ 2.1 Create'ing



If a new volume group is to be created, make a group character device file for the new volume group.

```
# mknod /dev/vg01/group c 64 0xZZ0000
```

Where:

0x

indicates what follows is a hex number

**Key**

**Operation**

ZZ

HEXADECIMAL group number

0000

will always be 0000 for the group file

- This file is used by LVM to access the Volume Group structures and is spelled just as you see it here. Do not remove the group file..

**NOTE:** You can not use the insf or mkfs commands to create the group file

## ◆ 2.1 Create'ing

**Creating an HP-UX LVM File System on a new Disk**

4) create a volume group

```
# vgcreate /dev/vg01 /dev/rdisk/c0t6d0
```

**NEW LVM DISK**

<b>P V R A</b>
<b>V G R A</b>
<b>BAD BLOCK POOL</b>

Volume Group Reserved Area Contents

- \* Volume Group Descriptor Area (VGDA)  
Identifies logical and physical volumes  
Physical to logical extent mapping
- \* Volume Group Status Area (VGSA)  
Physical Volume status (missing/present)  
Physical extent status (stale/ok)
- \* Mirror Consistency Record (MCR)  
Lists disk writes in progress

'vgcreate' creates or updates '/etc/lvmtab'  
Adds volume group information to '/etc/lvmtab'

Create the volume group on the new disk.

```
# vgcreate /dev/vg01 /dev/dsk/c0t6d0
```

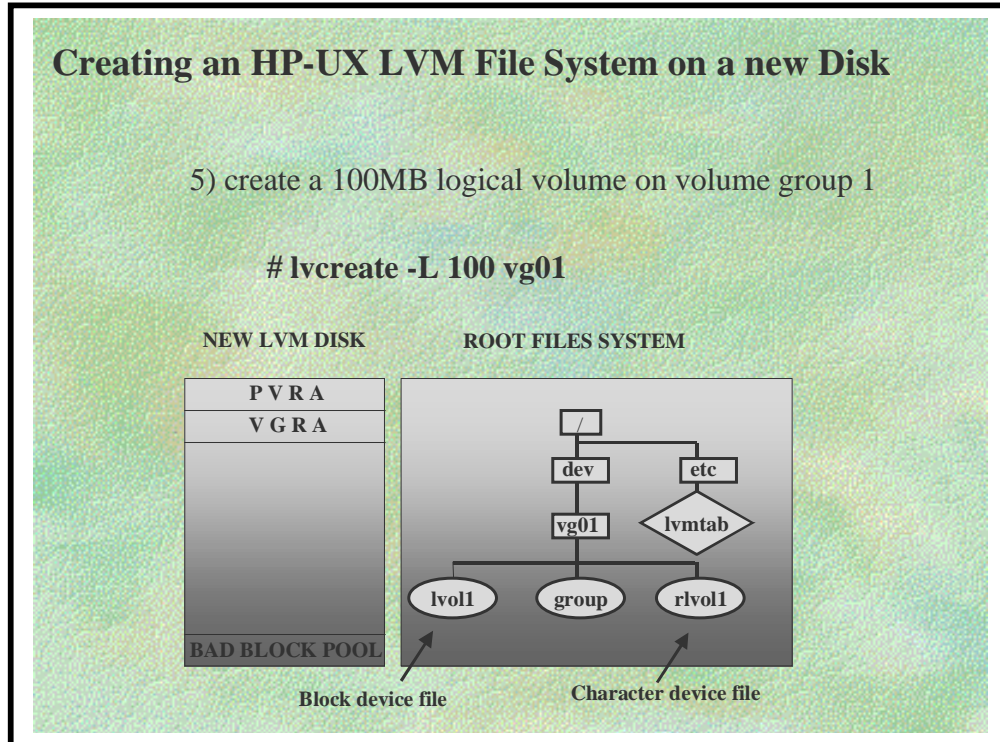
Additional disks may be specified on the command line.

`vgcreate` creates the **Volume Group Reserved Area (VGRA)**.

This contains:

- **Volume Group Descriptor Area (VGDA)**
  - \* Identifies logical and physical volumes
  - \* Logical to Physical Extent mapping
- **Volume Group Status Area (VGSA)**
  - \* Physical Volume status (missing/present)
  - \* Physical Extent status (stale/current)

## ◆ 2.1 Create'ing

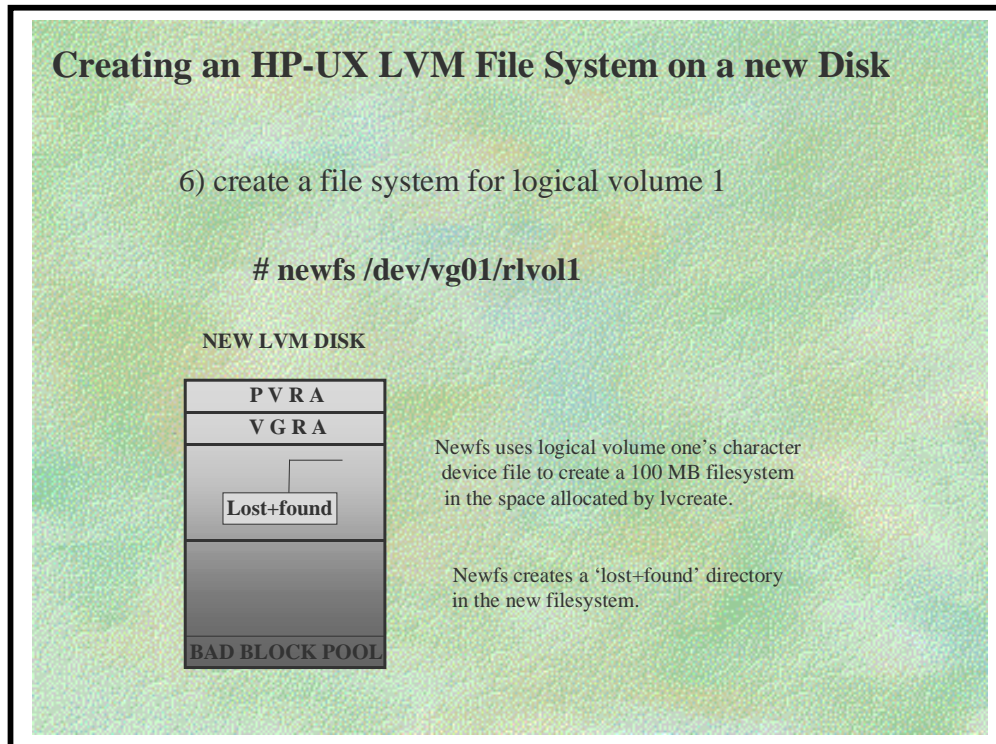


Create a 100-Megabyte logical volume on the new disk.

```
# lvcreate -L 100 vg01
```

- \* Adds Logical Volume information to PVRA and VGRA
- \* Creates device files `/dev/vg01/lvol1` and `/dev/vg01/rlvol1`

## ◆ 2.1 Create'ing



Create the file system for the new logical volume.

```
# newfs /dev/vg01/rlvol1
```

- newfs no longer requires you to specify disk\_type on the command line, and, by default, does not use /etc/disktab to get the geometry parameters. Instead it uses these defaults:

File system size: disk size returned by DIOC\_CAPACITY  
(Minus swap/boot)

Block size: 8192

Fragment size: 1024

Tracks per cylinder: calculated as shown below

Sectors per track: calculated as shown below

RPM: 3600



---

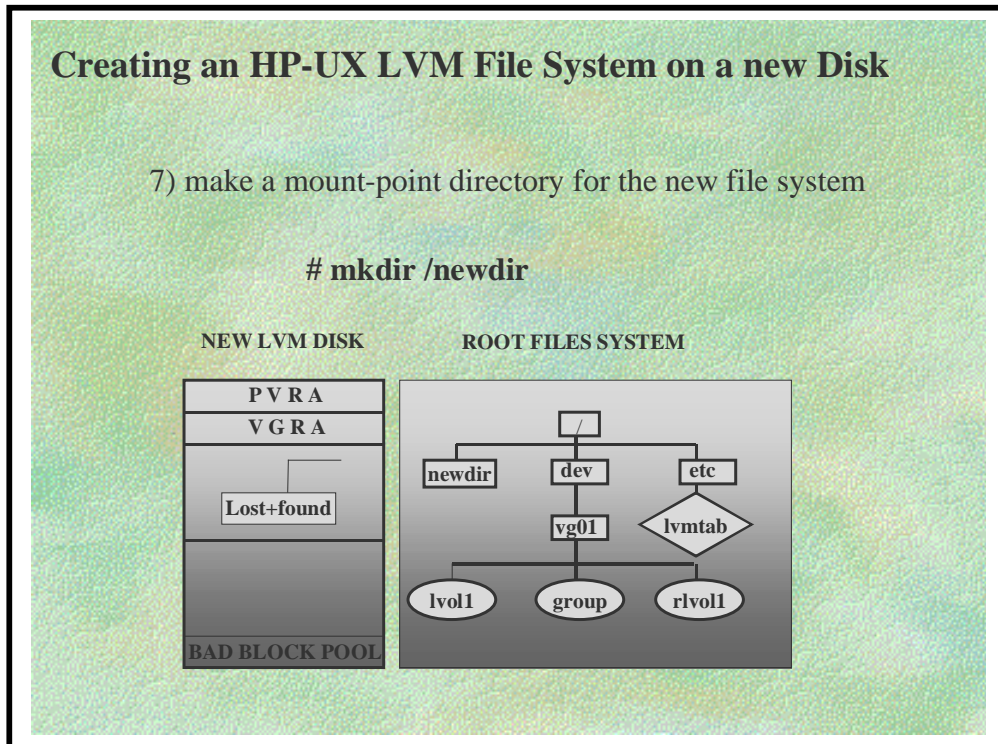
◆ 2.1 Create'ing

---

Tracks per cylinder and sectors per track are calculated according to the size of the file system, as follows:

File System Size	Tracks/Cylinder	Sectors/Track
<= 500 MB	7	22
501 MB - 1 GB	12	28
> 1 GB	16	39

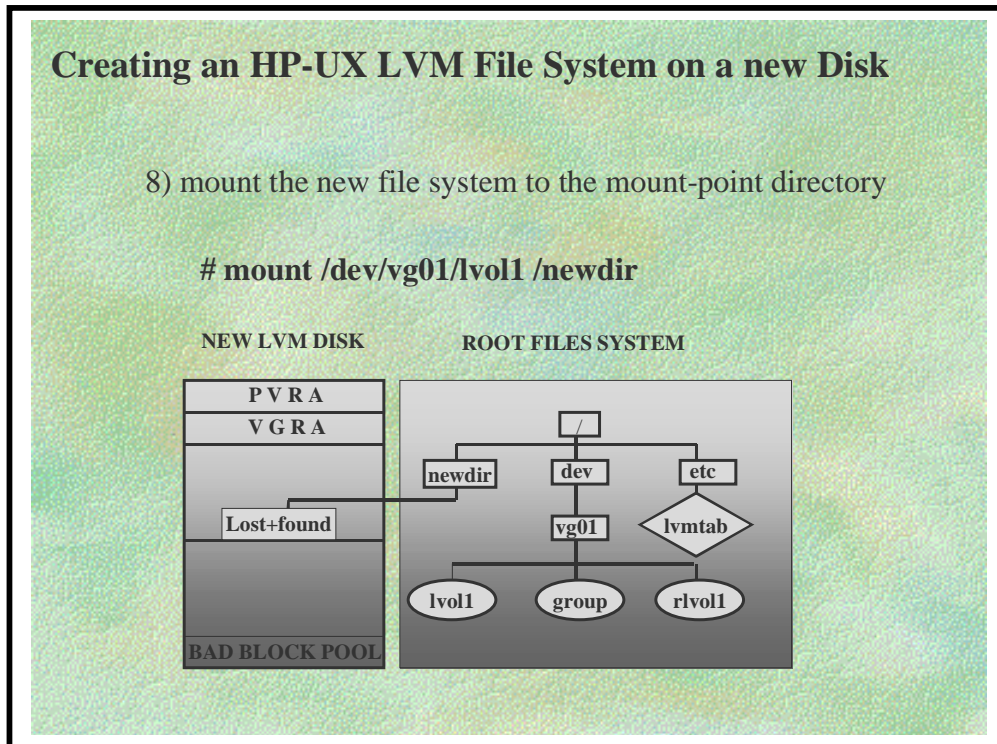
## ◆ 2.1 Create'ing



Make a mount-point-directory on root (/) for the new logical volume.

```
# mkdir /newdir
```

## ◆ 2.1 Create'ing



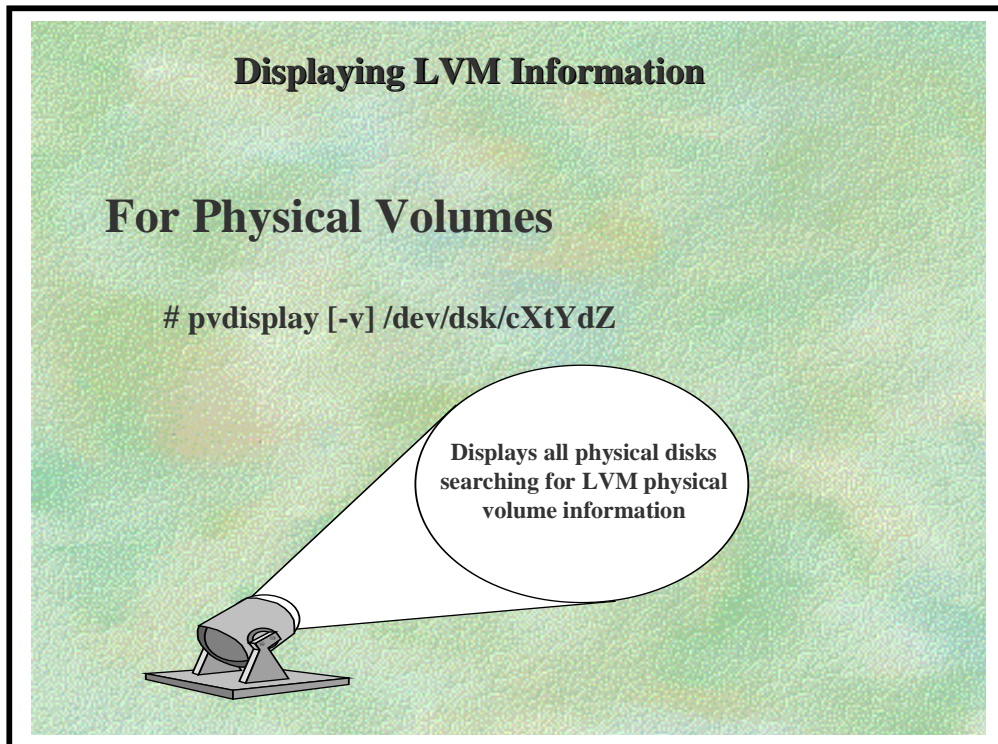
Add a new line in `/etc/fstab` for the new logical volume if this is to be made permanent such as the following sample entry below.

```
/dev/vg01/lvol1 /newdir hfs defaults 0 1
```

---

◆ 2.2 Display'ing LVM Information

---



**FOR PHYSICAL VOLUMES**

```
# pvdisplay [-v] /dev/dsk/cxtydz
```

(Displays all physical disks searching for LVM physical volume information)

---

## ◆ 2.2 Display'ing LVM Information

---

### Example:

```
# pvdisplay /dev/dsk/c0t6d0 /dev/dsk/c0t5d0
```

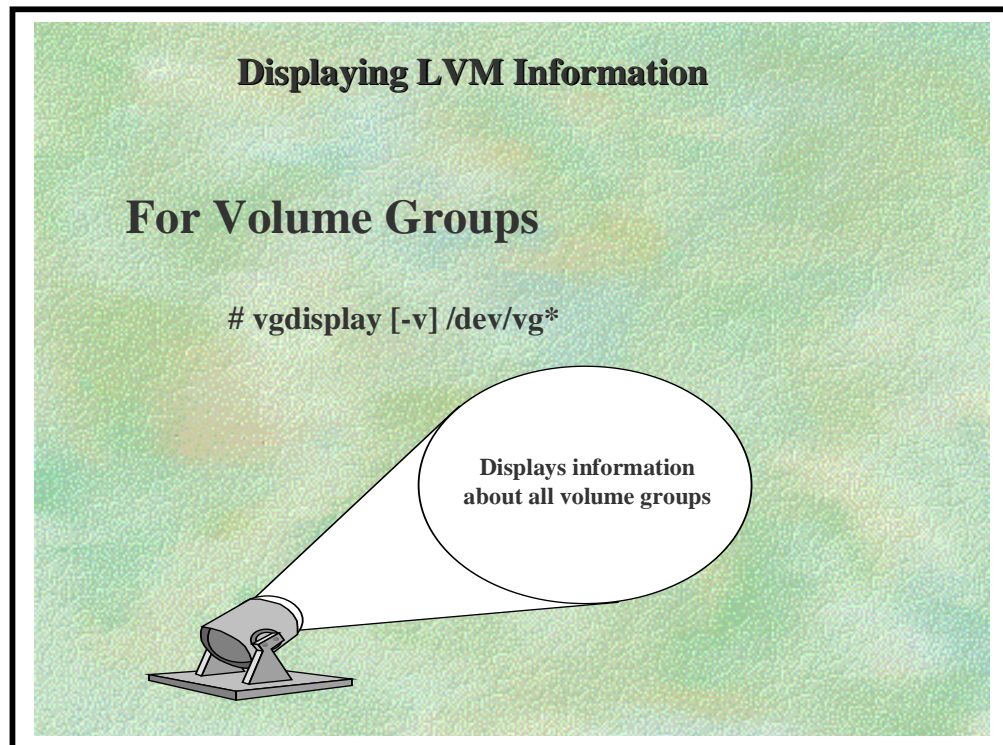
```
--- Physical volumes ---
PV Name           /dev/dsk/c0t6d0
VG Name           /dev/vg01
PV Status         available
Allocatable       yes
VGDA              2
Cur LV           1
PE Size (Mbytes)  8
Total PE          254
Free PE           0
Allocated PE      254
Stale PE          0
IO Timeout (Seconds) default

PV Name           /dev/dsk/c0t5d0
VG Name           /dev/vg01
PV Status         available
Allocatable       yes
VGDA              2
Cur LV           1
PE Size (Mbytes)  8
Total PE          254
Free PE           0
Allocated PE      254
Stale PE          0
IO Timeout (Seconds) default
```

---

**◆ 2.2 Display'ing LVM Information**

---



For Volume Groups

**# vdisplay /dev/vg\***

This command displays a short list of information on all volume groups

**# vdisplay -v /dev/vgroot**

The list contains information about the volume groups, logical volumes, and physical volumes currently configured. Look at the following listing for a small system:

---

**◆ 2.2 Display'ing LVM Information**


---

--- Volume groups ---

VG Name	/dev/vgroot
VG Status	available
Max LV	255 (max. no. of LV's allowed in VG)
Cur LV	4 (current no. of LV's configured)
Open LV	4
Max PV	16 (max. no. of PV's allowed)
Cur PV	2 (current no. of PV's configured)
Act PV	2 (actual no. of PV's activated)
Max PE per PV	1016
VGDA	4
PE Size (Mbytes)	4
Total PE	479 (total no. of PE's in VG)
Alloc PE	352 (total allocated PE's)
Free PE	127 (total no. of free PE's)
Total PVG	0

--- Logical volumes ---

LV Name	/dev/vgroot/lvroot
LV Status	available/syncd
LV Size (Mbytes)	92
Current LE	23
Allocated PE	46
Used PV	2

LV Name	/dev/vgroot/usr
LV Status	available/syncd
LV Size (Mbytes)	152
Current LE	38
Allocated PE	76
Used PV	1

LV Name	/dev/vgroot/swap
LV Status	available/syncd
LV Size (Mbytes)	152
Current LE	38
Allocated PE	38
Used PV	1

---

## ◆ 2.2 Display'ing LVM Information

---

```

LV Name           /dev/vgroot/scaff
LV Status         available/syncd
LV Size (Mbytes)  400
Current LE        100
Allocated PE      100
Used PV           2

```

--- Physical volumes ---

```

PV Name           /dev/dsk/c0t6d0
PV Status         available
Total PE          157
Free PE           0

```

```

PV Name           /dev/dsk/c0t5d0
PV Status         available
Total PE          322
Free PE           127

```

In the example output shown above, you can see:

One volume group: /dev/vgroot

Four logical volumes: /dev/vgroot/lvroot, /dev/vgroot/usr,  
/dev/vgroot/swap, /dev/vgroot/scaff

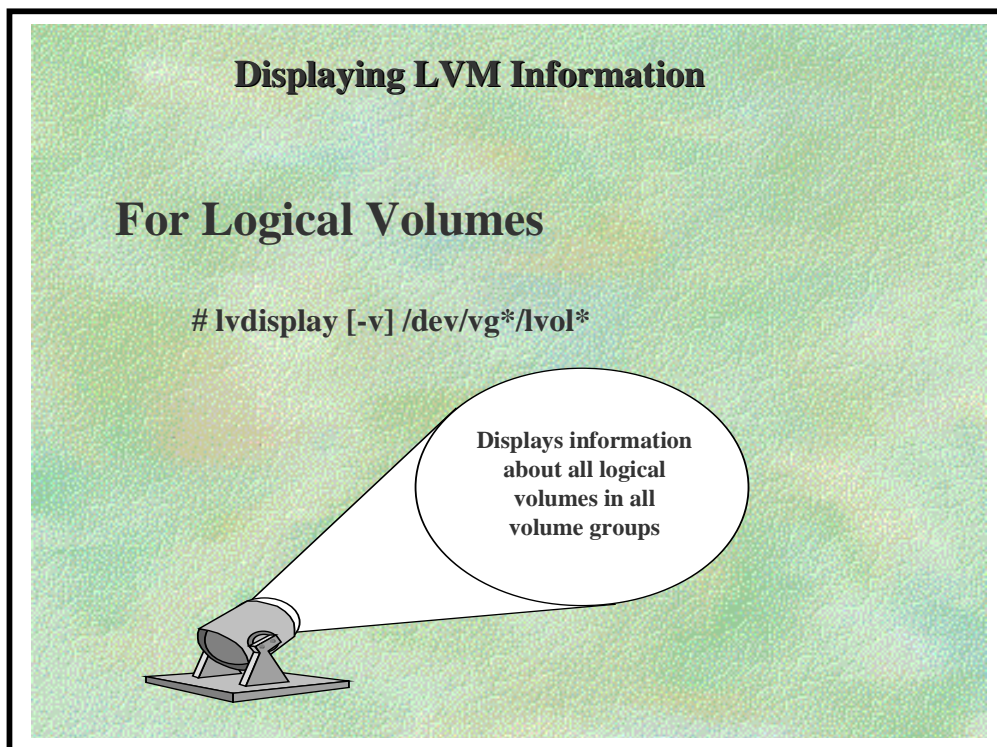
Two disks (physical volumes): /dev/dsk/c0t6d0, /dev/dsk/c0t5d0

The *vgdisplay -v* output also shows information about how large the logical volumes are and about how the data they contain is allocated to the disks in terms of extents

•



## ◆ 2.2 Display'ing LVM Information



For Logical Volumes

`# lvsdisplay [-v] /dev/vg*/lvol*`  
 (Displays all logical volumes in all volume groups)

```
# lvsdisplay /dev/vg01/usr
--- Logical volumes ---
LV Name           /dev/vg01/usr
VG Name           /dev/vg01
LV Permission     read/write
LV Status         available/syncd
Mirror copies     0
Consistency Recovery MWC
Schedule         parallel

LV Size (Mbytes) 100
```

## ◆ 2.2 Display'ing LVM Information

<i>Current LE</i>	<i>25</i>
<i>Allocated PE</i>	<i>25</i>
<i>Bad block</i>	<i>on</i>
<i>Allocation</i>	<i>strict</i>

**NOTE:** If you use the `-v` option with `lvdisplay` to display detailed information about one or more logical volumes it's helpful to pipe the display command to `more`.

```
lvdisplay -v /dev/vg01/usr | more
```

Along with the general information about the logical volume, you can see the distribution of its logical extents on physical volumes, how each logical extent corresponds to each physical extent, and the status of that physical extent, current or stale, if it is a mirror copy.

For example, a verbose display might include something like:

```
# lvdisplay -v /dev/vg01/usr

--- Logical volumes ---
LV Name
.
.
--- Distribution of logical volume ---
PV Name          LE on PV   PE on PV
/dev/dsk/c0t6d0  25         25

--- Logical extents ---
LE  PVI          PE1      Status 1
0000 /dev/dsk/c0t6d0  0028  current
0001 /dev/dsk/c0t6d0  0029  current
0002 /dev/dsk/c0t6d0  0030  current
0003 /dev/dsk/c0t6d0  0031  current
0004 /dev/dsk/c0t6d0  0032  current
0005 /dev/dsk/c0t6d0  0033  current
0006 /dev/dsk/c0t6d0  0034  current
```

## ◆ 2.2 Display'ing LVM Information

In this listing, we can see that there are 25 logical extents in */dev/vg01/usr*. Under the Distribution heading, we can see that there is one disk, */dev/dsk/c0t6d0* (under PV Name), that has the copies of the logical extents.

In the listing "--- Logical extents ---" we can see exactly how the data in the logical volume are distributed to physical extents. Logical extent ("LE") "0000" corresponds to physical extent ("PE1") "0028". If the data had been mirrored, there would be other columns headed "PE2" and "PE3" to show where the other physical copies are located.

To display kernel devices on LVM Bootable Disks

```
# lvinboot -v
(Displays root, primary swap, and dump logical volumes)
```

Example:

```
# lvinboot -v /dev/vg00
```

```

Boot Definitions for Volume Group /dev/vg00:
Physical Volumes belonging in Root Volume Group:
    /dev/dsk/c0t6d0 -- Boot Disk
    /dev/dsk/c0t5d0 -- Boot Disk
    /dev/dsk/c0t4d0
    /dev/dsk/c0t3d0 -- Boot Disk
Root: lvol1      on: /dev/dsk/c0t6d0
                  /dev/dsk/c0t5d0
Swap: lvol2     on: /dev/dsk/c0t6d0
                  /dev/dsk/c0t5d0
Dump: lvol2    on: /dev/dsk/c0t5d0, 0
```

The physical volumes (LVM disks) designated "Boot Disk" are bootable, having been initialized with *mkboot* and *pvcreeate -B*.

Multiple lines for *lvol1* and *lvol2* reveal that the root and swap logical volumes are being mirrored.

---

◆ **2.2 Display'ing LVM Information**

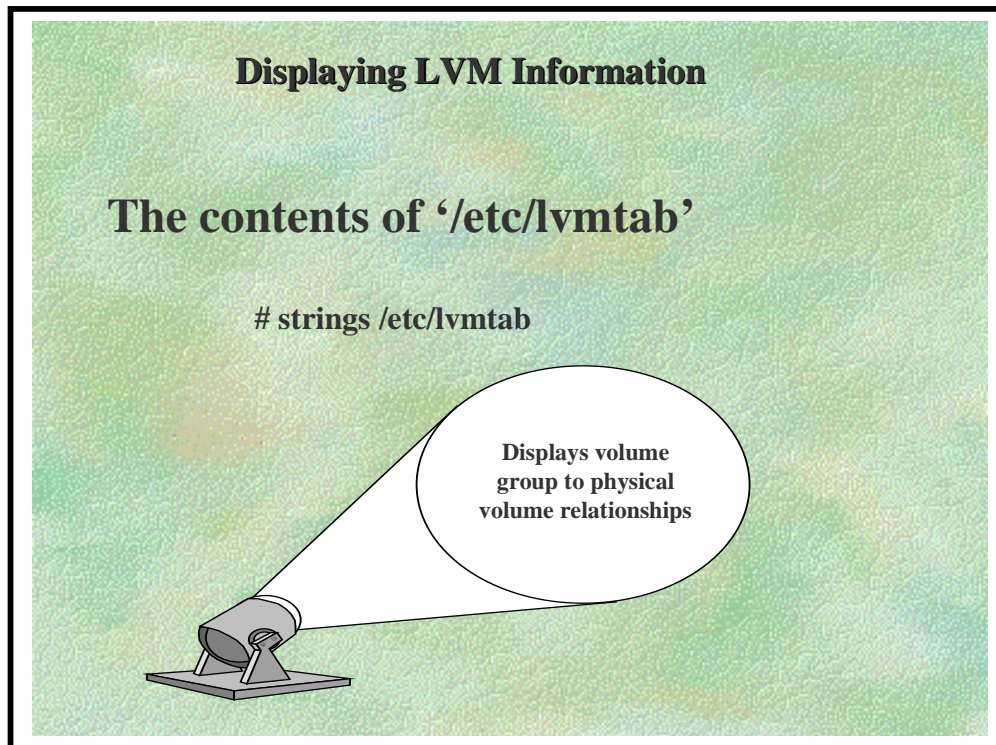
---

Notice that lvol2 is being used for both swap and dump, but that mirroring applies to only swap.

---

**◆ 2.2 Display'ing LVM Information**

---



*strings* - find the printable strings in an object or other binary file

To see the contents of */etc/lvmtab* type:

```
# strings /etc/lvmtab
```

(Displays volume group/physical volume relationships)

```
vg00  
/dev/dsk/c0t6d0  
/dev/dsk/c0t5d0  
vg01  
/dev/dsk/c0t4d0  
/dev/dsk/c0t3d0  
/dev/dsk/c0t2d0  
/dev/dsk/c0t1d0
```

## ◆ 2.2 Display'ing LVM Information

At the heart of the LVM configuration is the */etc/lvmtab* file, which is read by all LVM commands. */etc/lvmtab* is not readable or editable on-screen.

The */etc/lvmtab* file is run-time generated; that is, it is generated the first time you create an LVM entity using SAM or commands such as *vgcreate*, and updated every time you change the LVM configuration. Every configuration update or query reads the */etc/lvmtab* file.

LVM disk file names are recorded in */etc/lvmtab*. If */etc/lvmtab* file is destroyed, all configuration operations involving LVM data structures become impossible.

You can recover the */etc/lvmtab* file using *vgscan*.

To list any swap areas contained in logical volumes, type:

```
# swapinfo
```

	<i>Kb</i>	<i>Kb</i>	<i>Kb</i>	<i>PCT</i>	<i>START/</i>	<i>Kb</i>			
<i>TYPE</i>	<i>AVAIL</i>	<i>USED</i>	<i>FREE</i>	<i>USED</i>	<i>LIMIT</i>	<i>RESERVE</i>	<i>PRI</i>	<i>NAME</i>	
<i>dev</i>	48560	4120	44440	8%	0	-	1	/dev/dsk/c0t5d0	
<i>dev</i>	10240	0	10240	0%	0	-	1	/dev/vg00/lvol1	

You must use the *swapinfo* command to see the swap area since *bdf -b* doesn't show it.

## ◆ 2.2 Display'ing LVM Information

### LVM DEVICE FILES

#### Physical volumes

*/dev/[r]dsk/cxydz*

<b>Key</b>	<b>Operation</b>
[r]	If present, indicates character (raw) access
x	Integer logical unit (lu) number

**NOTE:** The */etc/lssf* command will only list Physical Volume device files. For all LVM device files in */dev/vg\** use *ll*.

#### Volume Group

*/dev/vgXX/group*

<b>Key</b>	<b>Operation</b>
XX	Volume group number (00...255)
group	Must be called group

#### Logical Volume

*/dev/vgXX/[r]lvolY*

<b>Key</b>	<b>Operation</b>
XX	Integer volume group number (0...255)
[r]	If present, indicates character (raw) access
Y	Integer logical volume number (1...255)

**NOTE:** For logical volumes which contain raw data, a name is recommended because this helps to quickly identify the contents. Its name can be specified with *lvcreate -n* and must not exceed 13 characters.

For example:

*/dev/vgdatabase/rlvaccounts*

---

**◆ 2.2 Display'ing LVM Information**


---

**LVM DEVICE FILES (CONTINUED)****Example:**

```
# ll /dev/vg02
```

```
crw-rw-rw-    1 root root    64 0x020000 Sep 21 10:59 group
brw-r-----    1 root root    64 0x020001 Sep 21 10:59 lvoll
crw-r-----    1 root root    64 0x020001 Sep 21 10:59 rlvoll
```

<b>Key</b>	<b>Operation</b>
64	Major number always 64
02	HEXADECIMAL volume group number
0000	always zeroes for group
01	HEXADECIMAL logical volume number



---

**◆ 2.2 Display'ing LVM Information**

---

**LVM COMMANDS - QUICK REFERENCE****Physical Volume Commands**

- pvcreate** Makes a disk an LVM disk (a physical volume).
- pvdisplay** Displays information about physical volumes in a volume group.
- pvchange** Sets physical volume characteristics to allow or deny allocation of additional physical extents from this disk.
- pvmove** Moves allocated physical extents from source to destination within a volume group.

**Volume Group Commands**

- vgcreate** Creates a volume group.
- vgdisplay** Displays information about volume groups.
- vgchange** Activates or deactivates one or more volume groups. Allows a volume group to mount with or without a quorum.
- vgextend** Extends a volume group by adding disks to it.
- vgreduce** Reduces a volume group by removing one or more disks from it.
- vgscan** Scans all disks and looks for logical volume groups.
- vgsync** Synchronizes mirrors that are stale in one or more logical volumes.
- vgremove** Removes definition(s) of volume group(s) from the system.
- vgexport** Removes a volume group from the system without modifying the information found on the physical volume(s).
- vgimport** Adds a volume group to the system by scanning physical volumes which have been exported using **vgexport**.
- Vgcfgbackup** Saves the configuration information for a volume group.  
Remember that a volume group is made up of one or more physical volumes group is made up of one or more physical volumes.
- vgcfgrestore** Restores the configuration information for a volume group.

**Logical Volume Commands**

- lvcreate** Creates a logical volume.
- lvdisplay** Displays information about logical volumes.

---


**◆ 2.2 Displaying LVM Information**

---

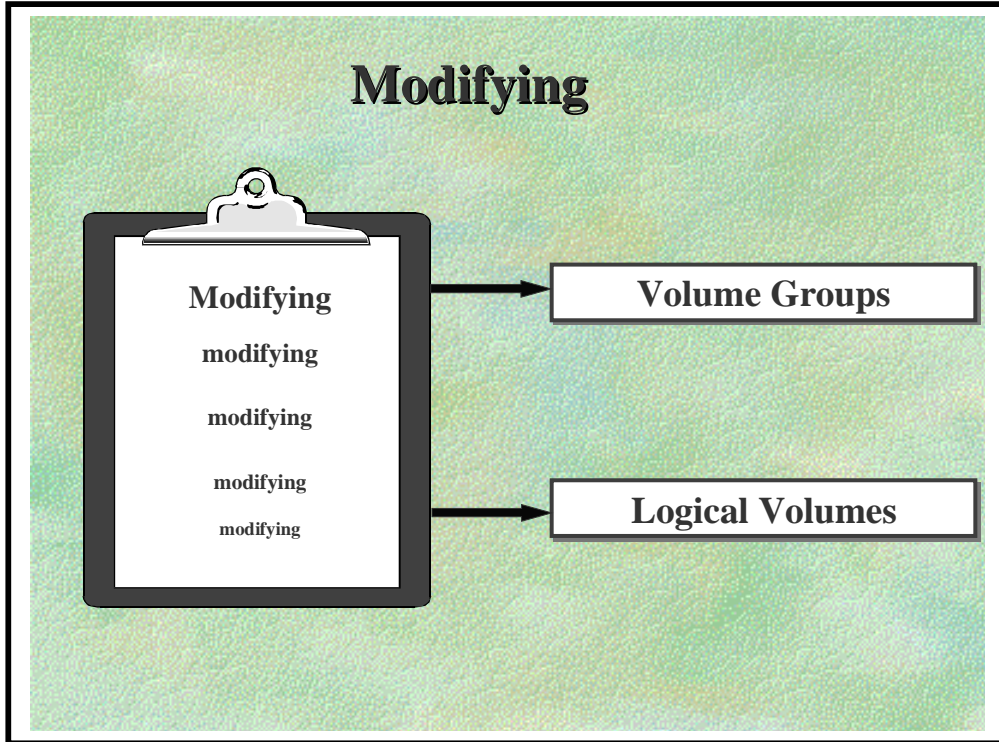
**LVM COMMANDS - QUICK REFERENCE**

lvchange	Changes characteristics of logical volume including availability, scheduling policy, permissions, block relocation policy, allocation policy, mirror cache availability.
lvextend	Increases disk space allocated to a logical volume.
extendfs	Extends the size of a filesystem residing on a logical volume.
lvreduce	Decreases disk space allocated to a logical volume.
lvremove	Removes one or more logical volumes from a volume group.
lvsplit	Splits a mirrored logical volume into two logical volumes.
lvmerge	Merges the lvsplit logical volumes into one logical volume.
lvsync	Synchronizes mirrors that are stale in one or more logical volumes.
lvmmigrate	Prepares a root file system for migration from partition to a logical volume.
lvlnboot	Used to set up a logical volume to be a root, primary, swap, .
lvrmboot	Use this if you don't want a logical volume to be a root, primary, swap, or dump volume.

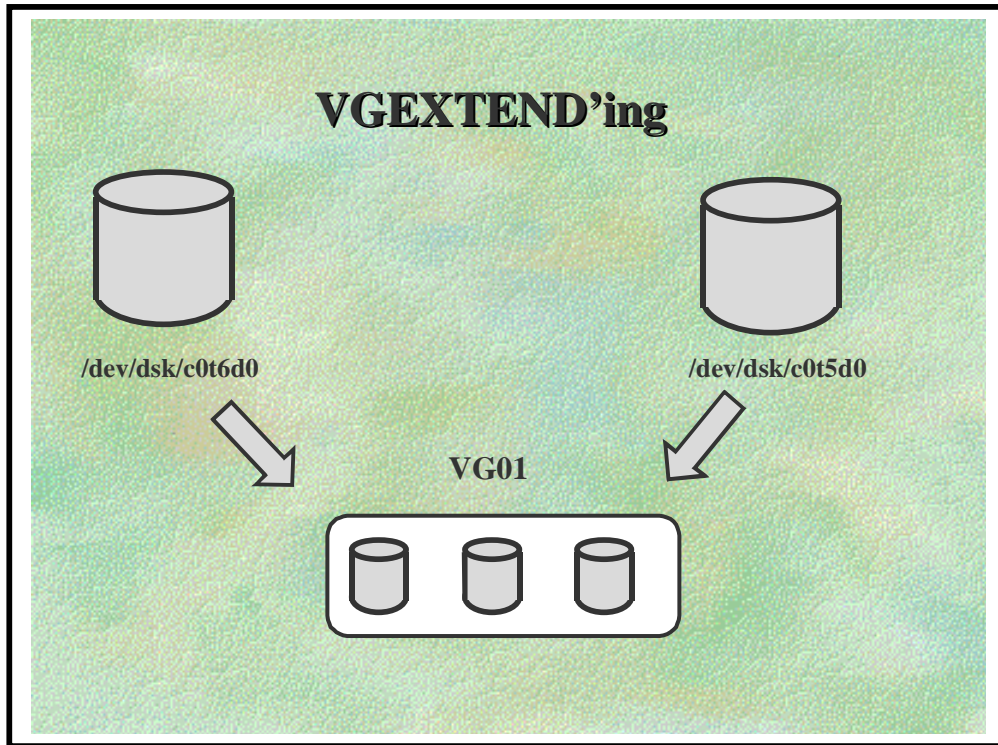
◆ 3.0 MODIFYING

<b>The Basics</b>	<b>Structures</b>	<b>Modifying</b> 
<b>Mirrors</b>	<b>Boot Disks</b>	<b>Recovery</b>

◆ 3.0 MODIFYING



### ◆ 3.1 Volume Group



Suppose you have an existing volume group, `/dev/vg01`, and you want to add two disks to it. You have installed the disks, Instance 6 and 5.

#### LVEXTEND EXAMPLE

1. Make the disks LVM disks (physical volumes); remember to use the disk's character device file.

```
# pvcreate /dev/rdisk/c0t6d0
# pvcreate /dev/rdisk/c0t5d0
```

2. Add the LVM disks to the volume group `/dev/vg01` (remember to use the disk's block device file).

```
# vextend /dev/vg01 /dev/dsk/c0t6d0 /dev/dsk/c0t5d0
```

---

### ◆ 3.1 Volume Group

---

3. To verify that the volume group now contains the disks, use the command:

```
# vgdisplay /dev/vg01
```

You can see the disks under the heading:

```
--- Physical Volumes ---.
```

### VGREDUCE

You can also remove disks from a volume group using the `vgreduce(1M)` command, but you must first move or remove any logical volumes on the physical volume.

---

**CAUTION** Whenever you add or remove disks from the root volume group (the volume group that contains `/`), you must run *the* `lvlnboot(1M)` command to update the boot data stored on the boot disks in the volume group if it has been set to not run automatically.

---

### VGREMOVE

If the volume group is to be removed, use the `vgremove` command. `Vgremove` will only remove a volume group after it has been `vgreduce`'d down to one physical volume. This will be seen if you do a `vgdisplay` and check these two lines:

```

      •
      •

Cur PV          1
Act PV          1

      •
      •

```

---

**◆ 3.1 Volume Group**

---

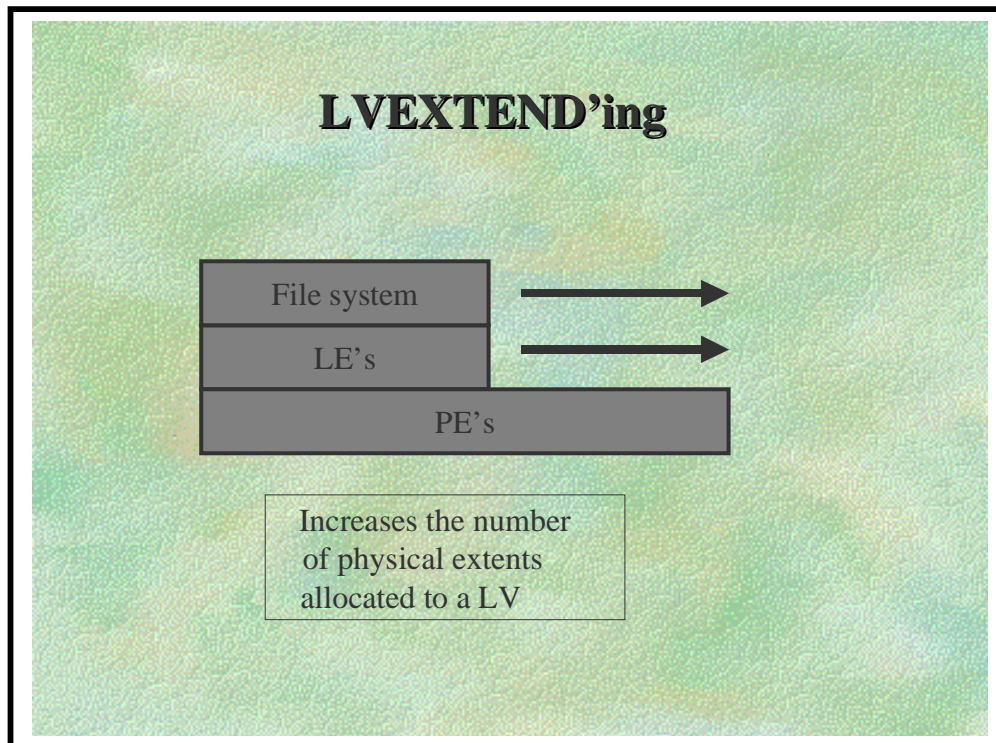
If the values are different from each other then you will not be able to remove the volume group. If the Cur PV value is higher but the volume group only shows that it has one physical volume in both */etc/lvmtab* and

a *vgdisplay -v* listing then you will never be able to *vgremove* it until these missing physical volumes are *vgcfgrestore*'d. If you can't do this then the only option is to do a *vgexport* of the volume group - this is highly effective!

This page is left blank intentionally



### ◆ 3.2 Logical Volume



#### INCREASING THE SIZE OF A LOGICAL VOLUME

You can increase the size of a logical volume by using the *lvextend(1M)* command. You can also specify that you want the increased disk space (either in terms of extents or megabytes) allocated to a specific disk (physical volume), or you can let LVM determine where to allocate it.

#### **Example:**

Suppose you want to increase a logical volume, */dev/vg01/lvol4*, which is currently 100 megabytes, to 200 megabytes. (If it contains a file system, make sure the file system is unmounted. An example of extending a logical volume and extending a file system in it follows.)

```
# lvextend -L 200 /dev/vg01/lvol4
```

Note that you specify the new size rather than the amount of increase.

---

## ◆ 3.2 Logical Volume

---

When LVM implements the new size, it will use whole extent sizes.

For example, if the extent size was four megabytes and you specified `-L 198`, LVM will still extend the logical volume to 200 megabytes.

You can also increase the size of a logical volume by increasing the number of its logical extents.

Using the previous example where we wanted to extend our logical volume to 200 megabytes we would now enter the number of logical extents required:

```
# lvextend -l 50 /dev/vg01/lvol4
```

Note that when specifying extents, you use the `-l` option (lowercase L) rather than `-L`, which is for specifying size in megabytes.

### **Extending a Logical Volume to a Specific Disk.**

Suppose you have several disks in a volume group and two of them are identical models. You want to extend a 275 megabyte logical volume that resides on one of the disks to 400 megabytes. Also, you want to make sure the increase is allocated to the other identical disk.

To extend a logical volume (e.g., `/dev/vg01/lvol4`) to a specific disk (e.g., `/dev/dsk/c0t3d0`), explicitly specify the disk's block device file.

```
# lvextend -L 400 /dev/vg01/lvol4 /dev/dsk/c0t3d0
```

---

## ◆ 3.2 Logical Volume

---

### Extending The File System

Use the *extendfs(1M)* command to increase the file system capacity in proportion to the increase in the logical volume. *extendfs* reads the current superblock to find out the current characteristics of the filesystem. It then uses this information to create the additional cylinder groups

required to fill the logical volume. Once this has completed it updates the superblock with the new information. Note that *extendfs(1M)* requires the use of the character device file.

```
# extendfs /dev/vg01/rlvol4
```

The results of file system extension are reported.

Mount the file system again; specify the mount point, */projects*, for example.

```
# mount /dev/vg01/lvol4 /projects
```

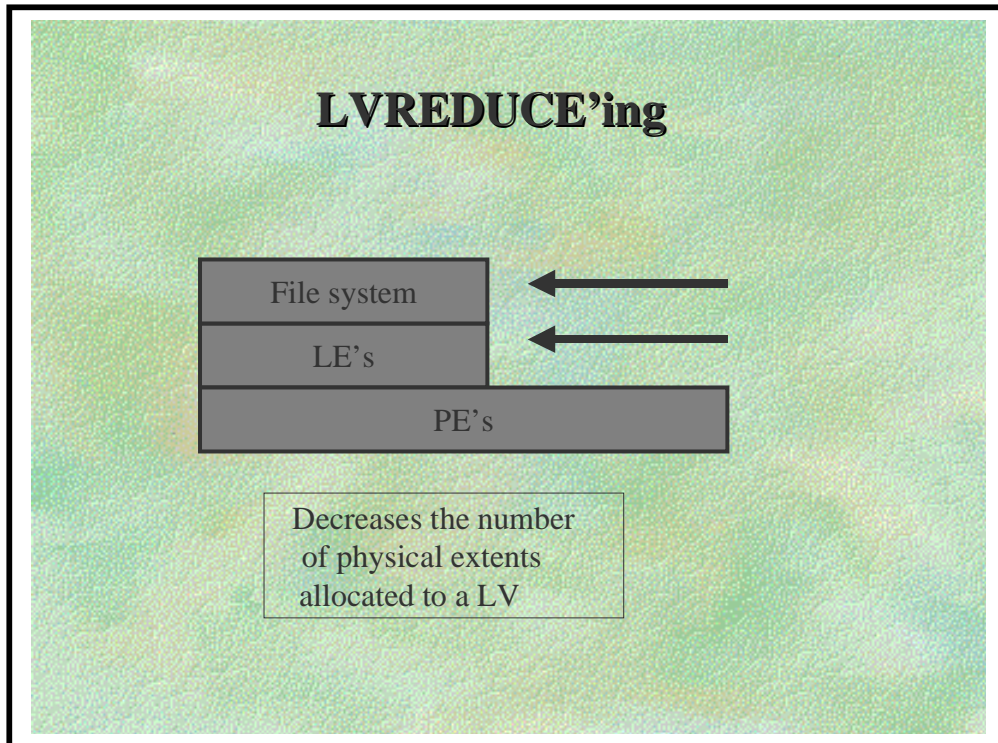
Run *bdf* and note the increased capacity.

NOTE: The filesystem must be unmounted before *extendfs* is run

---

**◆ 3.2 Logical Volume**

---



You can reduce the size of a logical volume by using the *lvreduce(1M)*

### **REDUCING THE SIZE OF A LOGICAL VOLUME**

command.

Reducing the size of a logical volume is appropriate when you want to use the logical volume for another purpose that requires less space.

#### **CAUTION:**

- When you reduce the size of a logical volume, you might lose data as LVM deallocates disk space.
- Reduce the size of a logical volume only if you have safely backed up the contents to tape or to another logical volume, or if you no longer need its current contents.
- You cannot “reduce” the size of a file system: use *newfs*

---

## ◆ 3.2 Logical Volume

---

### Example:

#### **Reducing the Size of a Logical Volume**

Suppose you have a logical volume of 80 megabytes. You no longer need the current data in the logical volume, but you would like to use it for another purpose that requires only 40 megabytes.

```
# lvreduce -L 40 /dev/vg03/lvol4
```

When you issue the command, LVM asks for confirmation. If you answer y to proceed, you get a confirming message:

```
Logical volume "/dev/vg03/lvol4" has been successfully reduced.
```

#### **Moving Data in Logical Volumes From Disk to Disk**

You can use the `pvmove(1M)` command to move data contained in logical volumes from one disk to another disk within a volume group.

For example, you can move a logical volume's data from one disk to another to use the space on the first disk for some other purpose.

Also, you can move all data from one disk to another. You might want to do this, for example, so you can remove a disk from a volume group. After removing the logical volume data off a disk, you can then remove the disk from the volume group.

### Example:

#### **Moving Logical Volume Data from One Disk to Another**

Suppose you want to move the data in a logical volume, `/dev/vg01/markets`, from the disk `/dev/dsk/c0t3d0` to the disk `/dev/dsk/c0t4d0`.

Note that when you issue the command, you must specify a specific logical volume on the source disk with the `-n` flag. You must also specify the source disk first in the command line. For example,

```
# pvmove -n /dev/vg01/markets /dev/dsk/c0t3d0 /dev/dsk/c0t4d0
```

---

## ◆ 3.2 Logical Volume

---

### Example:

#### Moving All Data on One LVM Disk to Another

If you want to move all the data on a given disk within a volume group, you can use the *pvmove*(1M) command to move the data to other specific disks or let LVM move the data to available space within the volume group.

To move data off disk */dev/dsk/c0t4d0* and let LVM transfer the data to available space in the volume group, you could enter:

```
# pvmove /dev/dsk/c0t4d0
```

To move data off disk */dev/dsk/c0t4d0* to the destination disk */dev/dsk/c0t5d0*, enter:

```
# pvmove /dev/dsk/c0t4d0 /dev/dsk/c0t5d0
```

If space doesn't exist on the destination disk, the *pvmove* command will not succeed.

#### Removing Logical Volumes

You can remove a logical volume with the *lvremove*(1M) command. For example, to remove an empty logical volume,

```
# lvremove /dev/vg01/lvol2
```

If the logical volume contains data, LVM prompts you for confirmation. You can use the *-f* flag to remove the logical volume without a confirmation request.

**End of Day 1 Reading.**

**Perform Lab Modules 1 - 9.**






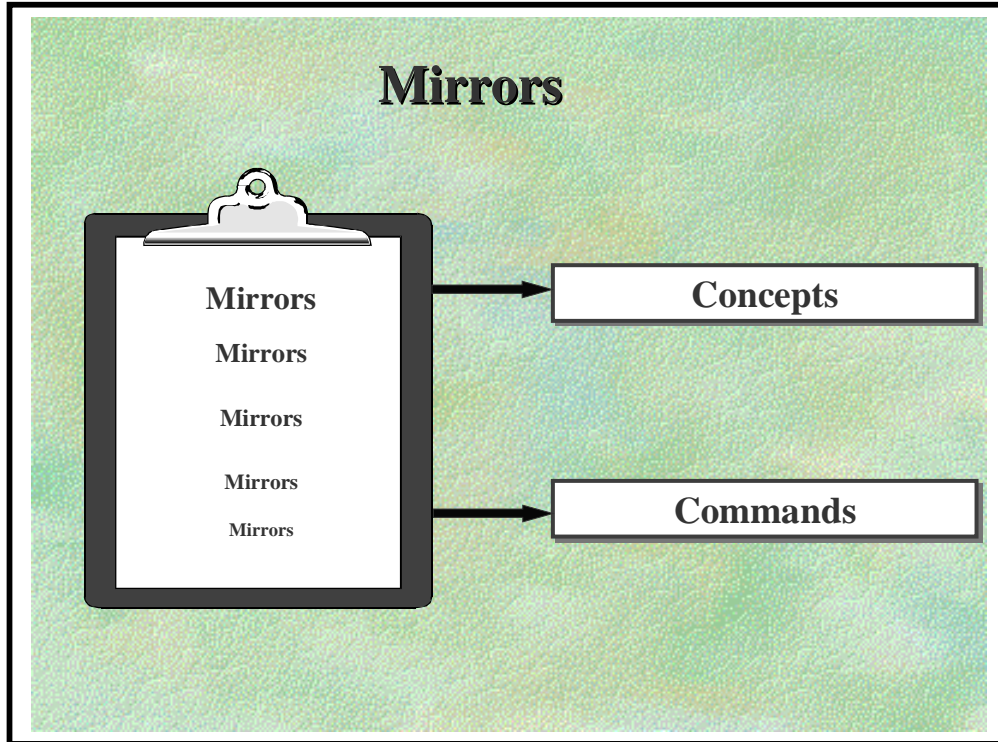
---

◆ 4.0 MIRRORING

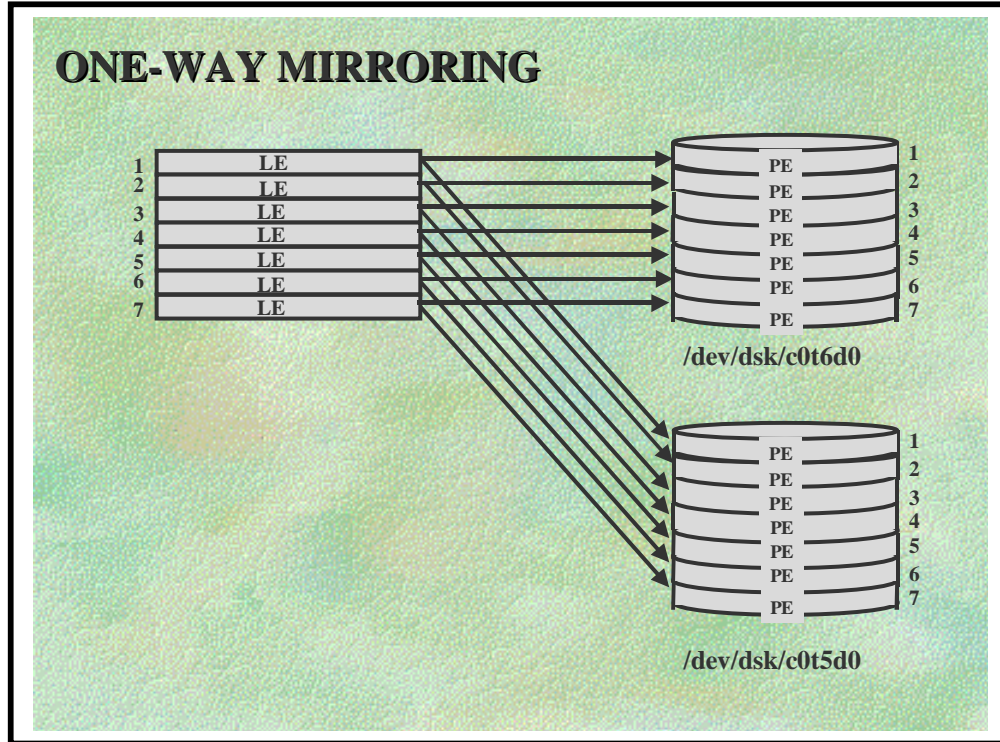
---

<b>The Basics</b>	<b>Structures</b>	<b>Modifying</b>
<b>Mirrors</b> 	<b>Boot Disks</b>	<b>Recovery</b>

◆ 4.0 MIRRORING



## ◆ 4.1 Concepts



- Requires optional software product MirrorDisk/UX
- Allows duplicate copies of the data to be kept on separate Physical Volumes
- Allows online backups

### **Allocation policies for mirrored Physical Extents:**

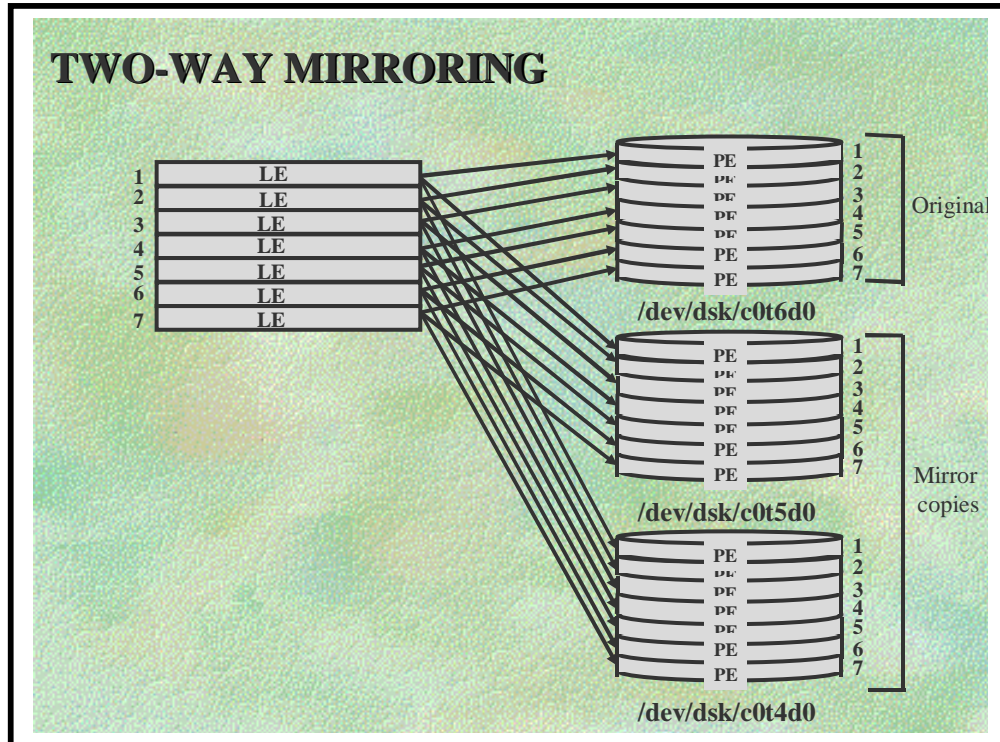
#### **strict (default)**

- PE's containing copies must be on separate Physical Volumes
- High protection against hardware faults

#### **non-strict**

- Mirrored PE's can be on the same Physical Volume
- No protection of data if Physical Volume fails

## ◆ 4.1 Concepts



- Two-way mirroring has three Physical Extents allocated for each Logical Extent.
- Useful for allowing on-line backups to take place While still having the on-line data mirrored
  - \* This can be desirable when data availability is critical
- A mirrored logical volume can be created using the *lvcreate* command with the *-m* option followed by the number of mirrored copies we want

---

## ◆ 4.2 Commands

---

### **Example:**

- **Creating a Mirrored Logical Volume.**

To create a logical volume of 40 megabytes in /dev/vg01, singly mirrored with strict allocation, with the Mirror Write Cache off, and the Mirror Consistency Recovery mechanism enabled, type:

```
# lvcreate -L 40 -m 1 -M n -c y /dev/vg01
```

A logical volume with name "lv01" will be created.

Logical volume "/dev/vg01/lv01" has been successfully created with lv number 1.

Logical volume "/dev/vg01/lv01" has been successfully extended.

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

The above command creates a logical volume /dev/vg01/lv01 since this is the first logical volume created in this volume group. The logical volume has a size 40 megabytes; if the default size of each logical extent is four megabytes, there are two sets of physical extents created on each of two disks in the volume group vg01. For this logical volume, the Mirror Write Cache is disabled and the Mirror Consistency Recovery mechanism is enabled.

Note that you must use either the -L (megabytes) or the -l (logical extents) option when creating a new mirrored logical volume; you cannot mirror a logical volume of zero size.

### **Example:**

- **Increasing the Number of Mirror Copies.**

To mirror an unmirrored logical volume, or increase the number of mirrored copies from one to two, you can use the *lvextend(1M)* command.

---

## ◆ 4.2 Commands

---

You can also, if necessary, specify to which disk LVM is to allocate the new mirror copies.

To mirror `/dev/vg01/lvol2`, which is currently unmirrored, you would type:

```
# lvextend -m 1 /dev/vg01/lvol2
```

*The newly allocated mirror is now being synchronized.*

*This operation will take some time. Please wait...*

*Logical volume “/dev/vg01/lvol2” has been successfully extended*

The `-m 1` option specifies that you want to create one mirror copy in addition to the original copy of the logical volume. After LVM creates the mirror, it synchronizes the data in the newly created physical extents with the original data. This can take some time, depending on how large the logical volume is.

To mirror `/dev/vg01/lvol2`, which is currently unmirrored, and have LVM place the mirror copy on the specific physical volume `/dev/dsk/c0t4d0`, type:

```
# lvextend -m 1 /dev/vg01/lvol2 /dev/dsk/c0t4d0
```

### **Example:**

- **Creating Mirrored Logical Volumes that Use Specific Disks.**

Let's suppose you want to create a singly mirrored logical volume of 60 megabytes for a file system. You want the logical volume to use two specific disks (physical volumes) of the same type; the volume group contains several disk types. By having the logical volume's mirrored copies on disks of the same type, you intend to optimize the file system's performance.

---

## ◆ 4.2 Commands

---

1. Create a logical volume, no size, no mirroring, with the name saleslv.

```
# lvcreate -n saleslv /dev/vg05
```

Let's assume logical volume /dev/vg5/saleslv is successfully created.

2. Extend the file system by 1 extent to one of the specific disks.

```
# lvextend -l 1 /dev/vg05/saleslv /dev/dsk/c0t4d0
```

3. Then extend the logical volume again, this time specifying that a mirror copy be created on the second desired disk.

```
# lvextend -m 1 /dev/vg05/saleslv /dev/dsk/c0t5d0
```

4. Now, extend the logical volume on both physical volumes to the total number of desired extents (or a specific size in megabytes).

```
# lvextend -L 60 /dev/vg05/saleslv /dev/dsk/c0t4d0 /dev/dsk/c0t5d0
```

This last step occurs quickly because the extents being added to the logical volume contain no data. You could have eliminated the last step by extending the logical volume to 60 megabytes in step two, but the subsequent mirroring step (step 3) would require time for resynchronization.

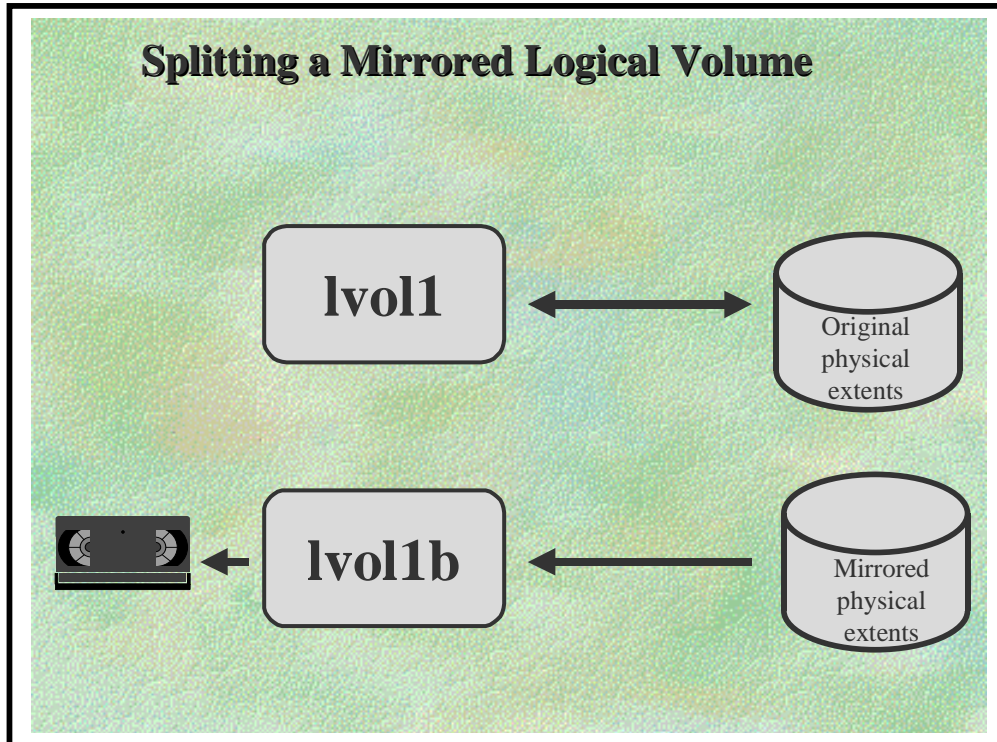
**NOTE:** In the future, when you extend a mirrored logical volume and want to continue to use the same specific physical volumes, remember to specify those specific physical volumes when you issue the command.

:

---

**◆ 4.2 Commands**

---



### Splitting and Merging a Mirrored Logical Volume.

You can split a mirrored logical volume into two logical volumes using the `lvsplit(1M)` command.

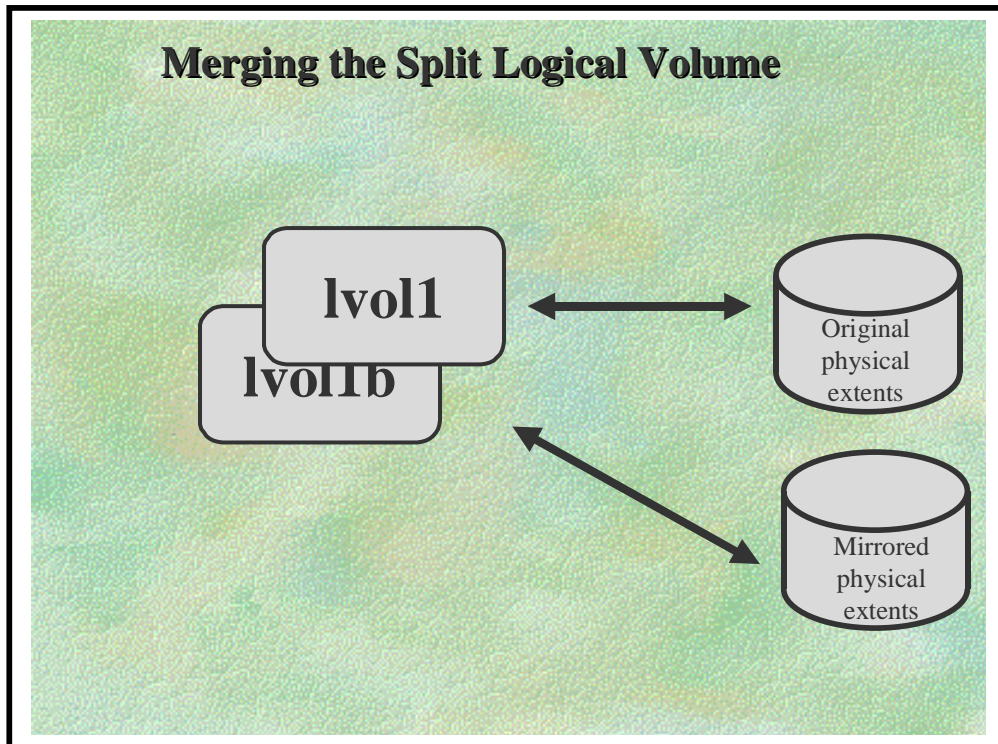
The original logical volume retains the original name, while the split-off logical volume has the suffix `b`. If you want to assign a specific name to the split-off logical volume, you can use the `-s` option to create a custom suffix.



---

**◆ 4.2 Commands**

---



You merge the split pair back into one mirrored logical volume using the *lvmerge(1M)* command.

**DISCUSSION, SPLITTING AND MERGING A MIRRORED LOGICAL VOLUME**

As an example, you can split a mirrored logical volume to back up one of the mirrored copies of a file system while leaving the other mirrored copy of the file system open for use. When you complete the backup activity, you can merge the split pair back to one mirrored logical volume.

For each mirrored logical volume that you split into two logical volumes, LVM dynamically creates a table that it uses to resynchronize the two when you merge them back again. The table reduces synchronization time. The table exists as long as neither of the split pair is extended, reduced, or split again, or until the system is rebooted. If no table exists, all data is resynchronized.

---

## ◆ 4.2 Commands

---

When LVM merges the split pair, it resynchronizes the data, updating the physical extents in the split-off copy based on changes made to the copy that remained in use. No resynchronization takes place if no changes occurred in either copy.

### Example:

- Online Backup Using *lvsplit*, *lvmerge*

Suppose you want to back up the file system */lptest* in the doubly mirrored logical volume */dev/vg02/lvol1*. Because you want to access the file system even while backing it up (that is, you do not want to *umount* it), you plan to temporarily split the logical volume in two).

1. Split the logical volume. (You could first use *lvdisplay -v /dev/vg02/lvol1* to verify the number of current logical extents and allocated physical extents.)

```
# lvsplit -s backup /dev/vg02/lvol1
```

*Logical volume "/dev/vg02/lvol1backup" has been successfully created with lv number 5.*

*Logical volume "/dev/vg02/lvol1" has been successfully split.*

The original logical volume */dev/vg02/lvol1* continues to be available for use; using the display commands, you can verify that it now has one fewer mirror copies of each logical extent; it would now have one mirror if it had two mirrors before you split it, and it would now have no mirrors if it had one mirror before.

The new logical volume */dev/vg02/lvol1backup*, split off from the original, now exists.

2. Perform backup (or other file system operation) on the file system in */dev/vg02/lvol1backup*.

**NOTE:** First perform file system consistency check using *fsck(1M)*. Then, you can mount it and back it up.

---

**◆ 4.2 Commands**

---

3. *Unmount* the file system (if you had mounted it) in the split-off logical volume when you are ready to merge the split pairs.

4. Merge the split-off logical volume and the original logical volume back into one mirrored logical volume.

```
# lvmerge /dev/vg02/lvol1backup /dev/vg02/lvol1
```

**CAUTION:** Notice the sequence.

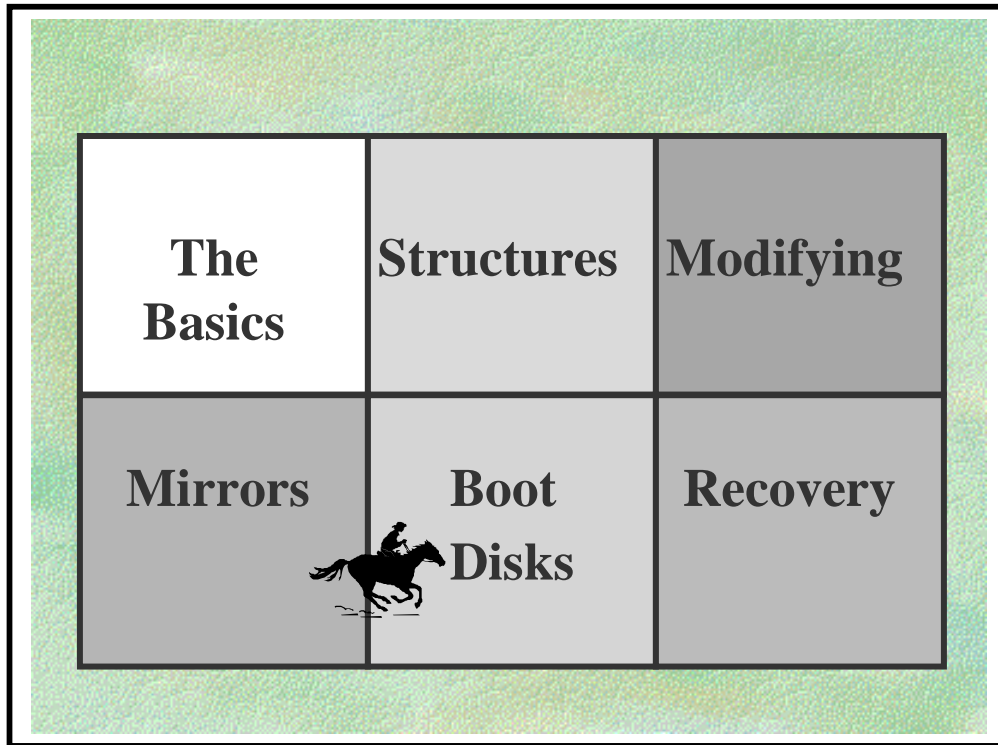
```
lvmerge split-off_logical_volume original_logical_volume
```

Do not reverse the order, or else the more recent data could be overwritten.

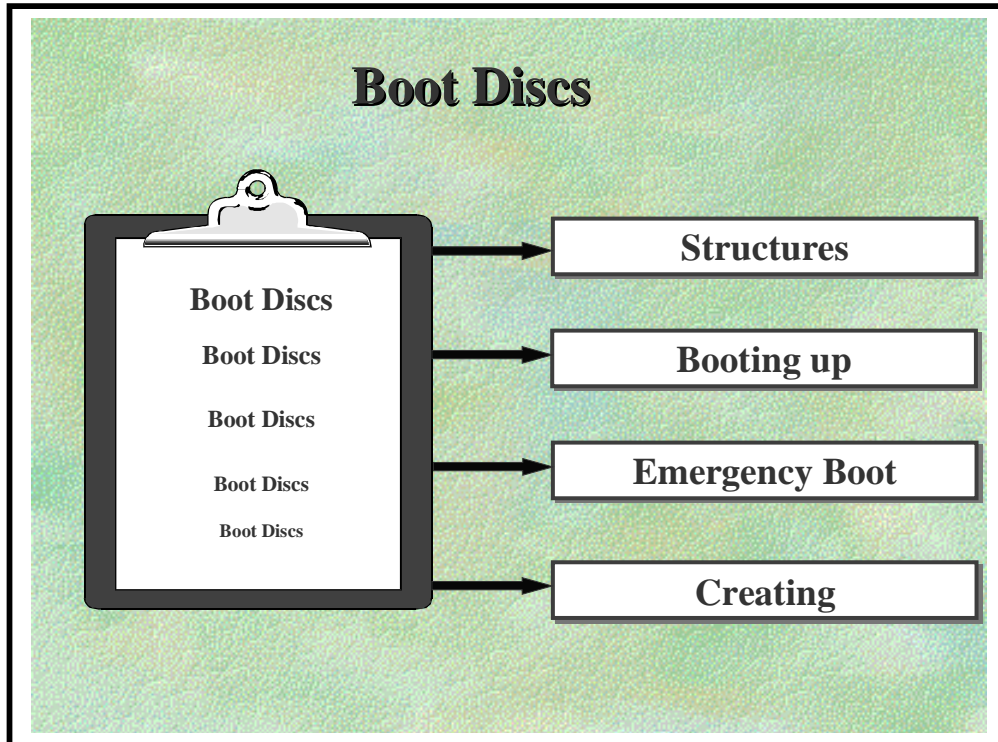
The split-off logical volume merges back with the original logical volume, now modified.

After the merge, you can verify that the logical volume's mirrored state matches the pre-split state and that the split-off logical volume no longer exists.

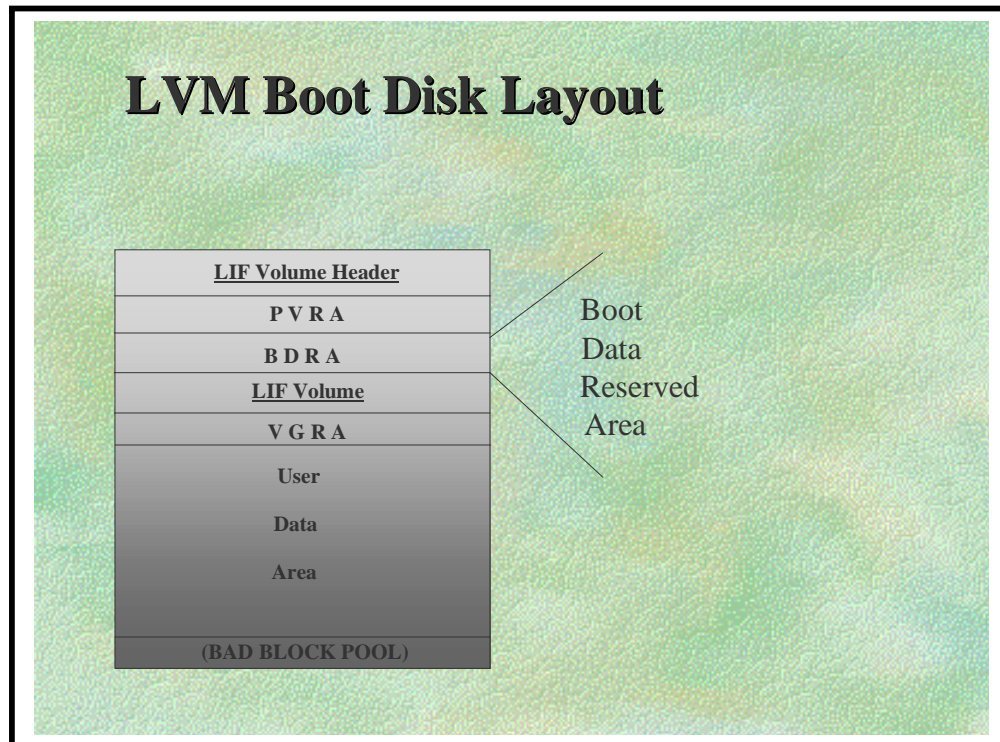
◆ 5.0 BOOT DISKS



◆ 5.0 BOOT DISKS



## ◆ 5.1 Structures



Boot Data Reserved Area (BDRA) contains:

- \* Location and sizes of disks in root Volume group
- \* Knows locations of root, primary swap and dump logical volumes
- \* Kernel uses this information to configure root and primary swap at boot-up

Maintained using *lvlnboot* and *lvrmboot*

Bad Block relocation is not supported on the root and primary swap logical volumes

---

**◆ 5.1 Structures**

---

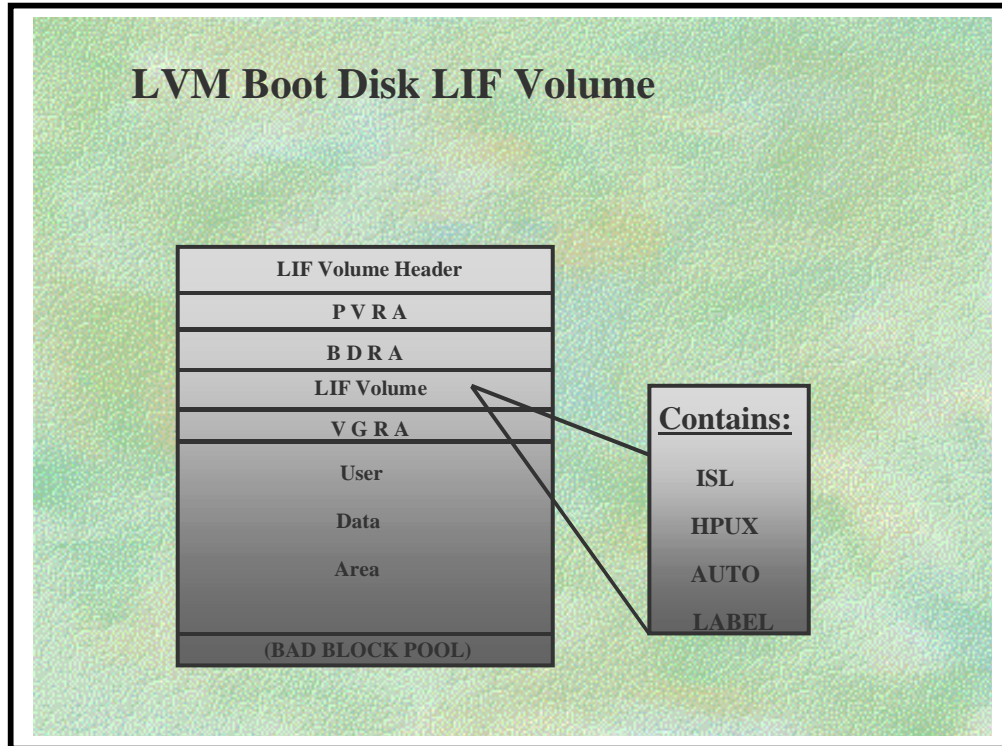
To boot the system, the kernel activates the volume group to which the system's root logical volume belongs. The location of the root logical volume is stored in the boot data reserved area (BDRA). The boot data reserved area contains the locations and sizes of LVM disks in the root volume group and other vital information to configure the root, primary swap, and dump logical volumes, and to mount the root file system.

The BDRA contains the following records about the system's root logical volumes: timestamp (indicating when the BDRA was last written), checksum for validating data, root volume group ID, the number of LVM disks in the root volume group, a list of the hardware addresses of the LVM disks in the root volume group, indices into that list for finding root, swap, and dump, and information needed to select the correct logical volumes for root, primary swap, and dumps.

---

**◆ 5.1 Structures**


---



Boot area in two parts: LIF header and LIF volume

LIF Header contains pointers to files in LIF volume

LIF volume contains:

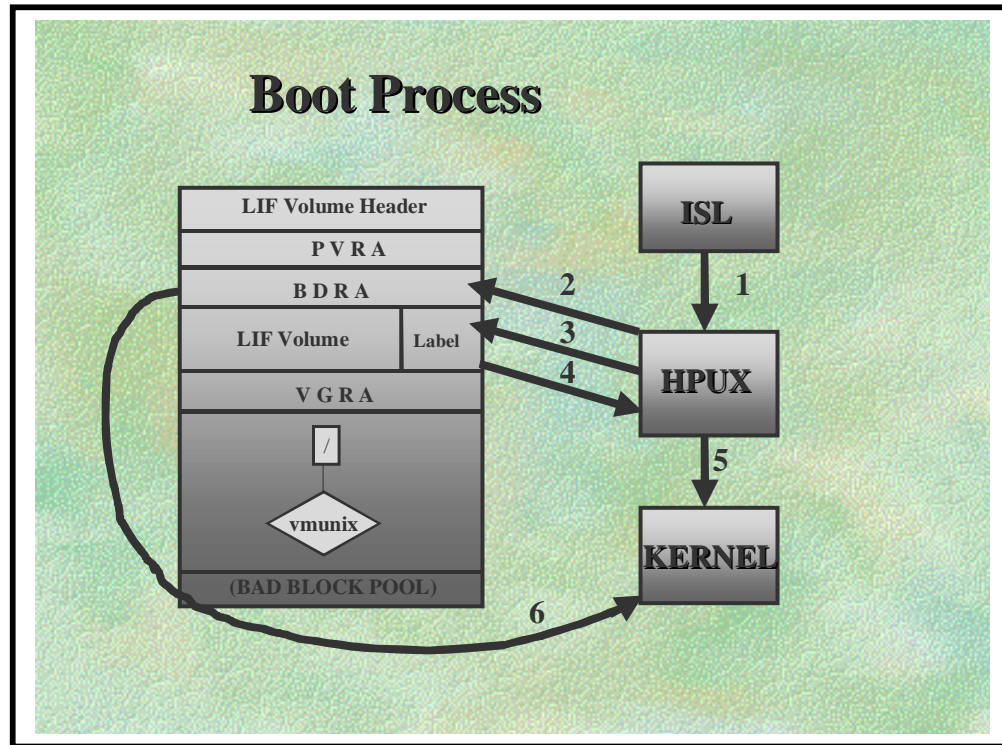
- **ISL** - Initial System Loader
- **HPUX** - kernel loader
- **AUTO** - contains the *autoboot* string
- **LABEL** - used by HPUX to locate the root logical volume during a normal boot. Updated by *lvlnboot* and *lvrmboot* commands.

Created only by the *mkboot* command

**CAUTION:** *dd* and *lifcp* should not be used to create a Boot Area. LVM information (PVRA,BDRA) will be overwritten!



## ◆ 5.2 Booting up



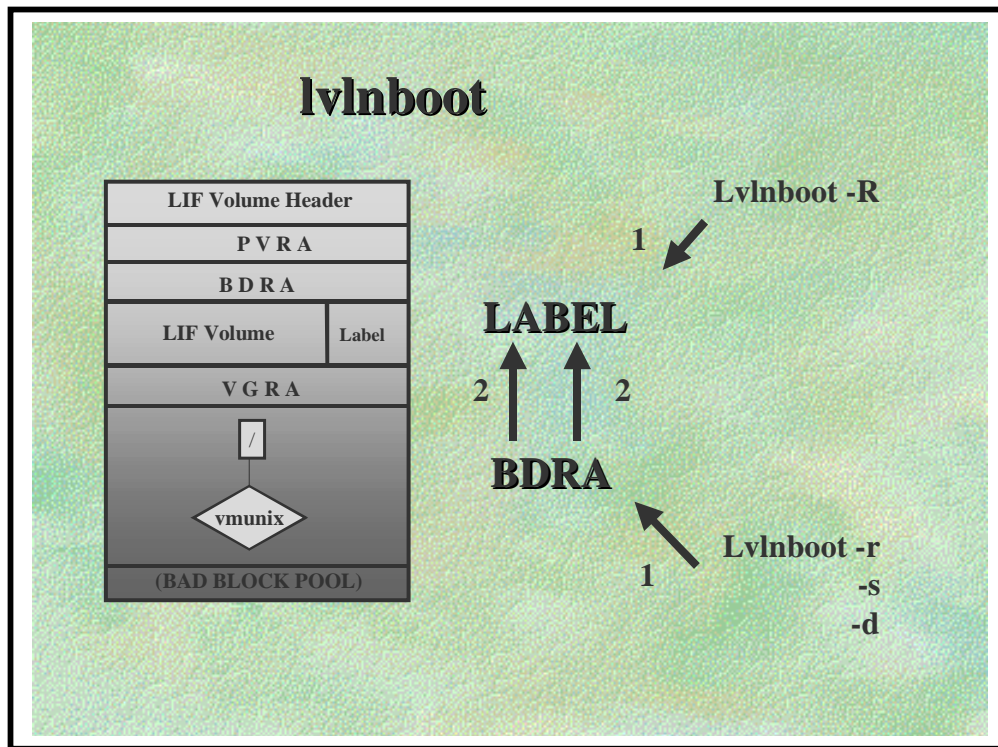
The boot process is as follows:

- 1) ISL loads HPUX loader.
- 2) HPUX loader checks in the BDRA to find the boot Physical Volume.
- 3) HPUX loader reads LABEL file from the boot Physical Volume.
- 4) ...to find the location of the root filesystem.
- 5) HPUX loader now loads the */stand/vmunix* kernel into memory.
- 6) Kernel reads BDRA to configure root logical volume, primary swap logical volume and any dump logical volumes.

---

**◆ 5.2 Booting up**


---



**LABEL** and **BDRA** are updated and created by *lvlnboot*

- **-r** updates *root* logical volume definition.
- **-s** updates *swap* logical volume definition.
- **-d** updates *dump* logical volume definition.

**NOTE:** All the above options update the BDRA and then the LABEL file

- **-R** updates LABEL files on all boot discs in the volume group specified with the contents of the BDRA.

**NOTE:** This option makes no changes to the contents of the BDRA: if it's empty, it will still be empty until one of the -r, -s, or -d options are used.

### ◆ 5.3 Emergency boot

## Booting a Corrupted LVM Bootable Disk

\* If the LABEL file is corrupt the location of the "root logical volume" is unknown. LVM architecture requires the "root logical volume" to be placed at a certain location on the boot disk. Using this known offset allows us to boot in maintenance mode.

\* To boot in "maintenance mode", type the following from the ISL> prompt

```
ISL>hpux -lm (;0)/stand/vmunix
```

Key

-lm Do not use information in BDRA and LABEL file  
Boot in LVM "maintenance mode"  
i.e. -system is in single-user mode  
-no volume groups are activated  
-must be rebooted

***ISL> hpux -lm (;0)/stand/vmunix***

Uses a known offset to find start of lvoll.

Does not use information in BDRA and LABEL file.

Boots in LVM "maintenance mode".

- system is in single user mode
- no volume groups are activated
- must be rebooted when finished.

Maintenance mode boot provides a means, during system installation, or when critical LVM data structures have been lost, to boot from an LVM disk without the LVM data structures.

---

**◆ 5.3 Emergency boot**

---

The system can be booted in maintenance mode using the *-lm* option to the *hpux* command at the *ISL>* prompt.

This causes the system to boot to single-user mode without primary swap, dump, or LVM to access the root file system.

**NOTE:** The system must not be brought to multi-user mode (that is, *init 2*) when in LVM maintenance mode. Corruption of the root file system might result.


This page is left blank intentionally

---


◆ 5.4 Creating

---

**Creating Boot Disk in the Root Volume Group**



Creating a bootable disk is useful for two specific cases



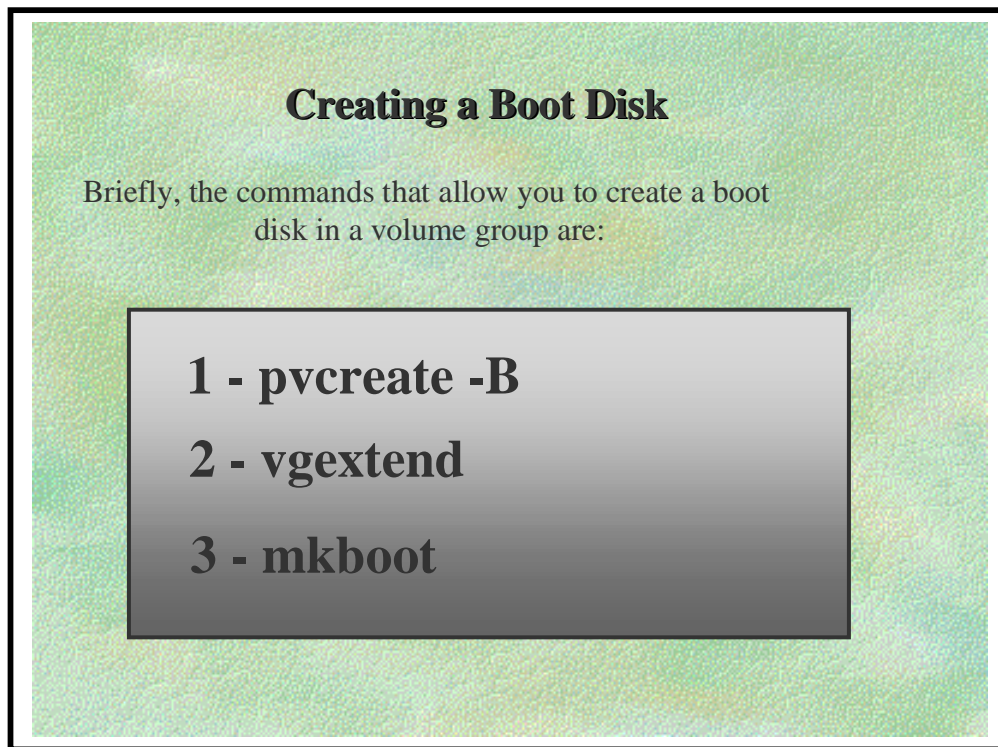
- \* Creating a mirror for the root logical volume
- \* Creating a new root logical volume (moving root from one disk to another)

- There are step-by-step cookbooks for creating bootable physical volumes in a later module

---

**◆ 5.4 Creating**

---



### Creating Boot Disks in the Root Volume Group

**Creating a bootable disk** is useful for two specific cases:

- 1) Creating a mirror for the root logical volume.
- 2) Creating a new root logical volume (moving root from one disk to another).

Briefly, the steps to **add a boot disk to the root volume group** are:

- 1) Use `pvcreate -B` when you make the disk a physical volume.
- 2) Add the disk to a volume group with `vgextend`.
- 3) Use `mkboot` to place boot utilities (LIF) in the boot area.
- 4) Use `mkboot -a` to modify the AUTO file in the boot LIF area.

## ◆ 5.4 Creating

The following was obtained from the HP-UX Reference Manual - Volume 2 Section 1M and is not intended to represent the whole man page for mkboot:

mkboot, rmboot - install, update or remove boot programs from disk

### SYNOPSIS

```
/usr/sbin/mkboot [-a auto_file_string] [-v] device
```

```
/usr/sbin/rmboot device
```

### DESCRIPTION

Boot programs are stored in the boot area in Logical Interchange Format (LIF), which is similar to a file system. For a device to be bootable, the LIF volume on that device must contain at least the ISL (the initial system loader) and HPUX (the HP-UX bootstrap utility) LIF files. If, in addition, the device is an LVM physical volume, the LABEL file must be present (see lvinboot(1M)).

### Options

mkboot recognizes the following options:

- |               |  |
|---------------|--|
| -a auto_file_ | If the -a option is specified, mkboot creates an autoexecute file AUTO on device, if none exists mkboot deposits auto_file_string in that file. If this string contains spaces, it must be quoted so that It is a single parameter.                    |
| device        | Install the boot programs on the given device special file . The specified device can identify either a character-special or block-special device. However, mkboot requires that both the block and character device special files be present . mkboot |



---

**◆ 5.4 Creating**

---

attempts to determine whether device is character or block special by examining the specified path name.

For this reason, the complete path name must be supplied. If `mkboot` is unable to determine the corresponding device file, a message is written to the display, and `mkboot` exists

**EXAMPLES:**

To copy all LIF utilities from `/usr/lib/uxbootlf` to boot disk Instance 6:

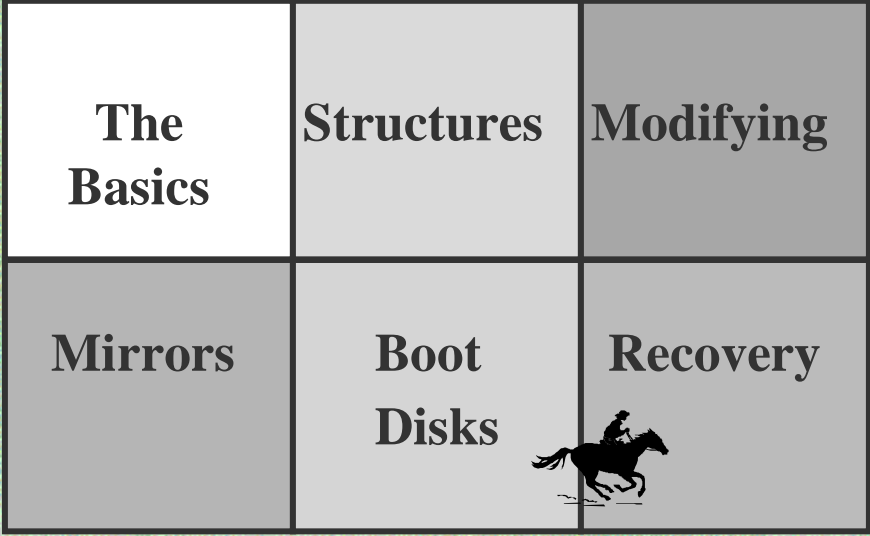
```
# mkboot /dev/rdisk/c0t6d0 (LVM File System)
```

To copy and update LIF's AUTO file with boot string information on Instance 6:

```
#mkboot -a "hpux (;0)/stand/vmunix" /dev/rdisk/c0t6d0 (LVM File System)
```

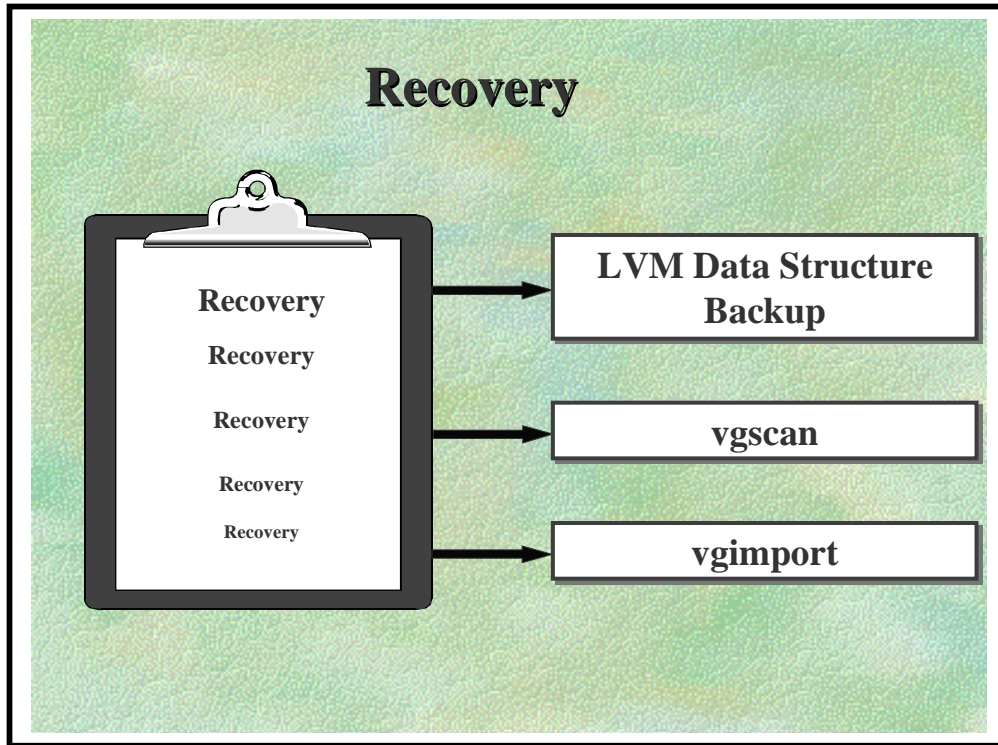
This page is left blank intentionally

◆ 6.0 RECOVERY

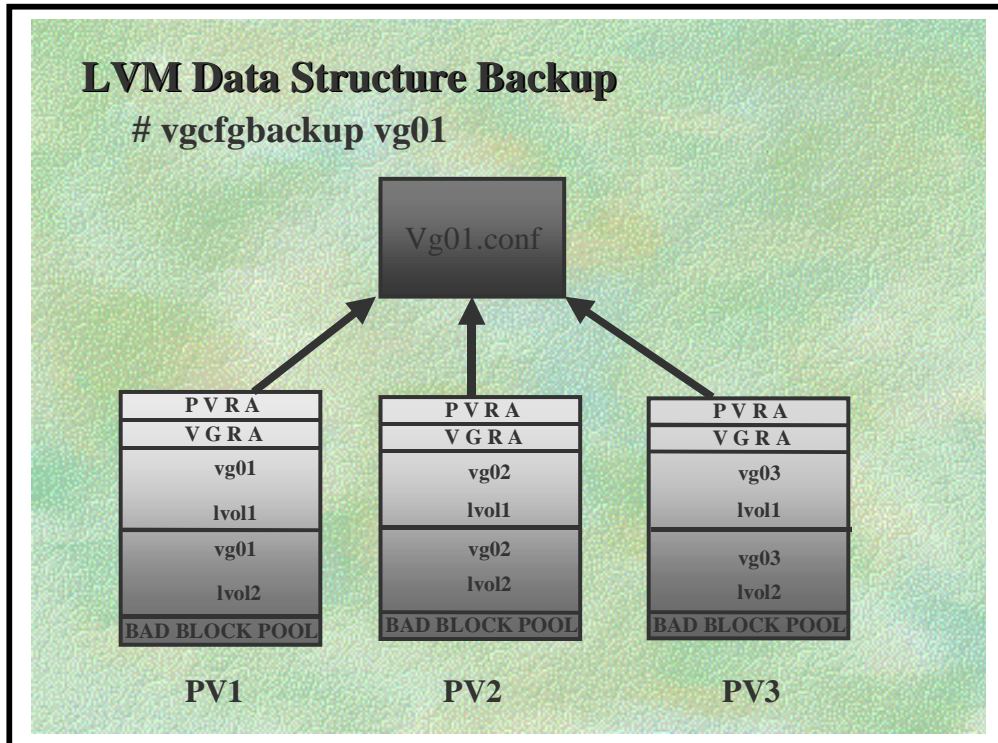


<b>The Basics</b>	<b>Structures</b>	<b>Modifying</b>
<b>Mirrors</b>	<b>Boot Disks</b>	<b>Recovery</b>

◆ 6.0 RECOVERY



## ◆ 6.1 LVM Data Structure Backup



- Only LVM data is backed up by `vgcfgbackup`.
- `vgcfgbackup` automatically runs whenever:
  - \* adding or removing disks to/from a volume group
  - \* changing boot disks in a volume group
  - \* creating or removing logical volumes and volume groups
  - \* extending or reducing logical volumes and volume groups
- `vgcfgbackup` saves the configuration information in `/etc/lvmconf/vgXX.conf`
- The `vgcfgbackup` command backs up volume-group configuration information into binary files, one file per volume group.

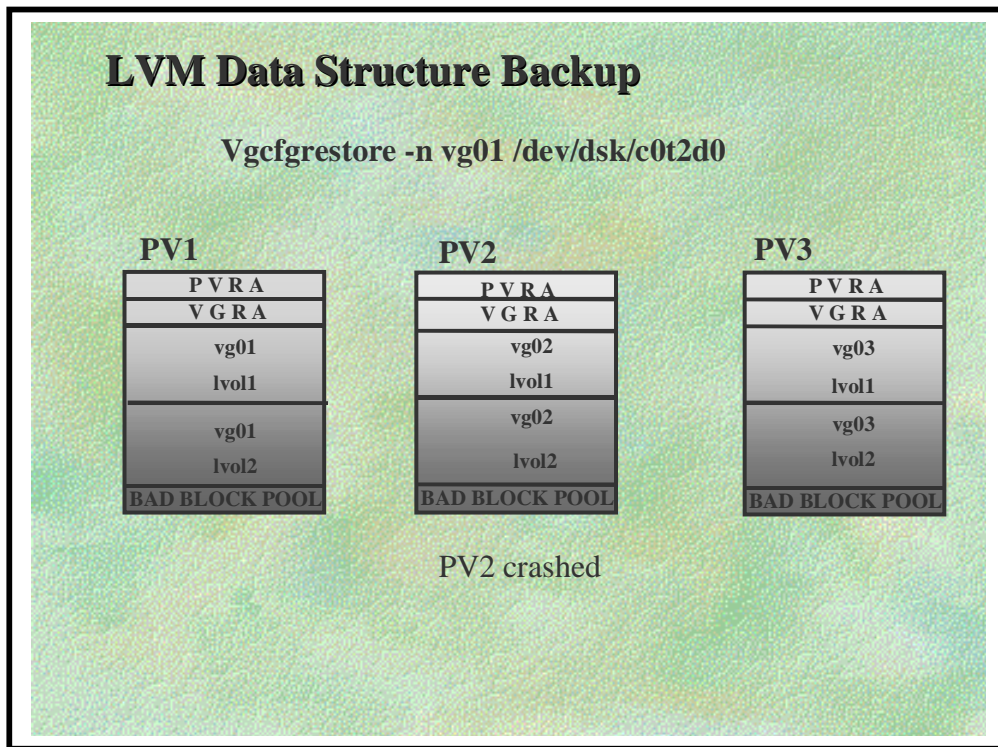
---

## ◆ 6.1 LVM Data Structure Backup

---

- *vgcfgbackup* backs up the LVM record of each LVM disk, the BDRA record for each bootable LVM disk, one copy each of the current VGDA and VGSA data structures, and LIF LABEL files.
- *vgcfgbackup* does not back up LIF header and files, nor bad block directory.
- *vgcfgbackup* can be turned off with the A -n option

## ◆ 6.1 LVM Data Structure Backup



Only LVM data is restored by *vgcfgrestore*.

Volume groups to be restored must be made not available first.

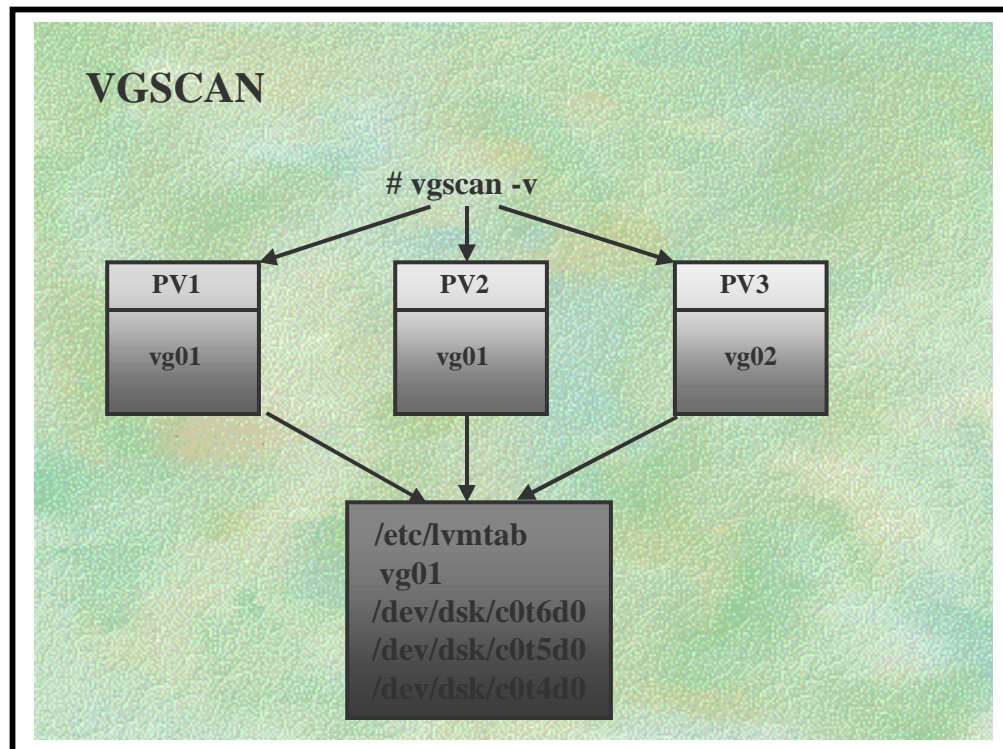
- `vgchange -a n vg01`

To restore the LVM data structures on volume group `vg01`:

```
# vgchange -a n vg01
# vgcfgrestore -n vg01 /dev/rdisk/c0t2d0
# vgchange -a y vg01
```

Then customer data has to be restored from backup to all the affected logical volumes if it could not be saved.

## ◆ 6.2 vgscan



- *vgscan* searches all disks powered-up and on-line looking for LVM information.
- Tries to group these by volume group and recreate their entries in */etc/lvmtab* if there is sufficient information available.
- If there is not sufficient information available then it will suggest that the group of disks should be *vgimport*'ed.

**NOTE:** In order to recreate */etc/lvmtab*, */etc/lvmtab* has to be renamed or removed first

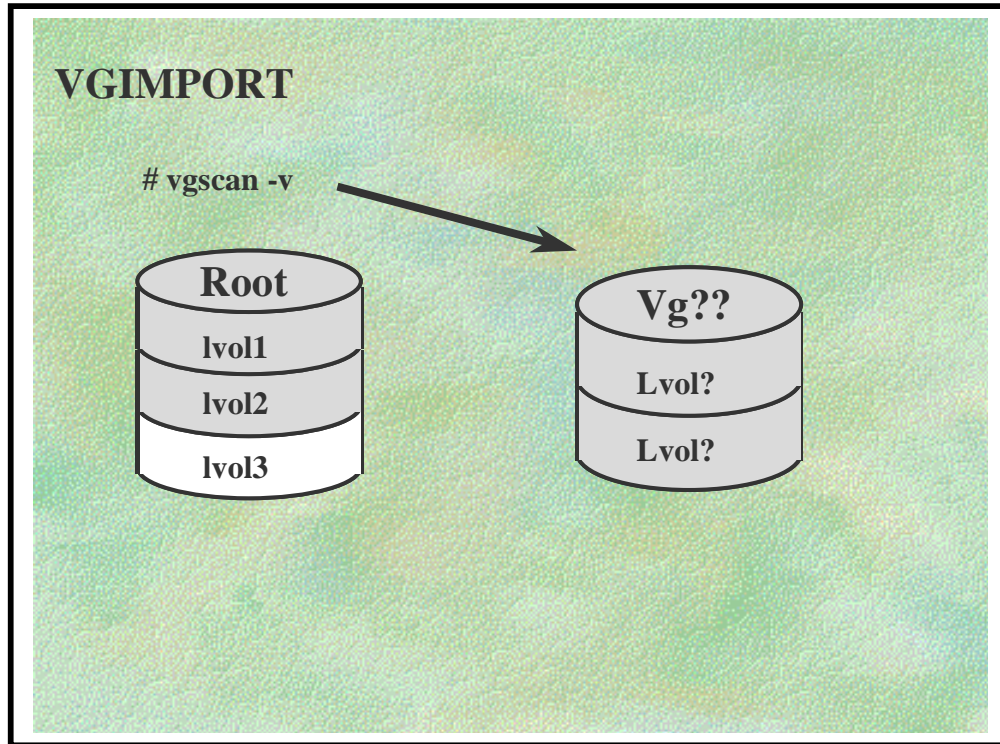


This page is left blank intentionally

---

**◆ 6.3 vgimport**

---



Using *vgimport* to recover a lost volume group involves two steps:

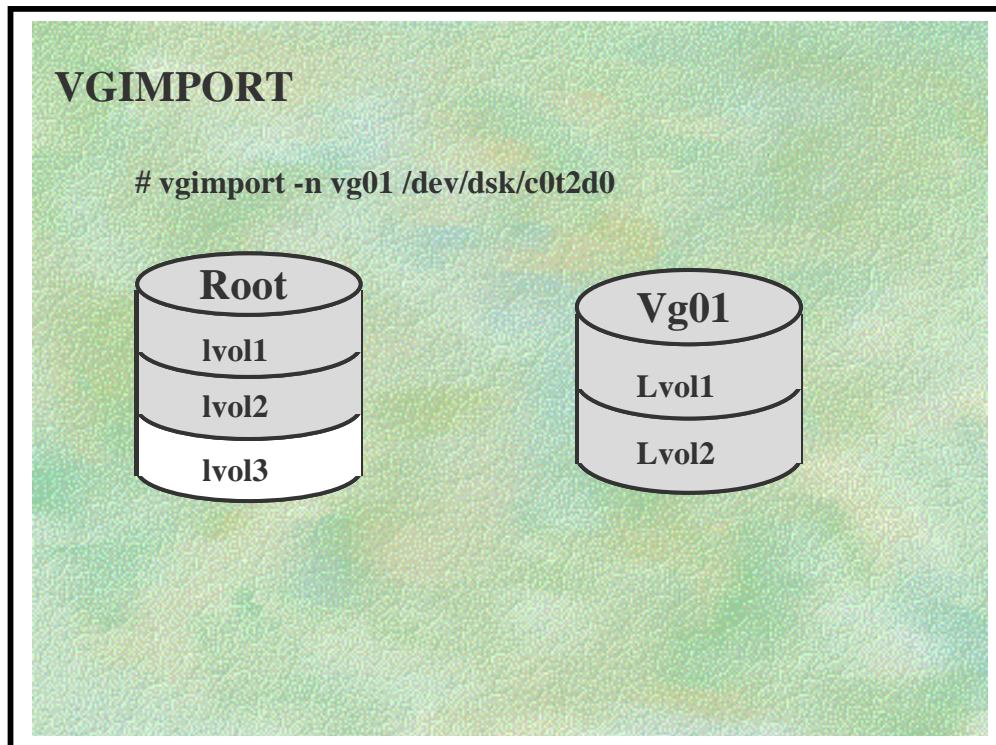
**1) vgscan**

This scans the disks and suggests which groups of disks should be *vgimport*ed.

---

**◆ 6.3 vgimport**

---



## 2) vgimport

The *group* file must be created first with *mknod* before doing this.

The *vgimport* command needs the desired name of the volume group and the device files, as found by *vgscan*, of all its Physical Volumes. *vgimport* will then create all the logical volume device files required and update */etc/lvmtab*.

**NOTE:** If the logical volumes had names rather than *lvol1*, *lvol2* etc., these can simply be renamed using *mv*. Alternatively, if a *mapfile* had been available then it could be specified with *vgimport -m*.

---

**◆ 6.3 vgimport**

---

**USING VGIMPORT AND VGEXPORT EFFECTIVELY**

*The following is taken from an article in PA-NEWS number 198.*

A system recovery which involves a re-install, many VG's, and re-ordered Instance's (thanks to auto-config) can be tedious.

The utility most people will try to use first is *vgscan*.

However, *vgscan* only works if:

- All LVM device files for the VG exist
- All the information contained in those device files is scrupulously correct.(minor no's, etc)
- The */etc/lvmtab* file contains no mention of the VG
- The */etc/lvmtab* file does not have the PV device file Instance's assigned to another VG

If the LVM information is incorrect or unknown then it is better to use *vgexport* and *vgimport* than to try to guess it using *rmsf/insf/vgscan*.

**Case #1 The Root disc crashes requiring a re-install**

After the re-install you realize that the Instance numbers have changed. Normally, a complete restore from a current full backup of the root disc (including device files) followed by a reboot should clear this up, but in real life the customer never has one when you need it. (Murphy's Law ...)

What are your alternatives?

1) If the Volume Group files exist, you could use *rmsf* and *insf* to get all the device files correctly assigned again and then *vgscan*. If the Volume Group files do not exist but their names and minor numbers are known then they could be recreated, followed by *rmsf/insf/vgscan* as above. This may work if the number of discs involved is small.

2) You could recreate all the VG's and their LV's and then restore the data. It's reliable but not very practical.

3) You could use *vgexport/vgimport*. This way you don't have to worry about Instances: you only need to know which disks belonged to which VG.

---

**◆ 6.3 vgimport**

---

**Example 1**

*/dev/dsk/c0t5d0*, *c0t6d0*, and *c0t7d0* are all members of */dev/vg05*. The system crashes and needs to be restored. After the re-install, *c0t5d0* is OK but *c0t6d0* and *c0t7d0* have been reconfigured as *c0t10d0* and *c0t12d0* respectively. Of course, the customer does not have a backup of */dev* or */etc/lvmtab*, so */dev/vg05* does not get activated at boot time.

You have to recreate the Volume Group.

**Example 1 Answer**

As there is no backup, the problems lie in determining the volume group number and name, and finding out which discs were originally in the configuration.

To recreate this group:

**1) # *vgscan***

*this will tell you which discs have the same associated information*

**2) # *mkdir /dev/vg05; mknod /dev/vg05/group c 64 0x0y0000***

*substitute VG number for y*

**3) # *vgimport /dev/vg05 /dev/dsk/c0t5d0 /dev/dsk/c0t10d0 /dev/dsk/c0t12d0*****Example 2**

Instead of a crash, the disks were moved around onto different busses to improve utilization. The mappings after modification and boot are also identical to the example above. This time 2 of the 3 PV's, that make up the group, have not lined up with their original instances, quorum was not met so the VG was not activated. Restore the LVM configuration.

---

## ◆ 6.3 vgimport

---

### Example 2 Answer

In this case, the directory `/dev/vg05` exists and includes its LV device files. Also, the VG and its original PV's are listed in `/etc/lvmtab`. We can use `vgexport/vgimport` to include the PV's in the VG with their new LU numbers.

- 1) Note the minor number of `/dev/vg05/group` - this will be removed by `vgexport`.
- 2) If the VG did get activated somehow (manually or quorum was met) deactivate it using `vgchange -a n /dev/vg05`
- 3) `vgexport -m /tmp/mapfile /dev/vg05`
- 4) `mkdir /dev/vg05; mknod /dev/vg05/group c 64 0x0y0000`  
*substitute VG number for y*
- 5) `vgimport /dev/vg05 /dev/dsk/c0t5d0 /dev/dsk/c0t10d0 /dev/dsk/c0t12d0`
- 6) `vgchange -a y /dev/vg05`

### **Case #2 Could we have a spare disc ready for online replacement of a failed disc?**

This is possible, but while the disc was being replaced the VG would not be available.

The better (and more costly) solution is LVM mirroring.

### Example 3

`/dev/dsk/c0t6d0`, `c0t7d0` and `c0t8d0` are members of `/dev/vg03`.  
`/dev/dsk/c0t9d0` is an empty disc configured into the system but not a member of any group. `c0t7d0` has problems: recover using `c0t9d0`.

---

**◆ 6.3 vgimport**

---

**Example 3 Answer**

Replace it with *c0t9d0*:

- 1) Note the minor number of */dev/vg03/group* - this will be removed by *vgexport*.
- 2) If the VG did get activated somehow (manually or quorum was met) deactivate it using *vgchange -a n /dev/vg03*
- 3) *vgexport -m /tmp/mapfile /dev/vg03*
- 4) *mkdir /dev/vg03; mknod /dev/vg03/group c 64 0x0y0000*  
*substitute VG number for y*
- 5) *vgcfgrestore /dev/vg03 -o /dev/dsk/c0t7d0 /dev/dsk/c0t9d0*
- 6) *vgimport /dev/vg03 /dev/dsk/c0t6d0 /dev/dsk/c0t9d0 /dev/dsk/c0t8d0*
- 7) *vgchange -a y /dev/vg05*
- 8) Use *pvdisplay -v /dev/dsk/c0t9d0 | more* to see which LV's were affected by the loss of this disc, and restore data to those LV's.

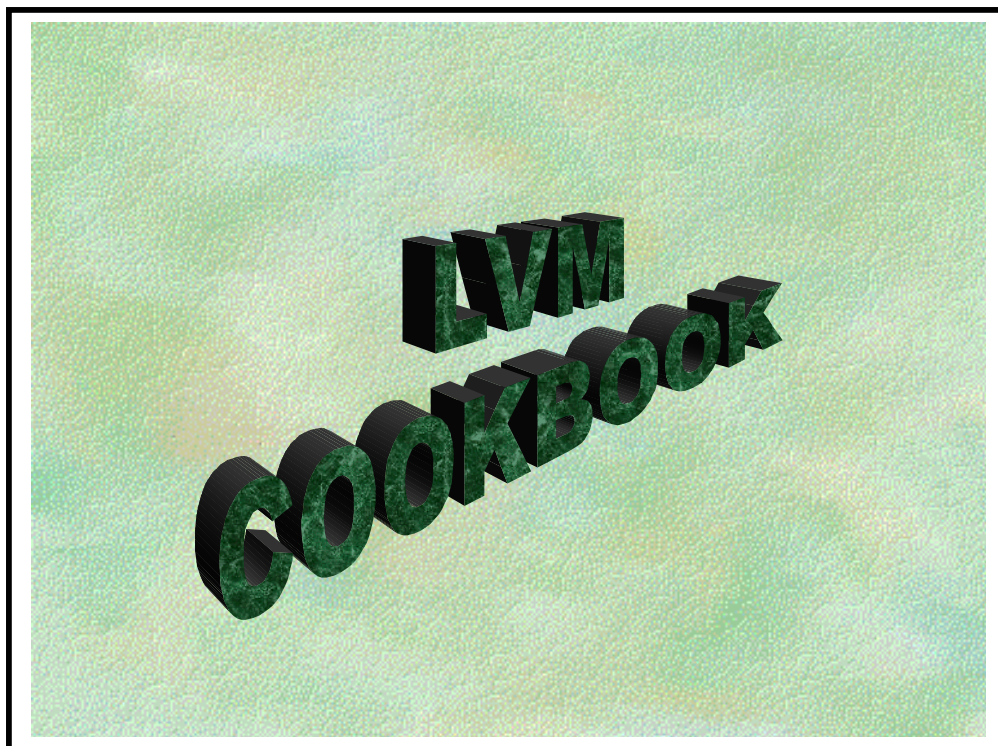
This page is left blank intentionally



---

◆ LVM Cookbook

---



---

◆ **LVM Cookbook**

---

**1/** Add an LVM Disk

**2/** Extend an LVM Logical Volume

**3/** Reduce the size of an LVM Logical Volume

**4/** Remove an LVM Volume Group

**5/** Moving a LVM Logical Volume

**6/** Exporting and Importing an LVM Volume Group

**7/** Adding Secondary Device Swap on a Logical volume

**8/** Adding Dumps Devices on a Logical Volume

**9/** Booting a Damaged LVM Bootable Disk

**10/** Mirroring a Root Disk

**11/** Lvsplit/lvmerge: Backing up a Mirrored Disk

**12/** Creating a Boot Disk

---

**◆ Add An LVM Disk**

---

1. Verify that the kernel has the correct drivers for the new disk drive. (Consult "Installing Peripherals" if you are unsure of the correct drivers.)

```
# lsdev | grep disc
```

If the correct drivers are not displayed, edit */stand/system*. Add the necessary drivers, then generate a new kernel.

2. Halt the system.

```
# shutdown -h 0
```

3. Install the new hardware. Verify that there are no bus address conflicts.
4. Boot the system. Once booted, use the following command to verify detection of the new disk drive. Note the assigned logical unit number for the new disk drive.

```
# ioscan -kfc disk
```

5. Create a physical volume for LVM use. Use the logical unit number obtained in step 4 of this procedure for the X in the command below:

```
# pvcreate -f /dev/rdisk/cXtYdZ
```

**CAUTION:** This will destroy any data on the disk.

6. Check *dev* to see if the volume group to reside on the new disk has a directory under */dev*. The default name for a volume group is *vgYY* where *YY* is the integer number of the group.

```
# ls /dev | grep vg
```

7. If necessary, create a new directory under */dev* for a new volume group to reside on the disk.

```
# mkdir /dev/vgYY
```

8. If a new volume group is to be created, make a group character device file for the new volume group.

```
# mknod /dev/vgYY/group c 64 0xZZ0000
```

---

## ◆ Add An LVM Disk

---

Where:

<b>Key</b>	<b>Operation</b>
------------	------------------

<i>ZZ</i>	HEXADECIMAL group number.
-----------	---------------------------

9. Create or extend the volume group to the new disk. Do ONE of the following:

```
# vgcreate /dev/vgYY /dev/dsk/cXtYdZ
```

Which is used to create a new logical volume.

```
# vgextend /dev/vgYY /dev/dsk/cXtYdZ
```

Which is used to extend a logical volume onto a new disk.

10. Create a logical volume on the new disk. Specify the size of the logical volume to be created in megabytes following the *L* option. The logical volume number will be the next integer not used (begins with 1).

```
# lvcreate -L SSS vgYY
```

Where:

<b>Key</b>	<b>Operation</b>
------------	------------------

<i>SSS</i>	The integer number of MB to be allocated for this logical volume.
------------	---

11. Create the physical file system for the new logical volume.

```
# newfs /dev/vgYY/rlvolZ
```

Where:

<b>Key</b>	<b>Operation</b>
------------	------------------

<i>Z</i>	The integer logical volume number.
----------	------------------------------------

12. Make a mount-point-directory on root (/) for the new logical volume.

```
# mkdir /vgYYlvolZ
```

13. Add a new line in /etc/fstab using vi or ed for the new logical volume.

```
/dev/vgYY/lvolZ /vgYYlvolZ hfs rw 0 P #/vgYYlvolZ
```

---

## ◆ Add An LVM Disk

---

Where:

Key	Operation
/dev/vgYY/lvolZ	the full path name of the block device file for the logical volume to be mounted.
/vgYYlvolZ	The full path name of the directory that the file system is to be mounted under.
hfs	Type (options): hfs - high performance file system, nfs - remote NFS file system, swap - swap file system, swapfs - dynamic swap file system, ignore - entry is ignored by mount and fsck.
rw	Options. (use default options, or comma separated list of options): ro - read only, rw - read/write (default), suid - Set-user-ID execution allowed, nosuid - Set-user-ID not allowed.
0	Backup frequency (set to 0).
P	Integer pass number P which determines the order in which fsck checks the file systems.
#	# begins the comment field.

14. Mount the new file system.

**# mount -a**

15. Backup the LVM data structures (PVRA and VGRA) for all groups affected. The default path name for the backup will be /etc/lvmconf/vgYY.conf.

**# vgcfgbackup /dev/vgYY** (Automatic unless turned off with A -n option)

---

**◆ Add An LVM Disk**

---

**LVM EXAMPLE:****\* Adding a New Disk; Volume Group; Logical Volumes.**

The following example adds a new disk to the system. The new disk will be placed into a new volume group. Next, two new 100MB logical volumes will be created on the new disk drive.

The commands assume the following:

The new disk's device files are /dev/[r]dsk/c0t4d0

The new volume group is vg05

The new logical volumes are lv01 and lv02

The following commands will accomplish the task:

```
# pvcreate -f /dev/rdsk/c0t4d0

# mkdir /dev/vg05

# mknod /dev/vg05/group c 64 0x050000

# vgcreate /dev/vg05 /dev/dsk/c0t4d0

# lvcreate -L 100 vg05

# lvcreate -L 100 vg05

# newfs /dev/vg05/rlvol1

# newfs /dev/vg05/rlvol2

# mkdir /vg5lv1 /vg5lv2

# mount /dev/vg05/lvol1 /vg5lv1

# mount /dev/vg05/lvol2 /vg5lv2

# vgcfgbackup vg05 (Automatic unless turned off with A -n option)
```

## ◆ Extend An LVM Logical Volume

1. Determine the device files for existing physical volumes in the associated volume group.

```
# strings /etc/lvmtab
```

2. Find all physical volumes associated with the volume group that the logical volume to be extended resides in. Look for free space with the following command.

```
# pvdisplay /dev/dsk/cXtYdZ
```

Where:

Key	Operation
Y	The physical volume's Instance number as indicated by <i>strings /etc/lvmtab</i> .

Repeat this for each physical volume in the logical volume's volume group.

Multiply the free PE extents times the PE extent size to determine if adequate free space is available to satisfy user space requirements.

NOTE: If sufficient free space is not available within the volume group, use the "Add an LVM Disk Cookbook" to add a new physical volume to the volume group. Once this has been accomplished, use this cookbook to extend the desired logical volume.

Determine the total size of the logical volume. The size in MB will be displayed. Note this value. It will be used in step 4 below.

Where:

Key	Operation
YY	The integer number of the volume group associated with the logical volume to be extended.

Z	The target logical volume's integer number.
---	---

4. Extend the logical volume.

```
# lvextend -L total_MB_in_lvvolume /dev/vgYY/lvolZ
```

Where:

---

## ◆ Extend An LVM Logical Volume

---

Key	Operation
total_MB_in_lv	Must be a larger number than that found in step 3 above.

5. Unmount the target logical volume.

```
# umount /dev/vgYY/lvolZ
```

Where:

Key	Operation
YY	The integer number of the volume group associated with the logical volume to be extended.
Z	The target logical volume's integer number.

6. Extend the physical file system.

```
# extendfs /dev/vgYY/rlvolZ
```

7. Mount the extended logical volume.

```
# mount -a
```

8. Backup LVM data structures (PVRA and VGRA).

```
# vgcfgbackup vgYY
```

(Automatic unless turned off with A -n option)



---

**◆ Extend An LVM Logical Volume**

---

**LVM Example:****\* Extend a Logical Volume.**

The following example extends a logical volume from 100 MB to 200 MB.

The commands assume the following:

The volume group is vg05

The logical volume to be extended is lvol2

The logical volume is in /etc/fstab

The following commands to accomplish the task:

```
# lvextend -L 200 /dev/vg05/lvol2
```

```
# umount /dev/vg05/lvol2
```

```
# extendfs /dev/vg05/rlvol2
```

```
# mount -a
```

```
# vgcfgbackup vg05 (Automatic unless turned off with A -n option)
```

---

## ◆ Reduce the Size of an LVM Logical Volume

---

1. Backup all user data in the logical volume to be reduced.

(Use `fbackup`, `cpio`, or `tar` backup utilities as appropriate or copy user data to another logical volume in the volume group using `pvmove`.)

2. Unmount the logical volume to be reduced using the `umount` command.

```
# cd / ; umount /dev/vgXX/lvolY
```

Where:

Key	Operation
XX logical	The integer number of the volume group associated with the volume to be reduced in size.
Y	The target logical volume's integer number.

3. Determine the present total size of the logical volume to be reduced using `lvdisplay`. This value is presented as LV Size (Mbytes).

```
# lvdisplay /dev/vgXX/lvolY
```

4. Reduce the size of the logical volume.

```
# lvreduce -L total_MB_in_lvolum /dev/vgXX/lvolY
```

Where:

Key	Operation
total_MB_ in_lvolum	Must be a smaller number than that found in step 3

5. Reduce the size of the physical file system.

```
# newfs /dev/vgXX/rlvolY
```

Where:

Key	Operation
Y	The integer logical volume number.

---

## ◆ Reduce the Size of an LVM Logical Volume

---

6. Mount the newly reduced logical volume using the mount command.

```
# mount -a
```

7. Backup LVM data structures (PVRA and VGRA).

```
# vgcfgbackup /dev/vgXX      (Automatic unless turned off with A -n
                             option)
```

### LVM Example:

#### Reducing the size of a Logical Volume.

The following example reduces a logical volume from 200 MB to 100 MB.

The commands assume the following:

The volume group is vg05

The logical volume to be reduced is lvol2

The logical volume contains /accounts and is in /etc/fstab

The following commands to accomplish the task:

```
# fbackup -f /dev/rmt/0m -0vHi /accounts
```

```
# umount /dev/vg05/lvol2
```

```
# lvreduce -L 100 /dev/vg05/lvol2
```

*Alternatively, the logical volume can be reduced from a specific physical volume as shown below.*

```
# lvreduce -L 100 /dev/vg05/lvol2 /dev/dsk/c0t6d0
```

```
# newfs /dev/vg05/rlvol2
```

```
# mount -a
```

If there is sufficient space then restore the data:

```
# frecover -f /dev/rmt/0m
```

```
# vgcfgbackup vg05      (Automatic unless turned off with A -n option)
```

## ◆ Remove an LVM Volume Group

---

1. Backup all user data in the volume to be removed.

(Use `fbbackup`, `cpio` or `tar` backup utilities as appropriate.)

2. Determine the names of the logical volumes residing in the volume groups to be removed.

```
# lvdisplay /dev/vgXX/lvol*
```

Where:

<b>Key</b>	<b>Operation</b>
XX	The integer number of the volume group associated with the logical volume to be reduced in size.

3. Unmount all logical volumes in the volume group to be removed.

```
# cd / ; umount /dev/vgXX/lvol*
```

4. Remove the logical volumes found in step 2 from the volume group.

```
# lvremove /dev/vgXX/lvolY
```

**NOTE:** Multiple logical volume paths (`/dev/vgXX/lvolY`) can be included in the same `lvremove` command line.

5. Determine if multiple physical volumes are associated with the volume group to be removed. This is indicated by the number of block device files listed for the volume group to be removed.

```
# strings /etc/lvmtab
```

6. If more than one physical volume (as indicated by the number of block device files listed for the volume group), **REMOVE ALL BUT ONE** of the physical volumes using the `vgreduce` command.

```
# vgreduce /dev/vgXX /dev/dsk/cXtYdZ
```

**NOTE:** Multiple physical volume paths (`/dev/dsk/cXtYdZ`) can be included in the same `vgreduce` command line.

7. Now, remove the volume group.

```
# vgremove /dev/vgXX
```

---

**◆ Moving an LVM Logical Volume**

---

**LVM Example:****\* Moving A Logical Volume.**

The following example moves a logical volume to a new disk that has more room.

The volume group is vg01

The logical volume to be moved is lvol1

The physical volume associated with lvol1 is /dev/rdisk/c0t1d0

The destination physical volume is /dev/rdisk/c0t2d0

The following commands will accomplish the task:

```
# pvcreate -f /dev/rdisk/c0t2d0
```

```
# vgextend /dev/vg01 /dev/dsk/c0t2d0
```

```
# pvmove -n /dev/vg01/lvol1 /dev/dsk/c0t1d0 /dev/dsk/c0t2d0
```

```
# vgcfgbackup vg01 (Automatic unless turned off with A -n option)
```

## ◆ Exporting and Importing an LVM Volume Group

1. Unmount any logical volumes associated with the volume group you wish to export.

```
# umount /dev/vgXX/lvol*
```

Where:

Key	Operation
XX	The integer number of the volume group to be exported.

2. Make the volume group unavailable.

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

3. Use `vgexport` to remove volume group information from `/etc/lvmtab`.

```
# vgexport -v /dev/vgXX
```

**NOTE:** If the logical volumes used specific names, instead of the defaults, and these need to be kept you will need to specify the `-m` option for `vgexport` followed by a filename:

```
# vgexport -v -m vgXXlvols /dev/vgXX
```

This file will need to be transferred to the other system by tape or network before the `vgimport` command is executed.

4. Using `vi` or `ed`, remove any lines in `/etc/fstab` that refer to logical volumes in the volume group to be exported.

5. Shut the system down.

```
# cd / ; shutdown -h 0
```

6. Detach the physical disks from the exported system. Physically attach them to the new host system. Check the bus addresses to assure no conflict, power the new disk(s) on, and boot the new host.

7. Create a new directory under `/dev` for the volume group you wish to import.

```
# mkdir /dev/vgYY
```

Where:

## ◆ Exporting and Importing an LVM Volume Group

<b>Key</b>	<b>Operation</b>
YY	The integer number of the volume group to be imported.

8. Make a group character device file for the volume group to be imported.

```
# mknod /dev/vgYY/group c 64 0xZZ0000
```

Where:

<b>Key</b>	<b>Operation</b>
ZZ	HEXADECIMAL group number.

9. Determine the logical unit number(s) for the new disks to be imported.

```
# ioscan -kfC disk
```

Note the logical unit number(s).

10. Now import the volume group.

```
# vgimport -v /dev/vgYY /dev/dsk/cXtYdZ
```

If multiple disks, include block device file for each disk being imported on this line.

**NOTE:** If a mapfile was created, specify its name as shown to keep the original logical volume names:

```
# vgimport -v -m vgXXlvols /dev/dsk/cXtYdZ
```

Where:

<b>Key</b>	<b>Operation</b>
YY	The integer number of the volume group to be imported.
T	The integer logical unit number of the imported disk drive as displayed in step 8 above.

11. Activate the newly imported volume group.

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

12. Make a mount-point-directory on root (/) for each logical volume in the newly imported volume group.

```
# mkdir /vgYYlvolM
```

## ◆ Exporting and Importing an LVM Volume Group

Where:

Key	Operation
M	The integer logical volume number.

13. Add a new line in /etc/fstab using vi or ed for the new logical volume(s).

```
/dev/vgYY/lvolM /vgYYlvolM hfs rw 0 P #/vgYYlvolM
```

Where:

Key	Operation
/dev/vgYY /lvolM	The full path name of the block device file for the logical volume to be mounted.
/vgYYlvolM	The full path name of the directory that the file system is to be mounted under.
hfs	Type (options): hfs - high performance file system, nfs- remote NFS file system, swap - swap file system, swapfs - dynamic swap file system, ignore - entry is ignored by mount and fsck.
rw	Options. (use default options, or comma separated list of options): ro - read only, rw - read/write (default), suid - Set-user-ID execution allowed, nosuid - Set-user-ID not allowed.
0	Backup frequency (set to 0).
P	Integer pass number P which determines the order in which fsck checks the file systems.
#	# begins the comment field.

14. Mount the new file system(s).

```
# mount -a
```

Where:

Key	Operation
-a	Attempts to mount everything in /etc/checklist.

15. Backup the LVM data structures (PVRA and VGRA) for all groups affected. The default path name for the backup will be /etc/lvmconf/vgYY.conf.

```
# vgcfbackup /dev/vgYY (Automatic unless turned off with A -n option)
```



---

## ◆ Adding Secondary Device Swap on a Logical Volume

---

Adding secondary swap space consists of :

- \* creating a logical volume
- \* adding a line to */etc/fstab*
- \* and finally, executing the *swapon* command.

A rough calculation of swap space requirements for a small standalone system can be made by determining the total size of the applications likely to be running at peak times and add this to the size of physical memory. A precise and detailed calculation can be made. To do this refer to the System Administration Tasks chapter titled Managing Swap Space and Dump Areas.

As a guideline, it is recommended that swap space be equal to 1.5 times the size of physical memory though this may be reduced for systems with very large amounts of memory.

Once the total swap size has been estimated or precisely calculated, proceed with the following:

1. Determine the current state of the system's swap space.

**# swapinfo**

The output of the *swapinfo* command will indicate the type of swap by location, how much of it is available, used, and free. If *hold* is displayed, it indicates how much space the system has reserved based on possible requirements of running processes.

2. Create a logical volume of sufficient size (when added to existing swap space) to bring total swap space to the required amount. Specify the size of the logical volume to be created in Megabytes following the *-L* parameter.

**# lvcreate -L SSS vgYY**

Where:

<b>Key</b>	<b>Operation</b>
SSS	The integer number of MB to be allocated for this logical volume.
YY	The integer number of the volume group to create the logical volume in.

3. Add a new line in */etc/fstab* using *vi* or *ed* for the new logical volume.

**/dev/vgYY/lvolZ /swap swap defaults 0 0**

---

## ◆ Adding Secondary Device Swap on a Logical Volume

---

Where:

Key	Operation
/dev/vgYY/lvolZ logical	The full path name of the block device file for the new swap volume.

4. Enable swapping on the new device.

```
# swapon -a
```

Where:

Key	Operation
-a	attempt to start swapping on everything listed in /etc/fstab.

**CAUTION:** If the logical volume previously contained a filesystem this will need to be destroyed. This can be done by using either of the following commands:

```
# swapon -f /dev/vgYY/lvolZ
```

or

```
# cp /stand/vmunix /dev/vgYY/rlvolZ
```

5. Verify that the new swap device is enabled.

```
# swapinfo
```

6. Backup the LVM data structures. The default path name for the backup will be /etc/lvmconf/vgYY.conf.

```
# vgcfgbackup /dev/vgYY (Automatic unless turned off with A -n option)
```

---

## ◆ Adding Dumps Devices on a Logical Volume

---

- \* **Dump areas must reside in the root volume group (normally vg00).**
- \* **The kernel must know where the dump areas are in the event of a system panic.**
- \* **The logical volumes used as dump areas must have contiguous physical extents.**
- \* **This must be done when the logical volume(s) are created with `lvcreate`.**

**NOTE:** If a swap logical volume is to be used for dump then it too must be in the root volume group and have been created with contiguous physical extents.

To configure dumps area(s), on logical volumes perform the following tasks:

1. Verify the current system dumps area configuration.

```
# lvlnboot -v
```

This command will display the current root, swap, and dumps devices.

2. If the dumps area has not been configured or if additional dumps area is required it will be necessary to locate or create contiguous logical volume(s) to be configured as dumps area. To determine if an existing logical volume was created in a contiguous manner use the `lvdisplay` command.

```
# lvdisplay /dev/vgYY/lvolZ
```

Look at the bottom line of the output. The Allocation field will indicate contiguous if the logical volume was created with contiguous physical extents. If this is not displayed, the logical volume cannot be used for dumps area.

**NOTE:** A logical volume can be converted from non-contiguous to contiguous using the `lvchange -C y` command, only if its physical extents match the constraints demanded by the contiguous allocation policy. If this is not the case, the logical volume must be removed, and if enough contiguous space is available, the logical volume can be created again using the `lvcreate -C y` option as shown later in this procedure.

3. Create a contiguous logical volume(s) of sufficient size (when added to any existing dumps area) to bring total dumps area size to an amount which is at least 5 MB more than installed physical memory.

```
# lvcreate -C y -L SSS vgYY
```

---

## ◆ Adding Dumps Devices on a Logical Volume

---

Where:

<b>Key</b>	<b>Operation</b>
SSS	An integer number indicating the size in MB of the logical volume to be created.
YY	The integer number of the volume group the logical volume will be associated with.

5. Update the Boot Data Reserved Area (BDRA) and LABEL file with the new dump locations.

```
# lvnboot -d /dev/vgYY/lvolZ
```

**NOTE:** Repeat the command for each logical volume to be added as a dumps device as done previously.

6. Generate a new kernel with the following line included in the kernel devices section of the /stand/system file if the following is not already present (it probably will be):

```
dump lvol
```

**NOTE:** This line may have to be added or an existing line modified using vi or ed.

7. Now, backup the LVM data structures. The default path name for the backup will be /etc/lvmconf/vgYY.conf.

```
# vgcfgbackup /dev/vgYY          (Automatic unless turned off with An
option)
```

8. The /sbin/rc1.d/S440savecore script saves kernel core dumps.

9. Reboot the computer to allow the kernel to configure the new dump areas.

```
# shutdown -r 0
```

---

## ◆ Booting a Damaged LVM Bootable Disk

---

The following procedure assumes that the Logical Interchange Format (LIF) sections of the bootable LVM disk is functioning to allow the user to load a kernel. If this is not so, use the Support Tools Media (Support Tape) to recover the system.

1. Attempt to boot the disk in LVM maintenance mode.

```
ISL> hpux -lm (;0)/stand/vmunix (try /stand/vmunix.BCKUP though not
always present)
```

The steps that follow will not apply in all situations, nor will the order of the commands presented be applicable in all cases. You must read and analyze error messages to arrive at a solution.

The following assumptions will apply to the commands that follow; the root volume group is vg00; and the root logical volume is lv011. If this is not the case, substitute the system's root volume group and root logical volume when typing the commands.

The key things to accomplish are:

- a. **Restore or update LVM data structures.**
  - b. **Make the root volume group available.**
2. Use `vgcfgrestore` to restore a current backup of LVM data structures (PVRA, VGRA, and BDRA) to vg00.

```
# vgcfgrestore -n vg00 /dev/rdisk/XtYdZ
```

Where: T is the logical unit number of the root disk.

3. Attempt to make the root volume group available.

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

4. The following will update LIF's LABEL file with information contained in the Boot Data Reserve Area.

```
# lvnboot -R /dev/vg00          Automatic unless turned off with A -n
option)
```

---

## ◆ Booting a Damaged LVM Bootable Disk

---

5. Make sure all LVM disks are powered, then use `vgscan` to rebuild the `/etc/lvmtab` file. Many LVM commands depend on the information contained in this file.

```
# mv /etc/lvmtab /etc/lvmtab.bk
```

```
# vgscan -v
```

Once the root volume group LVM data structures have been replaced or updated and the volume group is available, the system should boot normally.

---

**◆ Mirroring a Root Disk**

---

The following example will create a single mirror for the root and primary swap logical volumes. The commands assume the following:

- \* **The root volume group is vg00**
- \* **The root logical volume is lvol1**
- \* **The primary swap device is lvol2**
- \* **The root bootable system disk is Instance 0**
- \* **The mirror copy disk is Instance 1**

The following commands will accomplish the task:

```
# pvcreate -B /dev/rdisk/c0t1d0
# mkboot -l /dev/rdisk/c0t1d0
# mkboot -a "hpux (;0)/stand/vmunix" /dev/rdisk/c0t1d0
# vgextend /dev/vg00 /dev/dsk/c0t1d0
# lvextend -m 1 /dev/vg00/lvol1 /dev/dsk/c0t1d0
# lvextend -m 1 /dev/vg00/lvol2 /dev/dsk/c0t1d0
# lvolboot -r /dev/vg00/lvol1
# lvolboot -s /dev/vg00/lvol2
# lvolboot -R
# vgcfgbackup vg00
```

---

**◆ Lvsplit/lvmerge: Backing Up a Mirrored Disk**

---

The following example will split the root mirror created in the previous example, back up the data, and merge the two logical volumes back into the mirror.

Here are the commands to accomplish the backup and resynchronization:

Before proceeding, if the logical volume(s) contains a database, the database must be stopped before it is split. Otherwise, the data contained in the split half could be invalid and of no use if it needs to be restored from the backup at a later date.

```
# sync

# lvsplit -s backup /dev/vg00/lvol1

# fsck -p /dev/vg00/lvol1backup

# mkdir /lvol1backup

# mount /dev/vg00/lvol1backup /lvol1backup

# fbackup -f /dev/rmt/0h -0vHi /lvol1backup

# umount /lvol1backup

# lvmerge /dev/vg00/lvol1backup /dev/vg00/lvol1
```



---

**◆ Create a Boot Disk**

---

**WARNING:** Creating a Bootable Disk is useful for two specific cases:

- 1) Creating a mirror for the root logical volume
- 2) Creating a new root logical volume (moving root from one disk to an other)

The steps to **add a boot disk to the root volume group** are:

**1/** Create a Physical Volume (LVM Disk) `/dev/dsk/c0t6d0."B"` option allows the creation of a bootable disk.

```
# pvcreate -B /dev/rdisk/c0t6d0
```

**2/** Create the LIF area on the Boot Disk.

```
# mkboot -l /dev/rdisk/c0t6d0
```

**3/** Update the AUTO file of the LIF with the necessary command to automatically boot the system.

```
# mkboot -a "hpux (;0)/stand/vmunix" /dev/rdisk/c0t6d0
```

**4/** Add the new disk(`/dev/dsk/c0t6d0`) to the root volume group (`/dev/vg00`)

```
# vgextend /dev/vg00 /dev/dsk/c0t6d0
```

**5/** Check your job

```
# lvinboot -v
```

**End of Day 2 Reading.**

**Perform Lab Modules 10 - 17.**

---

◆ LVM Lab Exercises

---



---

**◆ LVM Lab Exercises**

---

**Objective:**

To assist the student in learning the basics of LVM file system management.

By successful completion of these lab exercises, the student will demonstrate the fundamental skills required to do common installation and maintenance tasks on LVM file systems.

**References:**

- \* HP-UX System Administration Tasks Manual.
- \* HP-UX 10.0 LOGICAL VOLUME MANAGER WHITE PAPER  
(Obtained from HP Supportline document id OALWP01950320).
- \* HP-UX Reference --Man Pages: Section (1M).
- \* Student Workbook.

**Key Commands Used:**

diskinfo	dmesg	extendfs	ioscan	lvcreate
lvchange	lvdisplay	lvextend	lvlnboot	lvreduce
lvremove	lvrmboot	mkboot	newfs	pvcreate
pvdisplay	strings	swapinfo	swapon	vgcfgbackup
vgcfgrestore	vgchange	vgcreate	vgdisplay	vgexport
vgextend	vgreduce	vgremove	vgscan	

---

**◆ LVM Lab Exercises**

---

**NOTE:** To reach the objective, these lab exercises are broken into the modules as shown below. As written in the introduction (page 2) of this workbook you must perform all the theory of each day before doing the labs.

**DAY 1**

- Module 1** Introduction.
- Module 2** Creating a Logical Volume.
- Module 3** Creating a Volume Group.
- Module 4** Extending a Logical Volume.
- Module 5** Reducing a Logical Volume.
- Module 6** Exporting/Importing a Volume Group.
- Module 7** Restoring Volume Group Structures.
- Module 8** Adding Secondary Swap Space.
- Module 9** Removing a Volume Group.

**DAY 2**

- Module 10** Extending a Volume Group.
- Module 11** Creating Dump Logical Volumes.
- Module 12** Mirroring a Logical Volume.
- Module 13** Reducing a Volume Group.
- Module 14** Mirroring the Boot disk.
- Module 15** Maintenance mode.
- Module 16** Creating /etc/lvmtab.
- Module 17** Logical Interchange Format.

**NOTE:** For the purpose of these lab exercises your LVM Root Disk will be designated as Root Disk; and the add-on disk drive will be designated as Disk1



---

**◆ Module 1 - Introduction**


---

**Objective:**

To build a picture of the current LVM configuration of your system.

**Commands:**

```
lvlnboot
diskinfo
ioscan
vgdisplay
```

**Tasks:**

1. Login as "root".
2. Type the following to obtain information about the root volume group:

```
# lvlnboot -v
```

Note the device file associated with the boot disk.

Device file:

2. Using *ioscan -kfC* disk and *diskinfo /dev/rdisk/cXtYdZ* (where Y is the Instance# obtained from *ioscan*'s output), complete the following table:

Disk	HP Model Number	I#	Hardware Path
Root Disk			
Disk 1			
Disk 2			

**NOTE:** For this entire LVM lab you will only need to use Disk 1 to work with.

---

**◆ Module 1 - Introduction**

---

3. Using `vgdisplay -v vg00` complete the following table:

Logical Volume	Size (Mb)	Number of P.E's
lvol1		
lvol2		
lvol3		
lvol4		
lvol5		
lvol6		
lvol7		
lvol8		



---

**◆ Module 2 – Creating a Logical Volume**

---

**Objective:**

To create a Logical Volume in an existing Volume Group.

**Commands:**

```
pvdisplay
lvcreate
lvdisplay
```

**Tasks:**

1. Execute the following command on your root disk:

```
# pvdisplay -v /dev/dsk/cAtBdC | more
```

This will display your entire root disk, showing you which Logical Volumes are assigned to what Physical Extents.

How many free PE's are there?

**Answer:**

2. Create a new logical volume in your root Volume Group (vg00) which uses these remaining Physical Extents.

**Command:**

3. Verify your success by displaying the Logical Volume information.

**Command:**

---

◆ **Module 2 – Creating a Logical Volume**

---

4. Back up the Volume Group structures! (This occurred already but lets practice it)

**Command:**

**NOTE:** At this point we have only created a Logical Volume name. It does not have a filesystem or mount point etc.

---

**◆ Module 3 – Creating a Volume Group**

---

**Objective:**

To add a new disk to an LVM system by creating a new Volume Group with two Logical Volumes.

**Commands:**

```
pvcreate
mkdir
mknod
vgcreate
vgchange
lvcreate
```

**Tasks:**

1. Using the reference documents, create a new volume group, `vg01`, and two 20Mb logical volumes: `lv_noncontig` and `lv_contig` on Disk1 (`/dev/dsk/cXtYdZ`) (make `lv_contig` contiguous by using the `-C y` option with the `lvcreate` command)

Add the Logical Volumes to `/etc/fstab`, mount them, and copy a file to each new logical volume.

Record all commands and arguments required to accomplish this in the space provided below.

**NOTE:** You might need to force the `pvcreate` with the `-f` option if a PVRA already exists on the disk.

**Commands:**

---

**◆ Module 3 – Creating a Volume Group**

---

2. Use the following commands to have a look at your newly created Volume Group and Logical Volumes:

```
# pvdisplay -v /dev/dsk/cXtYdZ(substitute for X, Y and Z)
# vgdisplay -v vg01
# lvdisplay -v /dev/vg01/lv_noncontig
# lvdisplay -v /dev/vg01/lv_contig
# bdf
# mount -a
# bdf
```

What is the full path of the file that was created as a result of the *vgcfgbackup* command automatically running?

**Answer:**

What is the size of *lv\_contig* & *lv\_noncontig*?

**Answer:**

---

◆ **Module 4 – Extending a Logical volume**

---

**Objective:**

To extend an existing Logical Volume.

**Commands:**

lvextend  
mount/umount  
extendfs

**Tasks:**

1. Using the reference documents, extend /dev/vg01/lv\_noncontig to 100Mb.

---

**REMEMBER:** You must specify the TOTAL new size!

---

Record all commands and arguments required accomplishing this in the space provided below.

Use the commands from Module 2, Task 3, above to verify your success.

**Commands:**

---

◆ **Module 4 – Extending a Logical volume**

---

2. Try to extend `/dev/vg01/lv_contig` by 12 Mb.

**Commands:**

What error message did you get?

**Answer:**

What is the reason for this?

**Answer:**

3. Use the `lvchange` command to make `lv_contig` non-contiguous and try again.

**Commands:**

Now remount `lv_contig`.

Were you successful this time?

**Answer:**

---

**◆ Module 5 – Reducing a logical volume**

---

**Objective:**

To reduce the size of an existing Logical Volume.

**Commands:**

lvreduce  
mount/umount  
newfs

**Tasks:**

1. Using the reference documentation reduce the size of */dev/vg01/lv\_noncontig* by 50 MB.

---

**NOTE:** Normally such a reduction would require backing up user data. We will skip that step as no critical data actually exists on this logical volume.

---

Record all commands and arguments required accomplishing this in the space below.

Remount *lv\_noncontig* when finished with *newfs* command

Use the *bdf* command to verify your success.

---

**REMEMBER:** After reducing the size of the logical volume, you must execute the *newfs* command to successfully reduce the size of the filesystem.

---

**Commands:**

---

**◆ Module 6 – Adding Secondary Swap Space**

---

**Objective:**

To make an existing Logical Volume a "Secondary Swap" LV.

**Commands:**

```
swapinfo  
swapon
```

**Tasks:**

1. Type the following to observe the current state of the system's swap device(s):

```
# swapinfo
```

Which logical volume is being used as the primary swap device?

**Answer:**

2. Since `/dev/vgnew/lv_contig` should already exist, use this space for secondary swap. To accomplish this, perform the following steps:

- a. *Unmount `lv_contig`*

- b. Edit `/etc/fstab` and change the line including `/dev/vgnew/lv_contig` to:

```
/dev/vgnew/lv_contig /swap swap 0 0
```

- c. Remove all traces of the previously mounted file system by copying a large file to `/dev/vgnew/lv_contig` by typing the following:

```
# cp /stand/vmunix /dev/vgnew/lv_contig
```

- d. Turn on swapping by typing the following:

```
# swapon -a
```



---

◆ **Module 6 – Adding Secondary Swap Space**

---

3. Verify that secondary swap has been enabled by using the *swapinfo* command.

How do you disable swap once it has been enabled?

**Answer:**

---

**◆ Module 7 – Removing a Volume Group**

---

**Objective:**

To remove all traces of a Volume Group from an LVM system.

**Commands:**

*lvremove*  
*vgremove*

**Tasks:**

1. Using the reference documentation, remove all traces of *vgnew* from the system.

Record all commands and arguments required to accomplish this in the space provided below. Do not forget to remove the mount information from */etc/fstab*

**Commands:**

---

**◆ Module 8– Extending a Volume Group**

---

**Objective:**

To extend an existing Volume Group to a new disk.

---

**NOTE: Extending the Root Volume Group is only recommended for mirroring**

---

**Commands:**

*pvcreate*  
*vgextend*  
*lvlnboot*

**Tasks:**

1. Using the reference documentation, extend Volume Group `vg00` to include `disk1`:

---

**NOTE:** You may need to force the *pvcreate* command with the *-f* option because the disk already has a PVRA from the previous labs.

---

Note the two steps required below.

**Commands:**

2. Execute the following command to have a look at the current contents of the BDRA:

*# lvlnboot -v*

---

**◆ Module 9 – Creating a Dump Logical volume**

---

**Objective:**

To create a Dump Logical Volume in an existing Volume Group.

**Commands:**

*lvlnboot*

**Tasks:**

1. The amount of disk space available for core dumps needs to accommodate the amount of core memory. To find the required size of the dumps devices, we need to determine the size of physical memory. To do so, type the following:

**# dmesg | grep Physical**

How many Mb of dumps area does the system need to obtain a complete core dump?

**Answer:**

2. Now, create a contiguous logical volume on Disk1 called "dump" with the necessary space required for a memory core dump. Write down the command in the space provided below.

**Command:**

---

**◆ Module 9 – Creating a Dump Logical volume**

---

3. The root volume group's configuration is automatically saved in `/etc/lvmconf/vg00.conf` but need to update the BDRA with `lvlnboot`.

**Commands:**

4. Review the `lvlnboot` command's function and arguments in the on-line manual, then answer the following questions:

Under what conditions should `lvlnboot` be invoked?

**Answer:**

What does `lvlnboot` do to the Boot Data Reserved Area (BDRA)?

**Answer:**

What could be the result of failure to update the BDRA after making changes to the root volume group?

**Answer:**

What are `lvlnboot`'s options for updating the BDRA?

**Answer:**

---

◆ **Module 9 – Creating a Dump Logical volume**

---

When will the *lvlnboot* commands take effect?

**Answer:**



---

**◆ Module 10 – Mirroring a Logical Volume**

---

**Objective:**

To mirror an existing Logical Volume and to perform an On-Line backup.

**Commands:**

*lvextend*

*lvsplit*

*lvmerge*

**Tasks:**

1. On your root disk you should have *lv19* which you created in Module 2, Task 2.
2. Using your workbook and the cookbooks, mirror *lv19* to *disk1* which is now part of your root Volume Group, and should have enough free Physical Extents.  
Note how long it takes.

**Command:****Time:**

3. Verify your success with the *lvdisplay* command.

**Command:**

From the result of the previous command, what two areas highlight that a Logical Volume is in fact being mirrored?

4. Remove "lv19".



---

◆ **Module 10 – Mirroring a Logical Volume**

---

5. Create a very small Logical Volume (Name: "lvmirror"; size: 4 MB) on your root disk .

**Command:**

Try to mirror "lvmirror" to your root disk.

**Command:**

What error message did you get?

**Answer:**

6. Mirror "lvmirror" to disk1 .

**Command:**

Extend mirror to use the remaining Physical extents on your root disk.

**Command:**

How long did that operation take?

**Time:**

---

**◆ Module 10 – Mirroring a Logical Volume**

---

7. Make a file system on "lvmirror", mount it under */mirror* and copy some files to it.  
Perform an on-line backup of mirror (Split the mirror, mount the off-line half to */backup*, run your favorite backup program, merge the mirror). Write down the commands you need:

**Commands:**

8. Run *lvdisplay -v* on "lvmirror".

**Command:**

Split "lvmirror".

**Command:**

Run *lvdisplay -v /dev/vg00/lvmirror* again.

---

◆ **Module 10 – Mirroring a Logical Volume**

---

Compare the *lvdisplay* outputs.  
What has changed?

**Answer:**

9. Try to merge the mirror using the following command:

```
# lvmerge /dev/vg00/lvmirror /dev/vg00/lvmirrorb
```

What happens?

**Answer:**

---

**◆ Module 11 – Reducing a Volume Group**

---

**Objective:**

To reduce an existing Volume Group to one physical volume.

**Commands:**

*vgreduce*

**Tasks:**

1. Remove the Logical Volume "lvmirror".

**Commands:**

Don't forget to remove it from */etc/fstab* as well.

2. Use the following commands to make lvol2, both the Primary Swap and Dump logical volume and to remove the dump logical volume.

*Commands: # lvrmbboot -d dump /dev/vg00*  
*# lvlnboot -d /dev/vg00/lvol2*  
*# lvremove /dev/vg00/dump*

3. Reduce disk1 from the root Volume Group (vg00).

**Command:**

---

**◆ Module 12 – Mirroring the Boot Disk**

---

**Objective:**

To mirror the Boot disk.

**Commands:**

```
pvcreate -B  
mkboot  
lvextend  
lvlnboot
```

**Tasks:**

1. Make disk1 a boot disk using the following commands:

Commands:

```
# pvcreate -f -B /dev/rdisk/cXtYdZ (make substitutions for X Y,  
    & Z)  
# mkboot /dev/rdisk/cXtYdZ  
# mkboot -a "hpux (;0)/stand/vmunix" /dev/rdisk/cXtYdZ
```

2. Extend the root Volume Group to disk1.

**Command:**

3. Mirror the Root and Swap Logical Volumes to disk1 using the following commands:

**Commands:**

```
# lvextend -m 1 /dev/vg00/lvol1 /dev/dsk/cXtYdZ  
# lvextend -m 1 /dev/vg00/lvol2 /dev/dsk/cXtYdZ
```

4. Verify the changes with the *lvlnboot -v* command.

**Command:**

---

**◆ Module 13 – Exporting/Importing a Volume Group**

---

**Objective:**

To get familiar with the commands *vgexport* and *vgimport* and to understand how Instance Numbers for disks are configured in the file */etc/lvmtab*.

**Commands:**

vgexport  
vgimport  
vgchange

**Tasks:**

1. Type *strings /etc/lvmtab* to look at the contents of this configuration file.

What is the usage of */etc/lvmtab*?

**Answer:**

Why do you have to use the command *strings* to list its contents?

**Answer:**

2. The goal of this lab is to export a Volume Group (remove the Volume Group structures from the system but not from its Physical Volume) and then to import its Physical Volume(s) again to a new Volume Group.

This is very often used to move a Volume Group from one system to another but can also be used to fix LVM problems.

To be able to export a Volume Group it must be made inactive using the *vgchange* command.

To deactivate a Volume Group all its filesystems must be unmounted and any swap Logical Volumes deactivated. Otherwise you will get the message:

*vgchange: Couldn't deactivate volume group /dev/vg01: Device busy*

---

**◆ Module 13 – Exporting/Importing a Volume Group**

---

*umount* the two Logical Volumes (*lv\_contig* and *lv\_noncontig*), then deactivate Volume Group *vg01*. Write down the major steps required:

**Commands:**

3. What error message do you get if you try to mount a filesystem of a deactivated Volume Group?

Try *mount -a* just to see the error message:

**Error message:**

4. Export Volume Group *vg01* from your system. Why do you need to use option *-m*? Write down all the major steps required:

**Commands:**

The *-m* option creates a file that contains a list of the names of the Logical Volumes in the Volume Group being exported.

5. Have a look at */etc/lvmtab*. How is it different compared to Task 1?

**Answer:**

---

**◆ Module 13 – Exporting/Importing a Volume Group**

---

6. *vgexport* in Task 2 removed the volume group *vg01* from your system but didn't modify its physical volume information (on disk). Create a new volume group called *vgnew* and import the previously exported physical volume to it. Write down the major steps required, shutdown and reboot your system at the end of this lab to verify your success.

**Commands:**

---

**REMEMBER:** *vi /etc/fstab* and change the *vg01* entries to *vgnew*. If you change your mount point name in *fstab* then don't forget to make new directories under root for the new mount points.

---

What would happen if you did not specify option *-m*?

**Answer:**



---

**◆ Module 14 - Rebuilding Volume Group Information**

---

**Objective:**

To recover a Volume Group's structures (PVRA & VGRA) using the *vgcfgrestore* command.

**Commands:**

*vgcfgbackup*  
*vgcfgrestore*  
*vgchange*

**Tasks:**

1. Backup Volume Group *vgnew*'s structures (PVRA & VGRA).

**Command:**

2. Deactivate Volume Group *vgnew*.

**Commands:**

3. Restore Volume Group *vgnew*.

**Command:**

4. Reactivate Volume Group *vgnew* and remount the filesystems

**Command:**

---

**◆ Module 15 – Maintenance Mode**

---

**Objective:**

To learn how to troubleshoot BDRA and LABEL file problems.

**Commands:**

*lvrmboot*

*lvlnboot*

*lifrm*

**Tasks:**

1. Execute the following commands to verify that the BDRA and LABEL file are correct for your root disk. Substitute for A, B, and C as appropriate for root disk.

*# lvlnboot -v* (to list the BDRA)

*# lifls -l /dev/rdisk/cAtBdC* (to list the LIF boot volume)

2. Remove the LABEL file which is part of the LIF boot volume by executing the following command:

*# lifrm /dev/rdisk/cAtBdC:LABEL*

*# lifls -l /dev/rdisk/cAtBdC* (To verify)

3. Shutdown and reboot your system.

What error message do you get?

**Answer:**

4. To fix the problem execute the following command to boot in maintenance mode:

*ISL> hpux -lm (;0)/stand/vmunix*

The system boots to single user mode without activating LVM.

---

◆ **Module 15 – Maintenance Mode**

---

5. Activate the root Volume Group.

**Command:**

6. Create a new LABEL file and update it by copying the root and swap device information contained in the BDRA to it:

**Command:**

7. Mount /usr to have the *lifs* command available.

Was the LABEL file created by *lvlnboot -R* ?  **YES**  **NO**

The problem should be fixed but you are still in maintenance mode.  
Should you boot from maintenance mode into multiuser mode  
using init 2 ?  **YES**  **NO**

---

**◆ Module 16 – Creating */etc/lvmtab***

---

**Objective:**

To create the */etc/lvmtab* file.

**Commands:**

*vgscan*

**Tasks:**

1. The */etc/lvmtab* file is created by executing the command *vgscan*.

---

**NOTE:** *vgscan* will NOT recreate */etc/lvmtab* if it already exists. Therefore you must rename (*mv*) or remove (*rm*) */etc/lvmtab* before executing *vgscan*.

---

Type the following to see the current contents of */etc/lvmtab*:

*# strings /etc/lvmtab*

2. Rename */etc/lvmtab* to */etc/lvmtab.bk*

**Command:**

3. Type the following to create and rebuild */etc/lvmtab*:

*# vgscan -v*

4. Examine the contents of */etc/lvmtab* with the strings command.

Is the result the same as that of Step 1?

---

**◆ Module 17 - Rebuilding the LVM Root Disk's Logical Interchange Format (LIF) Utilities**

---

**Objective:**

To remove and recreate an LVM boot disks LIF area.

**Commands:**

*rmboot*  
*mkboot*

**Tasks:**

1. Briefly review the on-line manual entry for *mkboot*, then answer the following questions:

What *mkboot* option should be used to create or update the AUTO file

**Answer:**

2. Execute the following command to display the current contents of the LIF area.

**# lifls -l /dev/rdisk/cAtBdC** (for root disk)

3. Execute the following command to see the current contents of LIF's "AUTO" file.

**# lifcp /dev/rdisk/cAtBdC:AUTO -**

What does it contain?

**Answer:**

---

**◆ Module 17 - Rebuilding the LVM Root Disk's Logical Interchange Format (LIF) Utilities**

---

4. Using shutdown, go to single-user mode, then mount all file systems.

**Commands:**

5. Remove the LIF area with the "*rmboot*" command and verify your success by executing the *lifs* command used in Task 2.

**Command:**

Try executing the *lifs -l* command you will get the message:

*lifs: Can't list /dev/rdisk/cAtBdC; not a LIF volume.*

6. Use *mkboot* to recreate the Root Disk's LIF area.

**Command:**

7. Update LIF's AUTO file with the correct boot string for your system.

**Command:**

8. Update the LIF area with *lvlnboot -R*.

**Command:**

Why is this necessary?

**Answer:**

9. Reboot to verify your success.

◆ **Module 17 - Rebuilding the LVM Root Disk's Logical Interchange Format (LIF) Utilities**

---

**WARNING : this will be your last lab. This is an interesting lab but it is possible the system goes in Panic .No patche is available to resolve that issue for the moment.**

---

- 10** Now remove the root and primary swap logical volume information from the BDRA by executing the following command:

```
# lvrmbboot -r /dev/vg00
```

**NOTE:** This also clears the information contained in the LABEL file but does not delete the file.

---

- 11** Display the BDRA information:

**Command:**

What message do you get?

**Answer:**

Display the LIF boot volume information:

**Command:**

---

◆ **Module 17 - Rebuilding the LVM Root Disk's Logical Interchange Format (LIF) Utilities**

---

---

**NOTE:** The LABEL file should be there. But it does not contain any information because we removed the info at the same time as we removed the info from the BDRA.

---

**12** Shutdown and reboot the system.

What error message do you get?

**Answer:**

**13** Boot in maintenance mode.

**Command:**

**14** Activate the root Volume Group.

**Command:**

**15** Update the BDRA and LABEL file with *lvlnboot* so that it contains the correct root and primary swap information:

**Command:**

**16** Reboot to verify.



This page is left blank intentionally

◆ LVM Lab Exercises Answers



---

**◆ LVM Lab Exercises Answers**

---

**Objective:**

To assist the student in learning the basics of LVM file system management.

By successful completion of these lab exercises, the student will demonstrate the fundamental skills required to do common installation and maintenance tasks on LVM file systems.

**References:**

- \* HP-UX System Administration Tasks Manual.
- \* HP-UX 10.0 LOGICAL VOLUME MANAGER WHITE PAPER (Obtained from HP Supportline document id OALWP01950320).
- \* HP-UX Reference --Man Pages: Section (1M).
- \* Student Workbook.

**Key Commands Used:**

diskinfo	dmesg	extendfs	ioscan	lvcreate
lvchange	lvdisplay	lvextend	lvlnboot	lvreduce
lvremove	lvrmboot	mkboot	newfs	pvcreate
pvdisplay	strings	swapinfo	swapon	vgcfgbackup
vgcfgrestore	vgchange	vgcreate	vgdisplay	vgexport
vgextend	vgreduce	vgremove	vgscan	

---

◆ **LVM Lab Exercises Answers**

---

**NOTE:** To accomplish the objective, these lab exercises are broken into the modules as shown below:

- Module 1** Introduction.
- Module 2** Creating a Logical Volume.
- Module 3** Creating a Volume Group.
- Module 4** Extending a Logical Volume.
- Module 5** Reducing a Logical Volume.
- Module 6** Exporting/Importing a Volume Group.
- Module 7** Restoring Volume Group Structures.
- Module 8** Adding Secondary Swap Space.
- Module 9** Removing a Volume Group.
- Module 10** Extending a Volume Group.
- Module 11** Creating Dump Logical Volumes.
- Module 12** Mirroring a Logical Volume.
- Module 13** Reducing a Volume Group.
- Module 14** Mirroring the Boot disk.
- Module 15** Maintenance mode.
- Module 16** Creating /etc/lvmtab.
- Module 17** Logical Interchange Format.

**NOTE:** For the purpose of these lab exercises your LVM Root Disk will be designated Root Disk; and the add-on disk drive will be designated Disk1

---

## ◆ Module 1 - Introduction

---

**Objective:**

To build a picture of the current LVM configuration of your system.

**Commands:**

```
lvmboot
diskinfo
ioscan
vgdisplay
```

**Tasks:**

1. Login as "root".
2. Type the following to obtain information about the root volume group:

```
# lvmboot -v
```

Note the device file associated with the boot disk.

Device file: **/dev/dsk/cAtBdC**

2. Using *ioscan -kfC disk* and *diskinfo /dev/rdisk/cAtBdC* (where B is the Instance# obtained from *ioscan*'s output), complete the following table:

Disk	HP Model Number	I#	Hardware Path
Root Disk	<b>/dev/dsk/cAtBdC</b>	<b>B</b>	<b>8/4.B.0</b>
Disk 1	<b>/dev/dsk/cXtYdZ</b>	<b>Y</b>	<b>8/4.Y.0</b>
Disk 2	<b>/dev/dsk/cUtVdW</b>	<b>V</b>	<b>8/4.V.0</b>

---

**◆ Module 1 - Introduction**

---

3. Using `vgdisplay -v vg00` complete the following table:

Logical Volume	Size (Mb)	Number of PE's
lvol1	<b>48</b>	<b>12</b>
lvol2	<b>512</b>	<b>128</b>
lvol3	<b>84</b>	<b>21</b>
lvol4	<b>32</b>	<b>8</b>
lvol5	<b>20</b>	<b>5</b>
lvol6	<b>280</b>	<b>70</b>
lvol7	<b>332</b>	<b>83</b>
lvol8	<b>500</b>	<b>125</b>

---

**◆ Module 2 – Creating a Logical Volume**

---

**Objective:**

To create a Logical Volume in an existing Volume Group.

**Commands:**

```
pvdisplay
lvcreate
lvdisplay
```

**Tasks:**

1. Execute the following command on your root disk:

```
# pvdisplay -v /dev/dsk/cAtBdC | more
```

This will display your entire root disk, showing you which Logical Volumes are assigned to what Physical Extents.

How many free PE's are there?

**Answer: This depends on which system you are running. If, for example your boot disk is ST32171W you should have 55 free Physical extents.**

2. Create a new logical volume in your root Volume Group (vg00) which uses these remaining Physical Extents.

**Command:       # lvcreate -l 56 vg00**

3. Verify your success by displaying the Logical Volume information.

**Command:       # lvdisplay -v /dev/vg00/lvol9**

---

◆ **Module 2 – Creating a Logical Volume**

---

4. Back up the Volume Group structures! (This occurred already but lets practice it)

**Command:**        **# vgcfgbackup vg00**

**NOTE:** At this point we have only created a Logical Volume name. It does not have a filesystem or mount point etc.



---

**◆ Module 3 – Creating a Volume Group**

---

**Objective:**

To add a new disk to an LVM system by creating a new Volume Group with two Logical Volumes.

**Commands:**

```
pvcreate
mkdir
mknod
vgcreate
vgchange
lvcreate
```

**Tasks:**

1. Using the reference documents, create a new volume group, `vg01`, and two 20Mb logical volumes: `lv_noncontig` and `lv_contig` on Disk1 (`/dev/dsk/cXtYdZ`) (make `lv_contig` contiguous by using the `-C y` option with the `lvcreate` command)

Add the Logical Volumes to `/etc/fstab`, mount them, and copy a file to each new logical volume.

Record all commands and arguments required accomplishing this in the space provided below.

**NOTE:** You might need to force the `pvcreate` with the `-f` option if a PVRA already exists on the disk.

---

**◆ Module 3 – Creating a Volume Group**


---

```

Commands:      # pvcreate (-f) /dev/rdisk/cXtYdZ
                  # mkdir /dev/vg01
                  # mknod /dev/vg01/group c 64 0x010000
                  # vgcreate /dev/vg01 /dev/dsk/c0t0d0
                  # lvcreate -L 20 -n lv_noncontig vg01
                  # lvcreate -L 20 -C y -n lv_contig vg01
                  # newfs /dev/vg01/rlv_noncontig
                  # newfs /dev/vg01/rlv_contig

```

Finally , make mount point directories and add the two new file systems to /etc/fstab and mount them.

```

i.e.           # mkdir /vg01_lv_noncontig /vg01_lv_contig
                  # vi /etc/fstab

```

```

/dev/vg01/lv_noncontig /vg01_lv_noncontig hfs defaults 0 1
/dev/vg01/lv_contig /vg01_lv_contig hfs 0 1

```

( instead of hfs you can enter vxfs .It depends on what is the default (newfs use the default).To check the default you will need to lauch the command

```

# more /etc/default/fs
LOCAL=hfs

```

2.Use the following commands to have a look at your newly created Volume Group and Logical Volumes:

```

# pvdisplay -v /dev/dsk/cXtYdZ | more      (substitute for X,
Y and Z)
# vgdisplay -v vg01
# lvdisplay -v /dev/vg01/lv_noncontig
# lvdisplay -v /dev/vg01/lv_contig
# bdf
# mount -a
# bdf

```

---

◆ **Module 3 – Creating a Volume Group**

---

What is the full path of the file which was created as a result of the *vgcfgbackup* command automatically running?

**Answer: /etc/lvmconf/vg01.conf**

What is the size of *lv\_contig* & *lv\_noncontig*?

**Answer: Again this will vary depending on which system you are using .If your disk is a ST32550W then your logical volume be about 19861Kb in size.**

---

**◆ Module 4 – Extending a Logical volume**

---

**Objective:**

To extend an existing Logical Volume.

**Commands:**

```
lvextend
mount/umount
extendfs
```

**Tasks:**

1. Using the reference documents, extend /dev/vg01/lv\_noncontig to 100Mb.

**REMEMBER:** You must specify the TOTAL new size!

Record all commands and arguments required accomplishing this in the space provided below.

Use the commands from Module 2, Task 3, above to verify your success.

```
Commands:      # umount /dev/vg01/lv_noncontig
                  # lvextend -L 100 /dev/vg01/lv_noncontig
                  # extendfs /dev/vg01/rlv_noncontig
                  # mount /dev/vg01lv_noncontig
                                      /vg01_lv_noncontig
                  # vgcfgbackup vg01
```

---

**◆ Module 4 – Extending a Logical volume**

---

2. Try to extend `/dev/vg01/lv_contig` by 12 Mb.

**Commands:**        `# umount /dev/vg01/lv_contig`  
                      `# lvextend -L 32 /dev/vg01/lv_contig`

What error message did you get?

**Answer:**    *lvextend: Not enough free physical extents available.  
Logical volume /dev/vg01/lv\_contig could not be  
extended.  
Failure possibly caused by contiguous allocation  
policy.  
Failure possibly caused by strict allocation policy.*

What is the reason for this?

**Answer:**    *lv\_contig is contiguous and there are no free PE's  
after LV*

3. Use the `lvchange` command to make `lv_contig` non-contiguous and try again.

**Commands:**        `# lvchange -C n /dev/vg01/lv_contig`  
                      `# lvextend -L 32 /dev/vg01/lv_contig`  
                      `# extenfs /dev/vg01/rlv_contig`

Now remount `lv_contig`.

Were you successful this time?

**Answer:** *Yes*

---

**◆ Module 5 – Reducing a logical volume**

---

**Objective:**

To reduce the size of an existing Logical Volume.

**Commands:**

```
lvreduce
mount/umount
newfs
```

**Tasks:**

1. Using the reference documentation reduce the size of */dev/vg01/lv\_noncontig* by 50 Mb.

---

**NOTE:** Normally such a reduction would require backing up user data. We will skip that step as no critical data actually exists on this logical volume.

---

Record all commands and arguments required to accomplish this in the space below.

Remount *lv\_noncontig* when finished with *newfs* command

Use the *bdf* command to verify your success.

---

**REMEMBER:** After reducing the size of the logical volume, you must execute the *newfs* command to successfully reduce the size of the filesystem.

---

```
Commands:      # umount /dev/vg01/lv_noncontig
                # lvreduce -L 50 /dev/vg01/lv_noncontig
                # newfs /dev/vg01/rlv_noncontig
```

Remount *lv\_noncontig*:

```
e.g. # mount /dev/vg01/lv_noncontig /vg01_noncontig
      # bdf
```

---

**◆ Module 6 – Adding Secondary Swap Space**

---

**Objective:**

To make an existing Logical Volume a "Secondary Swap" LV.

**Commands:**

```
swapinfo  
swapon
```

**Tasks:**

1. Type the following to observe the current state of the system's swap device(s):

```
# swapinfo
```

Which logical volume is being used as the primary swap device?

**Answer: /dev/vg00/lvol2**

2. Since `/dev/vgnew/lv_contig` should already exist, use this space for secondary swap. To accomplish this, perform the following steps:

- a. *Unmount `lv_contig`*
- b. Edit `/etc/fstab` and change the line including `/dev/vgnew/lv_contig` to:

```
/dev/vgnew/lv_contig    /swap swap 0 0
```

---

**NOTE:** The BDRA is aware of Primary Swap. For this reason the entry for Primary Swap in `/etc/fstab` is "ignore". Any further Secondary Swap is "swap"

---

- c. Remove all traces of the previously mounted file system by copying a large file to `/dev/vgnew/lv_contig` by typing the following:

```
# cp /stand/vmunix /dev/vgnew/lv_contig
```

---

**◆ Module 6 – Adding Secondary Swap Space**

---

- d. Turn on swapping by typing the following:
- ```
# swapon -a
```
3. Verify that secondary swap has been enabled by using the *swapinfo* command.

How do you disable swap once it has been enabled?

**Answer: You have to remove it's entry from */etc/fstab* and reboot.  
There is no on-line method for turning off swap**



---

**◆ Module 7 – Removing a Volume Group**

---

**Objective:**

To remove all traces of a Volume Group from an LVM system.

**Commands:**

```
lvremove  
vgremove
```

**Tasks:**

1. Using the reference documentation, remove all traces of *vgnew* from the system.

Record all commands and arguments required accomplishing this in the space provided below. Do not forget to remove the mount information from */etc/fstab*

**Commands:**      **# vi /etc/fstab**

**Comment out or remove the entries for the Logical Volumes in *vgnew*.**

---

**NOTE : You must reboot the system to disable swap so that the logical volume can be removed**

---

```
# shutdown -r 0  
# lvremove /dev/vgnew/lv*
```

---

**NOTE : You will get a warning that a file system exists, say yes and remove**

---

```
# vgremove /dev/vgnew  
# rm -r /dev/vgnew
```

---

**◆ Module 8 – Extending a Volume Group**

---

**Objective:**

To extend an existing Volume Group to a new disk.

---

**NOTE: Extending the Root Volume Group is only recommended for mirroring**

---

**Commands:**

*pvcreate*  
*vgextend*  
*lvlnboot*

**Tasks:**

1. Using the reference documentation, extend Volume Group `vg00` to include `disk1`:

**NOTE:** You may need to force the *pvcreate* command with the *-f* option because the disk already has a PVRA from the previous labs.

Note the two steps required below.

**Commands:**      **# pvcreate (-f) /dev/rdisk/cXtYdZ**  
                  **# vgextend vg00 /dev/dsk/cXtYdZ**

2. Execute the following command to have a look at the contents of the BDRA:

***# lvlnboot -v***

---

**◆ Module 9 – Creating a Dump Logical volume**

---

**Objective:**

To create a Dump Logical Volume in the Root Volume Group.

**Command:**

*lvlnboot*

**Tasks:**

1. The amount of disk space available for core dumps needs to accommodate the amount of core memory. To find the required size of the dumps devices, we need to determine the size of physical memory. To do so, type the following:

**# dmesg | grep Physical**

How many Mb of dumps area does the system need to obtain a complete core dump?

**Answer: The guideline is physical memory +30 Mb  
If the system has 64 Mb of memory, this will be 64+  
30Mb.**

2. Now, create a contiguous logical volume on Disk1 called "dump" with the necessary space required for a memory core dump. Write down the command in the space provided below.

**Command: # lvcreate -L 70 -C y -r n -n dump vg00**

---

**NOTE: 70Mb is an example. It might be different in your case**

---

---

**◆ Module 9 – Creating a Dump Logical volume**


---

3. The root volume group's configuration is automatically saved in `/etc/lvmconf/vg00.conf` but you need to update the BDRA with `lvlnboot`.

**Commands:**        `# lvlnboot -d /dev/vg00/dump`

4. Review the `lvlnboot` command's function and arguments in the on-line manual, then answer the following questions:

Under what conditions should `lvlnboot` be invoked?

**Answer: When a change is made to the root Volume Group such as the addition of a dump logical volume**

What does `lvlnboot` do to the Boot Data Reserved Area (BDRA)?

**Answer: it updates it with the definitions of root , swap and dump logical volumes.**

What could be the result of failure to update the BDRA after making changes to the root volume group?

**Answer: The system may fail to boot, or a physical volume or dump logical volume would not be configured during a normal boot of example.**

What are `lvlnboot`'s options for updating the BDRA?

**Answer**        `# lvlnboot -d .....(for boot)`  
                   `# lvlnboot -r .....(for root)`  
                   `# lvlnboot -s .....( for swap)`  
                   `# lvlnboot -d .....(for dump)`

---

◆ **Module 9 – Creating a Dump Logical volume**

---

**# lvinboot -R (To recover any missing links to the  
LV's)**  
**# lvinboot -v (To see the current contents of the  
BDRA)**

When will the *lvinboot* commands take effect?

**Answer: Once the command has been executed. You don't have to  
reboot.**

---

**◆ Module 10 – Mirroring a Logical Volume**

---

**Objective:**

To mirror an existing Logical Volume and to perform an On-Line backup.

**Commands:**

*lvextend*  
*lvsplit*  
*lvmerge*

**Tasks:**

1. On your root disk you should have *lvol9* which you created in Module 2, Task 2.
2. Using your workbook and the cookbooks, mirror *lvol9* to *disk1* which is now part of your root Volume Group, and should have enough free Physical Extents.  
Note how long it takes.

**Command: # *lvextend -m 1 /dev/vg00/lvol9 /dev/dsk/cXtYdZ***

**Time: Should take a few minutes**

3. Verify your success with the *lvdisplay* command.

**Command: # *lvdisplay -v /dev/vg00/lvol9***

From the result of the previous command, what two areas highlight that a Logical Volume is in fact being mirrored?

**a)Mirror copies 1.**

**b)Two lists of LE to PE mappings.**

4. Remove “*lvol9*”

**Command: # *lvremove /dev/vg00/lvol9***

---

**◆ Module 10 – Mirroring a Logical Volume**

---

5. Create a very small Logical Volume (Name: "lvmirror"; size: 4 MB) on your root disk .

**Command: # lvcreate -l 1 -n lvmirror vg00**

Try to mirror "lvmirror" to your root disk.

**Command: lvextend -m 1 /dev/vg00/lvmirror /dev/dsk/cAtBdC**

What error message did you get?

**Answer: lvextend: Not enough free physical extents available.**

**Logical volume: /dev/vg00/lvmirror could not be extended.**

This is because by default the allocation policy for a logical volume is "strict".

i.e. You cannot mirror to the same physical volume.

Mirroring to the same physical volume is only possible with the allocation policy "non-strict".

6. Mirror "lvmirror" to disk1 (/dev/dsk/**cXtYdZ**).

**Command: # lvextend -m 1 /dev/vg00/lvmirror /dev/dsk/  
cXtYdZ**

Extend "lvmirror" to use the remaining Physical extents on your root disk.

**Command: # lvextend -L 100 /dev/vg00/lvmirror  
# vgcfgbackup vg00**

How long did that operation take?

**Time: Should be quicker than it was in Task 2**

---

**◆ Module 10 – Mirroring a Logical Volume**


---

7. Make a file system on "lvmirror", mount it under */mirror* and copy some files to it.  
 Perform an on-line backup of mirror (Split the mirror, mount the off-line half to */backup*, run your favorite backup program, merge the mirror). Write down the commands you need:

**Commands:** # **newfs /dev/vg00/rlvmirror**  
 # **mkdir /mirror**  
 # **vi /etc/fstab** and add an entry for your  
                   logical volume  
 # **mount /dev/vg00/lvmirror /mirror**  
                   Or  
 # **mount -a**

copy some files to mirror

**e.g.** # **lvsplit /dev/vg00/lvmirror**  
 # **fsck /dev/vg00/lvmirrorb**  
 # **mkdir /backup**  
 # **mount /dev/vg00/lvmirrorb /backup**  
 # **tar cvf /dev/rmt/0m /backup**  
 # **umount /backup**  
 # **lvmerge /dev/vg00/lvmirrorb**

8. Run *lvdisplay -v* on "lvmirror".

**Command:** # **lvdisplay -v /dev/vg00/lvmirror**

Split "lvmirror".

**Command:** # **lvsplit /dev/vg00/lvmirror**

Run *lvdisplay -v /dev/vg00/lvmirror* again.



---

**◆ Module 10 – Mirroring a Logical Volume**

---

Compare the *lvdisplay* outputs.

What has changed?

**Answer: Mirror copies = 0 ; Only one LE to PE mapping.**

9. Try to merge the mirror using the following command:

```
# lvmerge /dev/vg00/lvmirror /dev/vg00/lvmirrorb
```

What happens?

**Answer: Couldn't reallocate the logical volume: Device busy**

---

**◆ Module 11 – Reducing a Volume Group**

---

**Objective:**

To reduce an existing Volume Group to one physical volume.

**Commands:**

*vgreduce*

**Tasks:**

1. Remove the Logical Volume "lvmirror".

**Commands:** # `umount /dev/vg00/lvmirror`  
# `lvremove /dev/vg00/*mirror*`

Don't forget to remove it from */etc/fstab* as well.

2. Use the following commands to make lvol2, both the Primary Swap and Dump logical volume and to remove the dump logical volume.

**Commands:** # `lvrmbboot -d dump /dev/vg00`  
# `lvlnboot -d /dev/vg00/lvol2`  
# `lvremove /dev/vg00/dump`

3. Reduce disk1 from the root Volume Group (vg00).

**Command:** # `vgreduce /dev/vg00 /dev/dsk/cXtYdZ`

---

## ◆ Module 12 – Mirroring the Boot Disk

---

**Objective:**

To mirror the Boot disk.

**Commands:**

```
pvcreate -B
mkboot
lvextend
lvlnboot
```

**Tasks:**

1. Make disk1 a boot disk using the following commands:

Commands:

```
# pvcreate -f -B /dev/rdisk/cXtYdZ (make substitutions for X Y, & Z)
# mkboot /dev/rdisk/cXtYdZ
# mkboot -a "hpux (;0)/stand/vmunix" /dev/rdisk/cXTYdZ
```

2. Extend the root Volume Group to disk1.

**Command:** *# vgextend /dev/vg00 /dev/dsk/cXtYdZ*

3. Mirror the Root and Swap Logical Volumes to disk1 using the following commands:

Commands:

```
# lvextend -m 1 /dev/vg00/lvol1 /dev/dsk/cXtYdZ
# lvextend -m 1 /dev/vg00/lvol2 /dev/dsk/cXtYdZ
```

4. Verify the changes with the *lvlnboot -v* command.

**Command:** *# lvlnboot -r /dev/vg00/lvol1*  
*# lvlnboot -s /dev/vg00/lvol2*  
*# lvlnboot -v*

---

**◆ Module 13 – Exporting/Importing a Volume Group**

---

**Objective:**

To get familiar with the commands *vgexport* and *vgimport* and to understand how Instance number for disks are configured in the file */etc/lvmtab*.

**Commands:**

```
vgexport
vgimport
vgchange
```

**Tasks:**

1. Type *strings /etc/lvmtab* to look at the contents of this configuration file.

What is the usage of */etc/lvmtab*?

**Answer: To keep a record of what Physical Volumes belong to which Volume Group**

Why do you have to use the command *strings* to list its contents?

**Answer: It is data file (use file */etc/lvmtab*)**

2. The goal of this lab is to export a Volume Group (remove the Volume Group structures from the system but not from its Physical Volume) and then to import its Physical Volume(s) again to a new Volume Group.

This is very often used to move a Volume Group from one system to another but can also be used to fix LVM problems.

To be able to export a Volume Group it must be made inactive using the *vgchange* command.

To deactivate a Volume Group all its filesystems must be unmounted and any swap Logical Volumes deactivated. Otherwise you will get the message:

*vgchange: Couldn't deactivate volume group /dev/vg01: Device busy*

---

**◆ Module 13 – Exporting/Importing a Volume Group**

---

*umount* the two Logical Volumes (*lv\_contig* and *lv\_noncontig*), then deactivate Volume Group *vg01*. Write down the major steps required:

**Commands:**        # *umount -a*  
                      # *vgchange -a n vg01*

3. What error message do you get if you try to mount a filesystem of a deactivated Volume Group?

Try *mount -a* just to see the error message:

**Error message: No such device or address**

4. Export Volume Group *vg01* from your system. Why do you need to use option *-m*? Write down all the major steps required:

**Commands:**        # *vgexport -m /tmp/vg01.map -v vg01*

**The *-m* option creates a file that contains a list of the names of the Logical Volumes in the Volume Group being exported.**

5. Have a look at */etc/lvmtab*. How is it different compared to Task 1?

**Answer: There is no longer a record of *vg01***

## ◆ Module 13 – Exporting/Importing a Volume Group

6. *vgexport* in Task 2 removed the volume group *vg01* from your system but didn't modify its physical volume information (on disk). Create a new volume group called *vgnew* and import the previously exported physical volume to it. Write down the major steps required, shutdown and reboot your system at the end of this lab to verify your success.

```
Commands:      # mkdir /dev/vgnew
                # mknod /dev/vgnew/group c 64 0x010000
                #vgimport -m /tmp/filename -v vgnew
                /dev/dsk/cXtYdZ
                # vgchange -a y vgnew
```

**REMEMBER:** *vi /etc/fstab* and change the *vg01* entries to *vgnew*. If you change your mount point name in *fstab* then don't forget to make new directories under root for the new mount points.

```
Commands:      # mount -a
                # vgcfgbackup vgnew
```

What would happen if you did not specify option *-m*?

**Answer:** The Logical Volumes would be imported using default names (*lv01*, *rlv01*, *lv02*, *rlv02* etc...)

**WARNING :** When you perform the *vgimport* command **BE AWARE OF** making a backup using the *Vgcfgbackup* after activating the Volume Group.

---

**◆ Module 14 - Rebuilding Volume Group Information**

---

**Objective:**

To recover a Volume Group's structures (PVRA & VGRA) using the *vgcfgrestore* command.

**Commands:**

*vgcfgbackup*  
*vgcfgrestore*  
*vgchange*

**Tasks:**

1. Backup Volume Group *vgnew*'s structures (PVRA & VGRA).

**Command:**       # *vgcfgbackup vgnew*

2. Deactivate Volume Group *vgnew*.

**Commands:**       # *umount -a*  
                  # *vgchange -a n vgnew*

3. Restore Volume Group *vgnew*.

**Command:**       # *vgcfgrestore -n vgnew /dev/rdisk/cXtYdZ*

4. Reactivate Volume Group *vgnew* and remount the filesystems

**Command:**       # *vgchange -a y vgnew*  
                  # *mount -a*

**In case of a recovery it is recommended to make a backup using *vgcfgbackup* .**

---

**◆ Module 15 – Maintenance Mode**


---

**Objective:**

To learn how to troubleshoot BDRA and LABEL file problems.

**Commands:**

*lvrmboot*

*lvlnboot*

*lifrm*

**Tasks:**

1. Execute the following commands to verify that the BDRA and LABEL file are correct for your root disk. Substitute for A, B, and C as appropriate for root disk.

*# lvlnboot -v* (to list the BDRA)

*# lifls -l /dev/rdisk/cAtBdC* (to list the LIF boot volume)

2. Remove the LABEL file which is part of the LIF boot volume by executing the following command:

*# lifrm /dev/rdisk/cAtBdC:LABEL*

*# lifls -l /dev/rdisk/cAtBdZ* (To verify)

3. Shutdown (-r option) and reboot your system.

What error message do you get?

**Answer: Exec failed: Cannot find /stand/vmunix or /vmunix**

4. To fix the problem execute the following command to boot in maintenance mode:

*ISL> hpux -lm (;0)/stand/vmunix*

The system boots to single user mode without activating LVM.



---

◆ **Module 15 – Maintenance Mode**

---

5. Activate the root Volume Group.

**Command: # vgchange -a y vg00**

6. Create a new LABEL file and update it by copying the root and swap device information contained in the BDRA to it:

**Command: # lvinboot -R**

7. Mount /usr to have the *lifs* command available.

Was the LABEL file created by *lvinboot -R* ?  **YES**  **NO**

The problem should be fixed but you are still in maintenance mode.  
Should you boot from maintenance mode into multiuser mode  
using init 2 ?  **YES**  **NO**

---

**◆ Module 16 – Creating */etc/lvmtab***

---

**Objective:**

To create the */etc/lvmtab* file.

**Commands:**

*vgscan*

**Tasks:**

1. The */etc/lvmtab* file is created by executing the command *vgscan*.

---

**NOTE:** *vgscan* will NOT recreate */etc/lvmtab* if it already exists. Therefore you must rename (*mv*) or remove (*rm*) */etc/lvmtab* before executing *vgscan*. BUT for recovery we better recommend to “mv”

---

Type the following to see the current contents of */etc/lvmtab*:

```
# strings /etc/lvmtab
```

2. Rename */etc/lvmtab* to */etc/lvmtab.bk*

**Command:** # *mv /etc/lvmtab /etc/lvmtab.bk*

3. Type the following to create and rebuild */etc/lvmtab*:

```
# vgscan -v
```

---

**NOTE:** You will get the following message as part of the result of the *vgscan* command

---

```
Vgscan: Couldn't access the list of physical volumes for volume
group “/dev/vg00”
Physical Volume “/dev/dsk/cUtVdW” is not part of a volume group
/dev/vg00
/dev/dsk/cAtBdC
```

4. Examine the contents of */etc/lvmtab* with the *strings* command.

Is the result the same as that of Step 1? **Yes**

---

**◆ Module 17 - Rebuilding the LVM Root Disk's Logical Interchange Format (LIF) Utilities**

---

**Objective:**

To remove and recreate an LVM boot disks LIF area.

**Commands:**

```
rmboot  
mkboot
```

**Tasks:**

1. Briefly review the on-line manual entry for *mkboot*, then answer the following questions:

What *mkboot* option should be used to create or update the AUTO file

**Answer: # mkboot -a**

2. Execute the following command to display the current contents of the LIF area.

```
# lifls -l /dev/rdisk/cAtBdC           (for root disk)
```

3. Execute the following command to see the current contents of LIF's "AUTO" file.

```
# lifcp /dev/rdisk/cAtBdC:AUTO -
```

What does it contain?

**Answer: hpux**

---

**◆ Module 17 - Rebuilding the LVM Root Disk's Logical Interchange Format (LIF) Utilities**

---

4. Using shutdown, go to single-user mode, then mount all file systems.

**Commands:**       # **shutdown 0**  
                  # **mount -a**

5. Remove the LIF area with the "*rmboot*" command and verify your success by executing the *lifs* command used in Task 2.

**Command:**       # **rmboot /dev/rdisk/cAtBdC**

Try executing the *lifs -l /dev/rdisk/cAtBdC* command you will get the message:

*lifs: Can't list /dev/rdisk/cAtBdC; not a LIF volume.*

6. Use *mkboot* to recreate the Root Disk's LIF area.

**Command:**       # **mkboot /dev/rdisk/cAtBdC**

7. Update LIF's AUTO file with the correct boot string for your system.

**Command:**       # **mkboot -a "hpux" /dev/rdisk/cAtBdC**

8. Update the LIF area with *lvlnboot -R*.

**Command:**       # **lvlnboot -R**

Why is this necessary?

**Answer: To create the LABEL file**

9. Reboot to verify your success.

---

◆ **Module 17 - Rebuilding the LVM Root Disk's Logical Interchange Format (LIF) Utilities**

---

**WARNING : this will be your last lab. This is an interesting lab but it is possible the system goes in Panic .No patche is available to resolve that issue for the moment.**

---

- 10** Remove the root and primary swap logical volume information from the BDRA by executing the following command:

```
# lvrmbboot -r /dev/vg00
```

---

**NOTE:** This also clears the information contained in the LABEL file but does not delete the file.

---

- 11** Display the BDRA information:

**Command:** `# lvinboot -v`

What message do you get?

**Answer: Boot definitions for volume group /dev/vg00:  
The Boot Data Area is empty**

Display the LIF boot volume information:

**Command:** `# lifls -l /dev/rdisk/cAtBdC`

---

**NOTE:** The LABEL file should be there. But it does not contain any information because we removed the info at the same time as we removed the info from the BDRA.

---

- 10.** Shutdown and reboot the system.

What error message do you get?

**Answer: Exec failed: Cannot find /stand/vmunix or /vmunix**

---

**◆ Module 17 - Rebuilding the LVM Root Disk's Logical Interchange Format (LIF) Utilities**

---

11. Boot in maintenance mode.

**Command:** ISL> hpux -lm (;0) /stand/vmunix

12. Activate the root Volume Group.

**Command:** # vgchange -a y vg00

13. Update the BDRA and LABEL file with *lvlnboot* so that it contains the correct root and primary swap information:

**Command:** # lvlnboot -r /dev/vg00/lvol3  
# lvlnboot -s /dev/vg00/lvol2  
# lvlnboot -d /dev/vg00/lvol2

14. Reboot to verify.