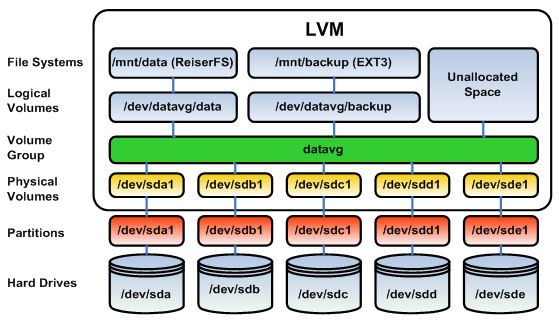
**LVM**

**LVM = Logical Volume Manager**

The LVM solution is widely used for managing disk storage. LVM enables you to accumulate spaces taken from one or several disks (called **physical volumes**) to form a large logical container (called, **volume group**), which can then be divided into partitions (called **logical volumes**). The following figure demonstrates LVM components, and you should notice the following:

* + the layered architecture
  + 5 hard drives each with 1 partition
  + Each of the 5 partitions that were created on the five different hard drives is declared as a physical volume (we’ll se how)
  + Then all these declared physical volumes are grouped together into a volume group which cumulates the whole capacity
  + On this volume group, logical volumes of desired size are created
  + On the logical volumes, file systems are created



LVM structure is made up of three key virtual objects called physical volume, volume group and logical volume.

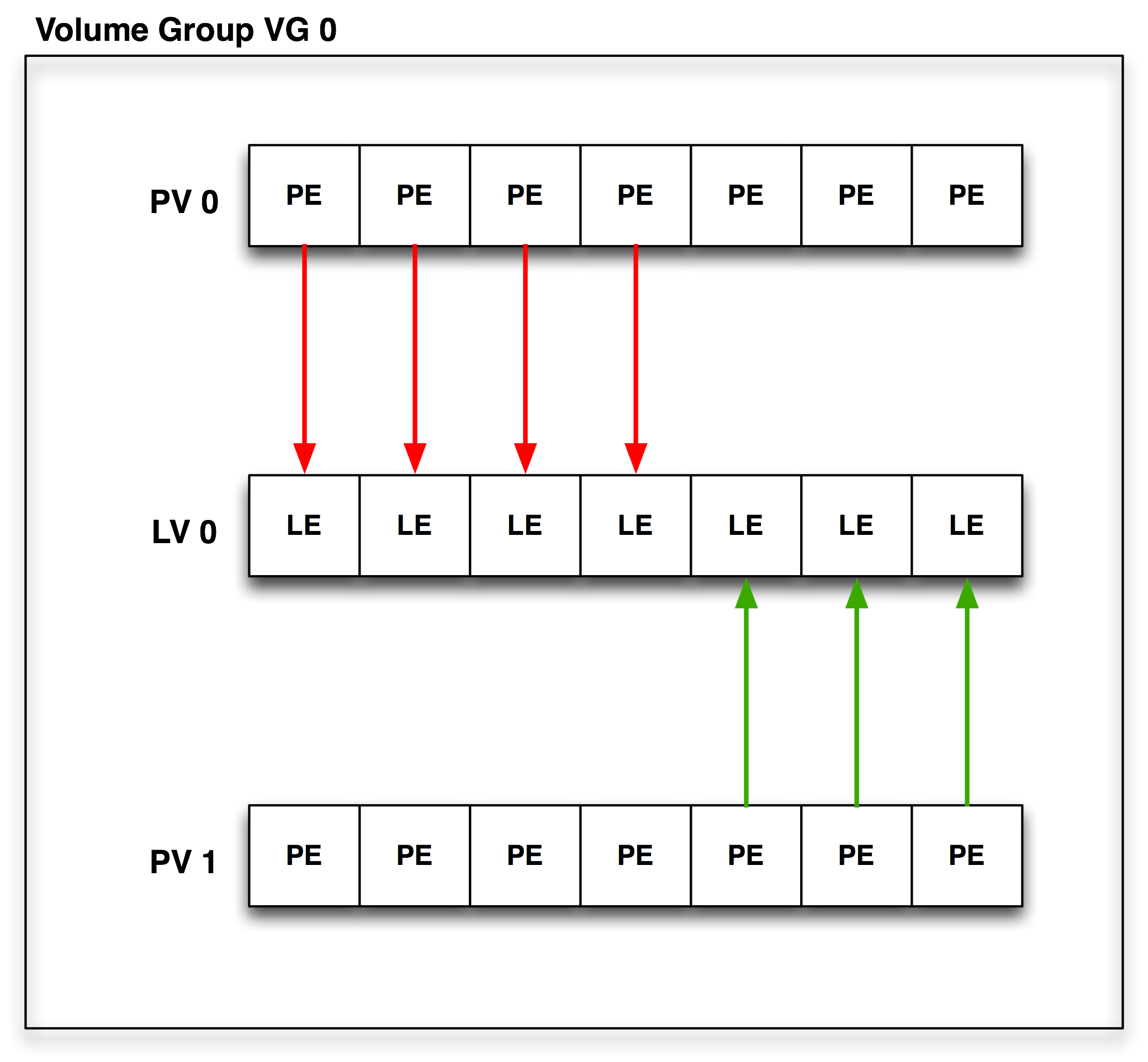
A physical volume (PV) is created when a standard partition, a software RAID partition or device, an entire disk are brought together under the LVM control by initializing the partition or disk, and constructing LVM data structures on it. A partition or disk must first be initialized (declared) as a physical volume before it can be used in a volume group.

A volume group (VG) is created when at least one physical volume is added to it. The spaces from all physical volumes is a volume group is summed up to form a large pool of storage, which is then used to build one or more logical volumes. The default naming convention for volume groups is VolGroup00, VolGroup01… however you may name them as you wish.

A logical volume (LV) is a partition of a volume group. A LV can be increased or decreased in size and can use space taken from several physical volumes inside the volume group. The default naming convention for logical volumes is lvol1, lvol2… however you may name them as you wish.

When a volume group is created, the physical volume added to it is divided into several smaller logical pieces calles PHYSICAL EXTENTS (PE). An extent is the smallest allocatable unit of space in LVM. At volume group creation time, you can either define the size of the PE or leave it to the default value which is 4MB. This means that and 8GB disk would contain approximately 2000PEs. All physical volumes in a volume group must use the same PE size.

A logical volume is made up of extents called LOGICAL EXTENTS (LE). LOGICAL EXTENTS point to PHYSICAL EXTENTS, just like in the following picture:



The PE and LE sizes are usually kept the same within a volume group. A logical extent, can be smaller or larger that a physical extent. The default size of the LE is just as for PE, 4MB. Notice in the above picture that LV 0 has LEs mapped to some PEs of PV0 but also some LEs mapped to PEs of PV1.

Managing disk space using LVM involves several tasks such as creating a physical volume, creating and displaying a volume group, creating and displaying a logical volume, extending a volume group, extending a logical volume, reducing a logical volume, renaming a logical volume, removing a logical volume, moving physical extents to another physical volume, reducing a volume group, removing a volume group and uninitializing a physical volume. In the following table you can find the available LVM commands:

|  |  |
| --- | --- |
| **Command** | **Description** |
| Lvcreate | Creates a logical volume |
| Lvdisplay | Displays details of a logical volume |
| Lvextend | Extends a logical volume |
| Lvreduce | Reduces a logical volume |
| Lvremove | Remove a logical volume |
| Lvrename | Renames a logical volume |
| Pvcreate | Initializes a disk or partition for LVM |
| Pvdisplay | Displays details of a physical volume |
| Pvmove | Moves data from one PV to another |
| Pvremove | Uninitializes a physical volume |
| Vgcreate | Creates a volume group |
| Vgdisplay | Displays details of a volume group |
| Vgextend | Extends a volume group |
| Vgremove | Removes a volume group |

In the following section we will create a volume group with 3 logical volumes on which ext3 filesystems will be created and then mounted.

**[root@Grendel ~]# fdisk -l**

Disk /dev/sda: 12.8 GB, 12884901888 bytes

255 heads, 63 sectors/track, 1566 cylinders

Units = cylinders of 16065 \* 512 = 8225280 bytes

Device Boot Start End Blocks Id System

/dev/sda1 \* 1 1239 9952236 83 Linux

/dev/sda2 1240 1272 265072+ 82 Linux swap / Solaris

/dev/sda3 1273 1275 24097+ 8e Linux LVM

/dev/sda4 1276 1566 2337457+ 5 Extended

/dev/sda5 1276 1288 104391 fd Linux raid autodetect

/dev/sda6 1289 1301 104391 fd Linux raid autodetect

**/dev/sda7 1302 1314 104391 8e Linux LVM**

**/dev/sda8 1315 1327 104391 8e Linux LVM**

Disk /dev/md0: 106 MB, 106823680 bytes

2 heads, 4 sectors/track, 26080 cylinders

Units = cylinders of 8 \* 512 = 4096 bytes

Disk /dev/md0 doesn't contain a valid partition table

**[root@Grendel ~]# pvcreate /dev/sda7 -> initialize /dev/sda7 partition as LVM physical volume**

Physical volume "/dev/sda7" successfully created

**[root@Grendel ~]# pvcreate /dev/sda8**

Physical volume "/dev/sda8" successfully created

[root@Grendel ~]#

**[root@Grendel ~]# vgcreate vg\_application /dev/sda7 /dev/sda8 -> create volume group named “vg\_application” which contains those 2 PVs**

Volume group "vg\_application" successfully created

**[root@Grendel ~]# pvs -> PV status**

PV VG Fmt Attr PSize PFree

/dev/sda7 vg\_application lvm2 a- 100.00M 100.00M

/dev/sda8 vg\_application lvm2 a- 100.00M 100.00M

**[root@Grendel ~]# vgs -> VG status**

VG #PV #LV #SN Attr VSize VFree

vg\_application 2 0 0 wz--n- 200.00M 200.00M

**[root@Grendel ~]# lvcreate -n lv\_oracle -L 50M vg\_application -> create a logical volume with name after –n and size after –L (size in MB/GB)or –l (size in LEs)**

Rounding up size to full physical extent 52.00 MB

Logical volume "lv\_oracle" created

**[root@Grendel ~]# lvs -> LV status**

LV VG Attr LSize Origin Snap% Move Log Copy% Convert

**lv\_oracle vg\_application -wi-a- 52.00M**

**[root@Grendel ~]# vgdisplay vg\_application | grep PE**

**PE Size 4.00 MB**

Total PE 50

Alloc PE / Size 13 / 52.00 MB

Free PE / Size 37 / 148.00 MB

[root@Grendel ~]#

**[root@Grendel ~]# lvcreate -n lv\_mysql -l 10 vg\_application**

Logical volume "lv\_mysql" created

**[root@Grendel ~]# lvs**

LV VG Attr LSize Origin Snap% Move Log Copy% Convert

lv\_mysql vg\_application -wi-a- 40.00M

lv\_oracle vg\_application -wi-a- 52.00M

[root@Grendel ~]#

**[root@Grendel ~]# pvdisplay /dev/sda7 -> display information about the physical volume /dev/sda7**

--- Physical volume ---

PV Name /dev/sda7

VG Name vg\_application

**PV Size 101.94 MB / not usable 1.94 MB**

Allocatable yes

**PE Size (KByte) 4096**

Total PE 25

Free PE 12

Allocated PE 13

PV UUID 30QikC-mMRQ-XLGd-1PFq-6o67-1ZBd-loHajV

[root@Grendel ~]#

[root@Grendel ~]#

**[root@Grendel ~]# vgdisplay vg\_application -> display information about volume group vg\_application**

--- Volume group ---

VG Name vg\_application

System ID

Format lvm2

Metadata Areas 2

Metadata Sequence No 3

VG Access read/write

VG Status resizable

MAX LV 0

Cur LV 2

Open LV 0

Max PV 0

Cur PV 2

Act PV 2

VG Size 200.00 MB

**PE Size 4.00 MB**

Total PE 50

Alloc PE / Size 23 / 92.00 MB

Free PE / Size 27 / 108.00 MB

VG UUID jdxomu-rEnb-raI4-WhLb-nW0V-FhLr-f4jYfd

**[root@Grendel ~]# vgdisplay -v vg\_application -> display more detailed info (-v)**

Using volume group(s) on command line

Finding volume group "vg\_application"

--- Volume group ---

VG Name vg\_application

System ID

Format lvm2

Metadata Areas 2

Metadata Sequence No 3

VG Access read/write

VG Status resizable

MAX LV 0

Cur LV 2

Open LV 0

Max PV 0

Cur PV 2

Act PV 2

VG Size 200.00 MB

PE Size 4.00 MB

Total PE 50

Alloc PE / Size 23 / 92.00 MB

Free PE / Size 27 / 108.00 MB

VG UUID jdxomu-rEnb-raI4-WhLb-nW0V-FhLr-f4jYfd

**--- Logical volume ---**

**LV Name /dev/vg\_application/lv\_oracle**

**VG Name vg\_application**

**LV UUID jZvTui-OdTG-Cj9g-PDMT-r1DQ-42M3-i3oOrQ**

**LV Write Access read/write**

**LV Status available**

**# open 0**

**LV Size 52.00 MB**

**Current LE 13**

**Segments 1**

**Allocation inherit**

**Read ahead sectors auto**

**- currently set to 256**

**Block device 253:0**

**--- Logical volume ---**

**LV Name /dev/vg\_application/lv\_mysql**

**VG Name vg\_application**

**LV UUID CpsFtw-n8Q1-IBnu-ZyD3-8sr0-McIE-lKoH2Z**

**LV Write Access read/write**

**LV Status available**

**# open 0**

**LV Size 40.00 MB**

**Current LE 10**

**Segments 1**

**Allocation inherit**

**Read ahead sectors auto**

**- currently set to 256**

**Block device 253:1**

**--- Physical volumes ---**

**PV Name /dev/sda7**

**PV UUID 30QikC-mMRQ-XLGd-1PFq-6o67-1ZBd-loHajV**

**PV Status allocatable**

**Total PE / Free PE 25 / 12**

**PV Name /dev/sda8**

**PV UUID MJaOfC-MK33-KX13-LUiH-0ORX-2jPw-okkaDy**

**PV Status allocatable**

**Total PE / Free PE 25 / 15**

[root@Grendel ~]#

[root@Grendel ~]#

**[root@Grendel ~]# lvdisplay /dev/vg\_application/lv\_oracle -> display info about the LV**

--- Logical volume ---

LV Name /dev/vg\_application/lv\_oracle

VG Name vg\_application

LV UUID jZvTui-OdTG-Cj9g-PDMT-r1DQ-42M3-i3oOrQ

LV Write Access read/write

LV Status available

# open 0

LV Size 52.00 MB

Current LE 13

Segments 1

Allocation inherit

Read ahead sectors auto

- currently set to 256

Block device 253:0

**[root@Grendel ~]# lvdisplay -v /dev/vg\_application/lv\_oracle**

Using logical volume(s) on command line

--- Logical volume ---

LV Name /dev/vg\_application/lv\_oracle

VG Name vg\_application

LV UUID jZvTui-OdTG-Cj9g-PDMT-r1DQ-42M3-i3oOrQ

LV Write Access read/write

LV Status available

# open 0

LV Size 52.00 MB

Current LE 13

Segments 1

Allocation inherit

Read ahead sectors auto

- currently set to 256

Block device 253:0

[root@Grendel ~]# lvs

LV VG Attr LSize Origin Snap% Move Log Copy% Convert

lv\_mysql vg\_application -wi-a- 40.00M

lv\_oracle vg\_application -wi-a- 52.00M

**[root@Grendel ~]# lvcreate -n lv\_apache -l 16 vg\_application**

Logical volume "lv\_apache" created

**[root@Grendel ~]# lvs**

LV VG Attr LSize Origin Snap% Move Log Copy% Convert

lv\_apache vg\_application -wi-a- 64.00M

lv\_mysql vg\_application -wi-a- 40.00M

lv\_oracle vg\_application -wi-a- 52.00M

**[root@Grendel ~]# mkfs -t ext3 /dev/vg\_application/lv\_oracle**

mke2fs 1.39 (29-May-2006)

Filesystem label=

OS type: Linux

Block size=1024 (log=0)

Fragment size=1024 (log=0)

13328 inodes, 53248 blocks

2662 blocks (5.00%) reserved for the super user

First data block=1

Maximum filesystem blocks=54525952

7 block groups

8192 blocks per group, 8192 fragments per group

1904 inodes per group

Superblock backups stored on blocks:

8193, 24577, 40961

Writing inode tables: done

Creating journal (4096 blocks): done

Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 33 mounts or

180 days, whichever comes first. Use tune2fs -c or -i to override.

**[root@Grendel ~]# mkfs -t ext3 /dev/vg\_application/lv\_mysql**

mke2fs 1.39 (29-May-2006)

Filesystem label=

OS type: Linux

Block size=1024 (log=0)

Fragment size=1024 (log=0)

10240 inodes, 40960 blocks

2048 blocks (5.00%) reserved for the super user

First data block=1

Maximum filesystem blocks=41943040

5 block groups

8192 blocks per group, 8192 fragments per group

2048 inodes per group

Superblock backups stored on blocks:

8193, 24577

Writing inode tables: done

Creating journal (4096 blocks): done

Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 25 mounts or

180 days, whichever comes first. Use tune2fs -c or -i to override.

**[root@Grendel ~]# mkfs -t ext3 /dev/vg\_application/lv\_apache**

mke2fs 1.39 (29-May-2006)

Filesystem label=

OS type: Linux

Block size=1024 (log=0)

Fragment size=1024 (log=0)

16384 inodes, 65536 blocks

3276 blocks (5.00%) reserved for the super user

First data block=1

Maximum filesystem blocks=67108864

8 block groups

8192 blocks per group, 8192 fragments per group

2048 inodes per group

Superblock backups stored on blocks:

8193, 24577, 40961, 57345

Writing inode tables: done

Creating journal (4096 blocks): done

Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 20 mounts or

180 days, whichever comes first. Use tune2fs -c or -i to override.

**[root@Grendel ~]# mkdir /oracle**

**[root@Grendel ~]# mkdir /mysql**

**[root@Grendel ~]# mkdir /apache**

**[root@Grendel ~]# mount -t ext3 /dev/vg\_application/lv\_oracle /oracle/**

**[root@Grendel ~]# mount -t ext3 /dev/vg\_application/lv\_mysql /mysql/**

**[root@Grendel ~]# mount -t ext3 /dev/vg\_application/lv\_apache /apache/**

**[root@Grendel ~]# df -h**

Filesystem Size Used Avail Use% Mounted on

/dev/sda1 9.2G 2.0G 6.8G 23% /

tmpfs 252M 0 252M 0% /dev/shm

/dev/md0 99M 5.6M 89M 6% /mnt/Music

**/dev/mapper/vg\_application-lv\_oracle**

**51M 4.9M 43M 11% /oracle**

**/dev/mapper/vg\_application-lv\_mysql**

**39M 4.5M 33M 13% /mysql**

**/dev/mapper/vg\_application-lv\_apache**

**62M 5.3M 54M 9% /apache**

Notice that when creating a logical volume we can specify it’s size in LEs or in MB/GB.

When we specify the size in LE we have to get the LE size (if we don’t change it at the creation of the logical volume it will be the default of 4MB) and then multiply it with the number of LEs required until you reach the desired capacity.

When we specify the size in MB/GB LVM will automatically adjust the size requested to the number of LEs needed. In our example we have requested a LV size of 50MB but LVM adjusted it to 52 MB (PE size = 4MB x 13 PEs = 52 MB).

**[root@Grendel ~]# history | grep lvcreate**

100 lvcreate -L 25 -n LV1 VG1

625 lvcreate -L 120M -n lv\_test1 vg\_local

660 lvcreate -L 100M -n lv\_test vg\_local

**1046 lvcreate -n lv\_oracle -L 50M vg\_application**

1049 lvcreate -n lv\_mysql -l 10 vg\_application

1059 lvcreate -n lv\_apache -l 16 vg\_application

1085 history | grep lvcreate

**[root@Grendel ~]# lvs**

LV VG Attr LSize Origin Snap% Move Log Copy% Convert

lv\_apache vg\_application -wi-ao 64.00M

lv\_mysql vg\_application -wi-ao 40.00M

**lv\_oracle vg\_application -wi-ao 52.00M**

1. **Exercises**

On your machine create 6 partitions: 1 for swap, 1 for Linux ext3, 2 for RAID, 2 for LVM.

Create swap filesystem and make sure it will available at reboot

Create an ext3 filesystem and mount it under /share and make sure it will available at reboot

Create a RAID0 concat device named md100, display the information about it and it’s process, create an ext3 filesystem on it and mount it under /storage.

Stop the RAID device.

Create a volume group named vg\_prod with 2 logical volumes named lv\_prod1 and lv\_prod2 of 10 and 20 MB size. Modify PE and LE size to 2MB. Create an ext3 filesystem on them and mount them under /app/prod1 and /app/prod2. Make sure that they will be available at reboot.