LOGICAL VOLUME MANAGEMENT – LVM

ORACLE®

LINUX

redhat

CentOS

chanaka.lasantha@gmail.com

What is LVM?

LVM is a logical volume manager for the Linux kernel that manages disk drives and similar mass-storage devices. Heinz Mauelshagen wrote the original code in 1998, taking its primary design guidelines from the HP-UX's volume manager.

The installers for the CrunchBang, CentOS, Debian, Fedora, Gentoo, Mandriva, MontaVista Linux, openSUSE, Pardus, Red Hat Enterprise Linux, Slackware, SLED, SLES, Linux Mint, Kali Linux, and Ubuntu distributions are LVM-aware and can install a bootable system with a root filesystem on a logical volume.

Common Uses

LVM is commonly used for the following purposes:

- Managing large hard disk farms by allowing disks to be added and replaced without downtime or service disruption, in combination with hot swapping.
- On small systems (like a desktop at home), instead of having to estimate at installation time how big a partition might need to be in the future, LVM allows file systems to be easily resized later as needed.
- Performing consistent backups by taking snapshots of the logical volumes.
- Creating single logical volumes of multiple physical volumes or entire hard disks (somewhat similar to RAID 0, but more similar to JBOD), allowing for dynamic volume resizing.
- the Ganeti solution stack relies on the Linux Logical Volume Manager

LVM can be considered as a thin software layer on top of the hard disks and partitions, which creates an abstraction of continuity and ease-of-use for managing hard drive replacement, re-partitioning, and backup.

Features

The LVM can:

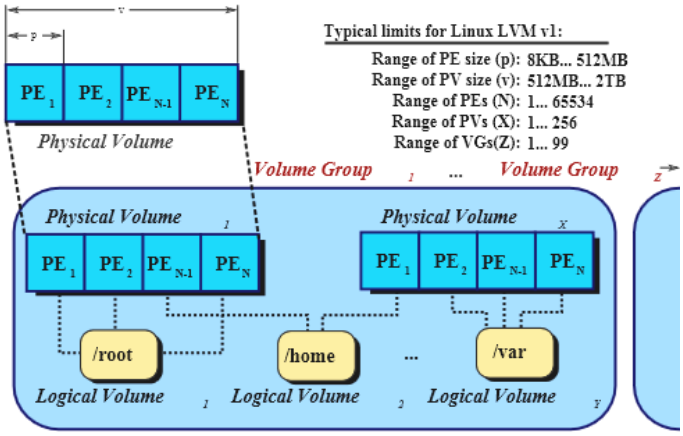
- Resize volume groups online by absorbing new physical volumes (PV) or ejecting existing ones.
- Resize logical volumes (LV) online by concatenating extents onto them or truncating extents from them.
- Create read-only snapshots of logical volumes (LVM1).
- Create read-write snapshots of logical volumes (LVM2).
- Create RAID logical volumes (available in newer LVM implementations): RAID 1, RAID 5, RAID 6, etc.
- Stripe whole or parts of logical volumes across multiple PVs, in a fashion similar to RAID 0.
- Configure a RAID 1 backend device (a PV) as *write-mostly*, resulting in reads being avoided to such devices

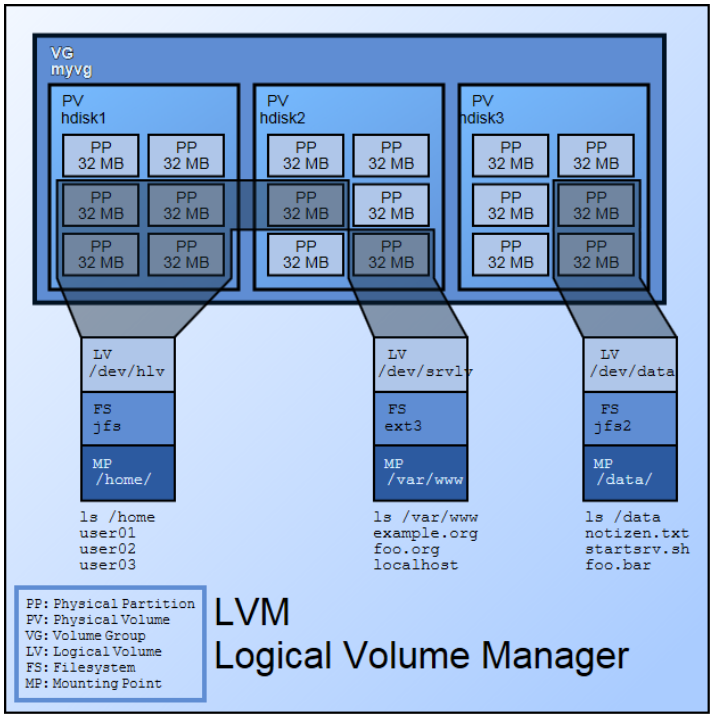
unless necessary.

- Allocate thin-provisioned logical volumes from a pool.
- Move online logical volumes between PVs.
- Split or merge volume groups *in situ* (as long as no logical volumes span the split). This can be useful when migrating whole logical volumes to or from offline storage.
- Create hybrid volumes by using the dm-cache target, which allows one or more fast storage devices, such as flash-based solid-state drives (SSDs), to act as a cache for one or more slower hard disk drives (HDDs).

The LVM will also work in a shared-storage cluster (where disks holding the PVs are shared between multiple host computers), but requires an additional daemon to propagate state changes between cluster nodes.

Implementation





LVM keeps a metadata header at the start of every physical volume, each of which is uniquely identified by a UUID. Each PV's header is a complete copy of the entire volume group's layout, including the UUIDs of all other PVs, the UUIDs of all logical volumes and an allocation map of PEs to LEs. This simplifies data recovery in the event of PV loss.

In the 2.6-series of the Linux Kernel, the LVM is implemented in terms of the device mapper, a simple block-level scheme for creating virtual block devices and mapping their contents onto other block devices. This minimizes the amount of

relatively hard-to-debug kernel code needed to implement the LVM. It also allows its I/O redirection services to be shared with other volume managers (such as EVMS). Any LVM-specific code is pushed out into its user-space tools, which merely manipulate these mappings and reconstruct their state from on-disk metadata upon each invocation.

To bring a volume group online, the "vgchange" tool:

1. Searches for PVs in all available block devices.
2. Parses the metadata header in each PV found.
3. Computes the layouts of all visible volume groups.
4. Loops over each logical volume in the volume group to be brought online and:
    1. Checks if the logical volume to be brought online has all its PVs visible.
    2. Creates a new, empty device mapping.
    3. Maps it (with the "linear" target) onto the data areas of the PVs the logical volume belongs to.

To move an online logical volume between PVs on the same Volume Group, use the "pvmove" tool:

1. Creates a new, empty device mapping for the destination.
2. Applies the "mirror" target to the original and destination maps. The kernel will start the mirror in "degraded" mode and begin copying data from the original to the destination to bring it into sync.
3. Replaces the original mapping with the destination when the mirror comes into sync, then destroys the original.

These device mapper operations take place transparently, without applications or file systems being aware that their underlying storage is moving.

Physical Volume

Verifying the Existing Partitions

Input Command

```
df -h
```

Output

```
Filesystem                Size  Used  Avail  Use%  Mounted on
/dev/mapper/VolGroup-lv_root
                          18G   1.5G  15G    9%    /
tmpfs                     494M  0     494M   0%    /dev/shm
/dev/sda1                 477M  53M   400M   12%   /boot
```

Obtaining More Details on Physical and Logical Drives

Input Command

```
fdisk –l

or

fdisk -l /dev/sdb
```

Output

```
Disk /dev/sda: 21.5 GB, 21474836480 bytes
255 heads, 63 sectors/track, 2610 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00063c7d

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1          64      512000   83  Linux
Partition 1 does not end on cylinder boundary.
/dev/sda2              64        2611    20458496   8e  Linux LVM

Disk /dev/sdb: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/mapper/VolGroup-lv_root: 18.9 GB, 18865979392 bytes
```

```
255 heads, 63 sectors/track, 2293 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/mapper/VolGroup-lv_swap: 2080 MB, 2080374784 bytes
255 heads, 63 sectors/track, 252 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000
```

Create a Physical Volume

Input Command

```
pvcreate  -ff  /dev/sdb
```

Output

```
Physical volume "/dev/sdb" successfully created
```

Display a status of Physical volumes

Input Command

```
pvdisplay  /dev/sdb
```

Output

```
"/dev/sdb" is a new physical volume of "10.00 GiB"
  --- NEW Physical volume ---
  PV Name               /dev/sdb
  VG Name
  PV Size               10.00 GiB
  Allocatable           NO
  PE Size               0
  Total PE              0
  Free PE               0
  Allocated PE          0
  PV UUID               vCrqg6-WYXP-ecjb-g6ig-hKx3-PNrT-F9GZUF
```

Change size of a Physical volume

Input Command

```
pvresize  --setphysicalvolumesize  9G  /dev/sdb
```

Output

```
"/dev/sdb" is a new physical volume of "9.00 GiB"
  --- NEW Physical volume ---
  PV Name               /dev/sdb
  VG Name
  PV Size               9.00 GiB
  Allocatable           NO
  PE Size               0
  Total PE              0
  Free PE               0
  Allocated PE          0
  PV UUID               vCrqg6-WYXP-ecjb-g6ig-hKx3-PNrT-F9GZUF
```

Output reports of Physical volumes

Input Command

```
pvs  /dev/sdb
```

Output

```
PV         VG   Fmt   Attr  PSize  PFree
/dev/sdb        lvm2  a--   9.00g  9.00g
```

Scan Physical volumes

Input Command

```
pvscan  /dev/sdb
```

Output

```
PV /dev/sda2   VG VolGroup       lvm2 [19.51 GiB / 0    free]
PV /dev/sdb                      lvm2 [9.00 GiB]
Total: 2 [28.51 GiB] / in use: 1 [19.51 GiB] / in no VG: 1 [9.00 GiB]
```

Remove a Physical volume

Input Command

```
pvremove  /dev/sdb
```

Output

```
Labels on physical volume "/dev/sdb" successfully wiped
```

Volume Group

Create a Volume Group

Input Command

```
vgcreate volg1 /dev/sdb

or

vgcreate volg1 /dev/sdb  /dev/sdc
```

Output

```
Volume group "volg1" successfully created
```

Display Volume Groups

Input Command

```
vgdisplay
```

Output

```
--- Volume group ---
  VG Name               volg1
  System ID
  Format                lvm2
  Metadata Areas        1
  Metadata Sequence No  1
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                0
```

```
  Open LV               0
  Max PV                0
  Cur PV                1
  Act PV                1
  VG Size               10.00 GiB
  PE Size               4.00 MiB
  Total PE              2559
  Alloc PE / Size       0 / 0
  Free  PE / Size       2559 / 10.00 GiB
  VG UUID               tDMrau-HZLI-8tgG-oKTV-vAsA-8LzQ-P1MQwx
```

Rename a Volume Group

Input Command

```
vgrename volg1 vg_data
```

Output

```
Volume group "volg1" successfully renamed to "vg_data"
```

Output reports of Volume Groups

Input Command

```
vgs
```

Output

```
  VG         #PV   #LV  #SN      Attr      VSize    VFree
  VolGroup     1    2       0      wz--n-  19.51g     0
  vg_data      1    0       0      wz--n-  10.00g   10.00g
```

Scan Volume Groups

Input Command

```
vgscan
```

Output

```
Reading all physical volumes.  This may take a while...
Found volume group "vg_data" using metadata type lvm2
Found volume group "VolGroup" using metadata type lvm2
```

Extend a Volume Group

Input Command

```
vgextend vg_data /dev/sdc
```

Output

```
Volume group "vg_data" successfully extended
```

Verifying Extended Volume Group

Input Command

```
vgdisplay
```

Output

```
--- Volume group ---
  VG Name               vg_data
  System ID
  Format                lvm2
  Metadata Areas        2
  Metadata Sequence No  4
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                0
  Open LV               0
  Max PV                0
  Cur PV                2
  Act PV                2
  VG Size               14.99 GiB
  PE Size               4.00 MiB
  Total PE              3838
  Alloc PE / Size       0 / 0
  Free  PE / Size       3838 / 14.99 GiB
  VG UUID               tDMrau-HZLI-8tgG-oKTV-vAsA-8LzQ-P1MQwx
```

Reduce a Volume Group

Input Command

```
vgreduce vg_data /dev/sdc
```

Output

```
Removed "/dev/sdc" from volume group "vg_data"
```

Verifying Reduced Volume Group

Input Command

```
vgdisplay
```

Output

```
--- Volume group ---
  VG Name               vg_data
  System ID
  Format                lvm2
  Metadata Areas        1
  Metadata Sequence No  5
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                0
  Open LV               0
  Max PV                0
  Cur PV                1
  Act PV                1
  VG Size               10.00 GiB
  PE Size               4.00 MiB
  Total PE              2559
  Alloc PE / Size       0 / 0
  Free  PE / Size       2559 / 10.00 GiB
  VG UUID               tDMrau-HZLI-8tgG-oKTV-vAsA-8LzQ-P1MQwx
```

Remove a Volume Group

Input Command

```
vgchange -a n vg_data
```

NOTE: Turn non-active first

Output

```
0 logical volume(s) in volume group "vg_data" now active
```

Input Command

```
vgremove vg_data
```

Output

```
Volume group "vg_data" successfully removed
```

Logical Volume

Create a Logical Volume

Input Command

```
lvcreate -L 8M -n lv_data vg_data
```

NOTE: create a Logical Volumes 'lv_data' as 8G in volume group 'vg_data'

Output

```
Logical volume "lv_data" created
```

Display status of Logical Volumes

Input Command

```
lvdisplay
```

Output

```
--- Logical volume ---
  LV Path                /dev/vg_data/lv_data
  LV Name                lv_data
  VG Name                vg_data
  LV UUID                Qd3D2I-KZiU-DuDg-il22-JH7F-VKJn-j5SyO8
  LV Write Access        read/write
  LV Creation host, time localhost.localdomain, 2014-11-03 01:36:01 +0530
  LV Status              available
  # open                 0
  LV Size                8.00 MiB
  Current LE             2
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           252:2
```

Rename a Logical Volume

Input Command

```
lvrename vg_data lv_data lv_storage
```

Output

```
Renamed "lv_data" to "lv_storage" in volume group "vg_data"
```

Verifying Input Command

```
lvdisplay
```

Output

```
--- Logical volume ---
  LV Path                /dev/vg_data/lv_storage
  LV Name                lv_storage
  VG Name                vg_data
  LV UUID                Qd3D2I-KZiU-DuDg-il22-JH7F-VKJn-j5SyO8
```

```
  LV Write Access        read/write
  LV Creation host, time localhost.localdomain, 2014-11-03 01:36:01 +0530
  LV Status              available
  # open                 0
  LV Size                8.00 MiB
  Current LE             2
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           252:2
```

Output a report of Logical Volumes

Input Command

```
lvs
```

Output

```
LV         VG       Attr       LSize  Pool Origin Data%  Move Log Cpy%Sync Convert
  lv_root    VolGroup -wi-ao---- 17.57g
  lv_swap    VolGroup -wi-ao----  1.94g
  lv_storage vg_data  -wi-a-----  8.00m
```

Scan Logical Volumes

Input Command

```
lvscan
```

Output

```
ACTIVE            '/dev/vg_data/lv_storage' [8.00 MiB] inherit
ACTIVE            '/dev/VolGroup/lv_root' [17.57 GiB] inherit
ACTIVE            '/dev/VolGroup/lv_swap' [1.94 GiB] inherit
```

Snapshot of a Logical Volume

Input Command

```
lvcreate -s -L 50G -n snap-lv_storage /dev/vg_data/lv_storage
```

Output

```
--- Logical volume ---
  LV Name
/dev/vg_data/snap-lv_storage

  VG Name
vg_data

  LV UUID
Dy0PCV-izK5-vbC5-qoxP-3w1k-GGvw-atL2lU

  LV Write Access
read/write

  LV snapshot status
active destination for /dev/vg_data/lv_storage

  LV Status
available

  # open
0

  LV Size
50.00 GB

  Current LE
12800

  Segments
1
  Allocation
inherit

  Read ahead sectors
auto

  - currently set to
256

  Block device
253:3
```

Formatting Logical Volume Before Mount It.

Input Command

```
mkfs.ext3 /dev/vg_data/lv_storage
```

Output

```
mke2fs 1.43-WIP (20-Jun-2013)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
Stride=0 blocks, Stripe width=0 blocks
2048 inodes, 8192 blocks
409 blocks (4.99%) reserved for the super user
First data block=1
Maximum filesystem blocks=8388608
1 block group
8192 blocks per group, 8192 fragments per group
2048 inodes per group

Allocating group tables: done
Writing inode tables: done
Creating journal (1024 blocks): done
Writing superblocks and filesystem accounting information: done
```

Extend a Logical Volume - it's possible to execute with keeping mounted

Input Command

```
lvextend -L +9G /dev/vg_data/lv_storage
```

Output

```
Extending logical volume lv_storage to 9.01 GiB
 Logical volume lv_storage successfully resized
```

Verifying Extended Logical Volume

```
lvdisplay
```

Output

```
--- Logical volume ---
  LV Path                /dev/vg_data/lv_storage
  LV Name                lv_storage
  VG Name                vg_data
  LV UUID                Qd3D2I-KZiU-DuDg-il22-JH7F-VKJn-j5SyO8
  LV Write Access        read/write
  LV Creation host, time localhost.localdomain, 2014-11-03 01:36:01 +0530
  LV Status              available
  # open                 0
  LV Size                9.01 GiB
  Current LE             2306
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           252:2
```

Next Last Command

```
resize2fs /dev/vg_data/lv_storage
```

Output

```
resize2fs 1.43-WIP (02-Nov-2014)
Resizing the filesystem on /dev/vg_data/lv_storage to 9445376 (1k) blocks.
The filesystem on /dev/vg_data/lv_storage is now 9445376 blocks long.
```

Reduce a Logical Volume - This destroys a file system, so Take Back-up first.

Input Command

```
lvreduce -L 5G /dev/vg_data/lv_storage
```

Output

```
WARNING: Reducing active logical volume to 5.00 GiB
THIS MAY DESTROY YOUR DATA (filesystem etc.)
Do you really want to reduce lv_storage? [y/n]: y
  Reducing logical volume lv_storage to 5.00 GiB
  Logical volume lv_storage successfully resized
```

Mounting Logical Volume into a Specific User's Folder

Input Command

```
mount /dev/vg_data/lv_storage /home/chanaka/
```

Make it Permanent when During Boot up

```
vim /etc/fstab
/dev/vg_data/lv_storage      /home/chanaka/          ext3    defaults        0 0
```

Verifying Volume Group

Input Command

```
vgs
```

Output

```
VG                  #PV #LV #SN Attr   VSize  VFree
 VolGroup             1   2   0 wz--n-  19.51g    0
 vg_data              1   1   0 wz--n-  10.00g 5.00g
```

Remove Logical Volume - unmounts a file system first.

Input Command

1. umount /dev/vg_data/lv_storage /home/chanaka/
2. mkfs.ext3 /dev/vg_data/lv_storage
3. lvremove /dev/vg_data/lv_storage

Output When "lvremove"

```
Do you really want to remove active logical volume lv_storage? [y/n]: y
  Logical volume "lv_storage" successfully removed
```

Verifying

```
lvdisplay
```