
Hands-on Lab
Add and Remove Volumes, Partition Disks, and Work with LVM
Linux Academy
Cloud Assessments

Contents

In this lab, we use Logical Volume Manager (LVM) and `gdisk` to take two disks, partition them, and combine them into a single volume group. From here, we want to create a logical volume for use as log storage. When finished, we review how to remove logical and physical volumes, as well as volume groups.

Add Partitions

Log in to the lab using the credentials provided on the Hands-on Lab page. Switch to the root user (`sudo su -`).

Two additional devices have been added to this lab for use with LVM. To see what these devices are called, run:

```
[root@LinuxAcademy ~]# ls /dev/xvd*
```

`/dev/xvdf` and `/dev/xvdj` are the two additional disks.

Check to see if `gdisk` is installed:

```
[root@LinuxAcademy ~]# gdisk
-bash: gdisk: command not found
```

Since it is not, we need to install it before we can continue:

```
[root@LinuxAcademy ~]# yum install gdisk
```

Now we can use `gdisk` to partition and format one of our provided drives. We'll start with `/dev/xvdf`:

```
[root@LinuxAcademy ~]# gdisk /dev/xvdf
```

This drops us into the `gdisk` configuration prompt, wherein we'll be asked to specify the settings we want to use for our disk. In this instance, we want to create a new partition, set the partition number to 1, partition the entire provided device, set the file system type to LVM (8e00), and write the changes to the disk. Many of these settings are the default.

```
GPT fdisk (gdisk) version 0.8.10

Partition table scan:
  MBR: not present
  BSD: not present
  APM: not present
  GPT: not present
```

```
Creating new GPT entries.

Command (? for help): n
Partition number (1-128, default 1):
First sector (34-41943006, default = 2048) or {+-}size{KMGTP}:
Last sector (2048-41943006, default = 41943006) or {+-}
size{KMGTP}:
Current type is 'Linux filesystem'
Hex code or GUID (L to show codes, Enter = 8300): 8e00
Changed type of partition to 'Linux LVM'

Command (? for help): w

Final checks complete. About to write GPT data. THIS WILL
OVERWRITE EXISTING
PARTITIONS!!

Do you want to proceed? (Y/N): y
OK; writing new GUID partition table (GPT) to /dev/xvdf.
The operation has completed successfully.
```

We can now partition our second device. We again want to do this with `gdisk`:

```
[root@LinuxAcademy ~]# gdisk /dev/xvdj
```

As before, we want to create a new partition using the defaults for partition number, first sector, and last section. We also want to continue using the Linux file system for our file system type. Write the changes and proceed:

```
GPT fdisk (gdisk) version 0.8.10

Partition table scan:
  MBR: not present
  BSD: not present
  APM: not present
  GPT: not present

Creating new GPT entries.

Command (? for help): n
Partition number (1-128, default 1):
First sector (34-41943006, default = 2048) or {+-}size{KMGTP}:
Last sector (2048-41943006, default = 41943006) or {+-}
size{KMGTP}:
Current type is 'Linux filesystem'
Hex code or GUID (L to show codes, Enter = 8300): 8e00
Changed type of partition to 'Linux LVM'

Command (? for help): w

Final checks complete. About to write GPT data. THIS WILL
```

```
OVERWRITE EXISTING
PARTITIONS!!

Do you want to proceed? (Y/N): y
OK; writing new GUID partition table (GPT) to /dev/xvdj.
The operation has completed successfully.
```

Create the Physical Volumes, Volume Group, and Logical Volume

Prepare the partitions by initializing them as a physical volume using the `pvcreate` command:

```
[root@LinuxAcademy ~]# pvcreate /dev/xvdf1 /dev/xvdj1
  Physical volume "/dev/xvdf1" successfully created
  Physical volume "/dev/xvdj1" successfully created
```

Confirm that the volumes have been created, and review their settings:

```
[root@LinuxAcademy ~]# pvdisplay
  "/dev/xvdf1" is a new physical volume of "20.00 GiB"
  --- NEW Physical volume ---
  PV Name               /dev/xvdf1
  VG Name
  PV Size               20.00 GiB
  Allocatable           NO
  PE Size               0
  Total PE              0
  Free PE               0
  Allocated PE          0
  PV UUID               k8gf50-dtIT-29Yp-RMPd-eH6E-x2J3-LyQJnx

  "/dev/xvdj1" is a new physical volume of "20.00 GiB"
  --- NEW Physical volume ---
  PV Name               /dev/xvdj1
  VG Name
  PV Size               20.00 GiB
  Allocatable           NO
  PE Size               0
  Total PE              0
  Free PE               0
  Allocated PE          0
  PV UUID               b5hihP-7m5K-x5v5-9ICx-wn3x-QNIx-MUFmox
```

Here we can see that both 20 GB volumes, `/dev/xvdf1` and `/dev/xvdj1` are available as physical volumes for LVM to use.

We can now create our volume group, comprised of both physical volumes. We called our

volume group log_vg:

```
[root@LinuxAcademy ~]# vgcreate log_vg /dev/xvdf1 /dev/xvdj1
  Volume group "log_vg" successfully created
```

Review the volume group:

```
[root@LinuxAcademy ~]# vgdisplay
  --- Volume group ---
  VG Name               log_vg
  System ID
  Format                lvm2
  Metadata Areas        2
  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                2
  Act PV                2
  VG Size               39.99 GiB
  PE Size               4.00 MiB
  Total PE              10238
  Alloc PE / Size       0 / 0
  Free  PE / Size       10238 / 39.99 GiB
  VG UUID               x8X7D6-RIHL-Oot9-1asA-UZ1Z-X4rA-NssgU1
```

Notice that the size of our volume group is about 40 GB – a combination of the two 20 GB volumes we included in the group.

Now we can create our logical volume from this volume group. To do this, we use the `lvcreate` command:

```
[root@LinuxAcademy ~]# lvcreate -n log_lv -L 10GB log_vg
  Logical volume "log_lv" created.
```

The `-n` flag denotes that we're naming the logical volume log_lv, while the `-L` flag sets the volume size – in this case, 10 GB. `log_vg` references back to the volume group we are working from.

Review the newly-created logical volume:

```
[root@LinuxAcademy ~]# lvdisplay
  --- Logical volume ---
  LV Path                /dev/log_vg/log_lv
  LV Name                log_lv
  VG Name                log_vg
```

```
  LV UUID                WYTPo1-6Gk9-lgxh-Q8Rl-PhpW-INGI-PMmHBM
  LV Write Access        read/write
  LV Creation host, time LinuxAcademy, 2017-12-14 10:37:38 -0500
  LV Status              available
  # open                 0
  LV Size                10.00 GiB
  Current LE             2560
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:0
```

From here, we can see our logical volume's path (`/dev/log_vg/log_lv`), as well as general information about the logical volume itself, including name, size, write access, creation host, and more. Copy the LV path.

We now need to format our volume to work as an ext4 file system:

```
[root@LinuxAcademy ~]# mkfs -t ext4 /dev/log_vg/log_lv
```

Create a mount point for the file system:

```
[root@LinuxAcademy ~]# mkdir /mnt/log_files
```

Finally, mount the file system:

```
[root@LinuxAcademy ~]# mount /dev/log_vg/log_lv /mnt/log_files/
```

Confirm that the file system mounted successfully:

```
[root@LinuxAcademy ~]# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/xvde1            5.8G  1.5G  4.1G  26% /
tmpfs                 270M     0  270M   0% /dev/shm
/dev/mapper/log_vg-log_lv
                      9.9G  151M  9.2G   2% /mnt/log_files
```

Remove the Logical Volume

Now that we know how to create and mount logical volumes, we also need to review how to remove them. To do this, we can remove the volume group:

```
[root@LinuxAcademy ~]# vgremove log_vg
Do you really want to remove volume group "log_vg" containing 1
```

```
logical volumes? [y/n]: y
  Logical volume log_vg/log_lv contains a filesystem in use.
```

However, this fails, outputting the error `Logical volume log_vg/log_lv contains a filesystem in use.`. This is because our file system is still mounted. Unmount the file system:

```
[root@LinuxAcademy ~]# umount /mnt/log_files/
```

We also need to remove the logical volume:

```
[root@LinuxAcademy ~]# lvremove /dev/log_vg/log_lv
Do you really want to remove active logical volume log_lv? [y/n]:
y
  Logical volume "log_lv" successfully removed
```

Finally, we can rerun the `vgremove` command to remove the volume group:

```
[root@LinuxAcademy ~]# vgremove log_vg
  Volume group "log_vg" successfully removed
```

Confirm the removal of the volume with the `vgdisplay` command:

```
[root@LinuxAcademy ~]# vgdisplay
```

Although we've removed the volume group, we do still have the two physical volumes left over. We can remove these, too, using the `pvremove` command:

```
[root@LinuxAcademy ~]# pvremove /dev/xvdf1 /dev/xvdj1
  Labels on physical volume "/dev/xvdf1" successfully wiped
  Labels on physical volume "/dev/xvdj1" successfully wiped
```

Review

We have now walked through the process of setting up physical volumes, volume groups, and logical volumes using a series of Logical Volume Manager commands, as well as how to remove these volumes and groups when finished with them. With LVM, we can merge physical disks into single volumes, easily resize our devices, and —if needed— manage large farms of disks.

This lab is now complete!