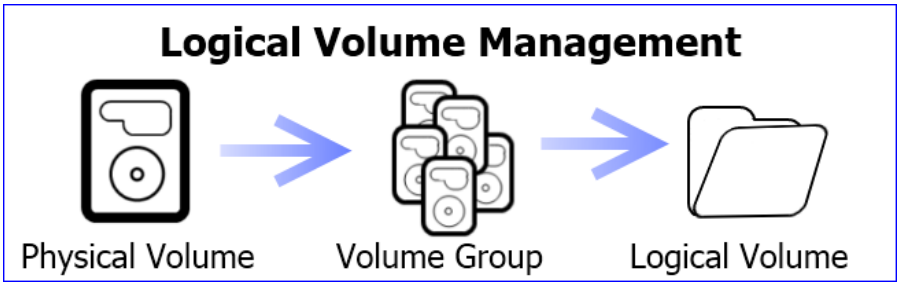
How to Manage and Use LVM (Logical Volume Management) in Ubuntu



In our previous article we told you what LVM is and what you may want to use it for, and today we are going to walk you through some of the key management tools of LVM so you will be confident when setting up or expanding your installation.

As stated before, LVM is a abstraction layer between your operating system and physical hard drives. What that means is your physical hard drives and partitions are no longer tied to the hard drives and partitions they reside on. Rather, the hard drives and partitions that your operating system sees can be any number of separate hard drives pooled together or in a software RAID.

To manage LVM there are GUI tools available but to really understand what is happening with your LVM configuration it is probably best to know what the command line tools are. This will be especially useful if you are managing LVM on a server or distribution that does not offer GUI tools.

Most of the commands in LVM are very similar to each other. Each valid command is preceded by one of the following:

- Physical Volume = pv
- Volume Group = vg
- Logical Volume = lv

The physical volume commands are for adding or removing hard drives in volume groups. Volume group commands are for changing what abstracted set of physical partitions are presented to your operating in logical volumes. Logical volume commands will present the volume groups as partitions so that your operating system can use the designated space.

Downloadable LVM Cheat Sheet

To help you understand what commands are available for each prefix we made a LVM cheat sheet. We will cover some of the commands in this article, but there is still a lot you can do that won't be covered here.

All commands on this list will need to be run as root because you are changing system wide settings that will affect the entire machine.

LVM	Logical Volume Management	Physical Volume	Volume Group	Logical Volume
	LVM	PV	VG	LV
s	No	Yes	Yes	Yes
display	No	Yes	Yes	Yes
create	No	Yes	Yes	Yes
rename	No	No	Yes	Yes
change	Yes	Yes	Yes	Yes
move	No	Yes	Yes	No
extend	No	No	Yes	Yes
reduce	No	No	Yes	Yes
resize	No	Yes	No	Yes
split	No	No	Yes	No
merge	No	No	Yes	No
convert	No	No	Yes	Yes
import	No	No	Yes	No
export	No	No	Yes	No
importclone	No	No	Yes	No
cfgbackup	No	No	Yes	No
cfgrestore	No	No	Yes	No
ck	No	Yes	Yes	No
scan	diskscan	Yes	Yes	Yes
mknodes	No	No	Yes	No
remove	No	Yes	Yes	Yes
dump	Yes	No	No	No
How-To Geek				

(Click on the thumbnail for a full size image)

How to View Current LVM Information

The first thing you may need to do is check how your LVM is set up. The `s` and `display` commands work with physical volumes (pv), volume groups (vg), and logical volumes (lv) so it is a good place to start when trying to figure out the current settings.

The `display` command will format the information so it's easier to understand than the `s` command. For each command you will see the name and path of the pv/vg and it should also give information about free and used space.

```
root@ubuntu:/home/rothgar# pvdisplay
  --- Physical volume ---
  PV Name               /dev/sda5
  VG Name               ubuntu
  PV Size               11.76 GiB / not usable 2.00 MiB
  Allocatable           yes
  PE Size               4.00 MiB
  Total PE              3010
  Free PE               7
  Allocated PE          3003
  PV UUID               h835Dl-MzyW-qeXo-lgCI-llRQ-CyLS-m12MSP
```

The most important information will be the PV name and VG name. With those two pieces of information we can continue working on the LVM setup.

Creating a Logical Volume

Logical volumes are the partitions that your operating system uses in LVM. To create a logical volume we first need to have a physical volume and volume group. Here are all of the steps necessary to create a new logical volume.

Create physical volume

We will start from scratch with a brand new hard drive with no partitions or information on it. Start by finding which disk you will be working with. (/dev/sda, sdb, etc.)

Note: Remember all of the commands will need to be run as root or by adding 'sudo' to the beginning of the command.

```
fdisk -l
```

If your hard drive has never been formatted or partitioned before you will probably see something like this in the fdisk output. This is completely fine because we are going to create the needed partitions in the next steps.

```
Disk /dev/sdb doesn't contain a valid partition table
```

Our new disk is located at /dev/sdb so lets use fdisk to create a new partition on the drive.

There are a plethora of tools that can create a new partition with a GUI, including Gparted, but since we have the terminal open already, we will use fdisk to create the needed partition.

From a terminal type the following commands:

```
fdisk /dev/sdb
```

This will put you in a special fdisk prompt.

```
WARNING: DOS-compatible mode is deprecated. It's strongly recommended to
         switch off the mode (command 'c') and change display units to
         sectors (command 'u').

Command (m for help):
```

Enter the commands in the order given to create a new primary partition that uses 100% of the new hard drive and is ready for LVM. If you need to change the partition size or want multiple partions I suggest using GParted or reading about fdisk on your own.

Warning: The following steps will format your hard drive. Make sure you don't have any information on this hard drive before following these steps.

- n = create new partition
- p = creates primary partition
- 1 = makes partition the first on the disk

Push enter twice to accept the default first cylinder and last cylinder.

```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-1044, default 1):
Using default value 1
Last cylinder, +cylinders or +size{K,M,G} (1-1044, default 1044):
Using default value 1044
```

To prepare the partition to be used by LVM use the following two commands.

- t = change partition type
- 8e = changes to LVM partition type

```
Command (m for help): t
Selected partition 1
Hex code (type L to list codes): 8e
Changed system type of partition 1 to 8e (Linux LVM)
```

Verify and write the information to the hard drive.

- p = view partition setup so we can review before writing changes to disk
- w = write changes to disk

```
Command (m for help): p

Disk /dev/sdb: 8589 MB, 8589934592 bytes
255 heads, 63 sectors/track, 1044 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: 0xdf641837

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1               1        1044     8385898+  8e  Linux LVM

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
root@ubuntu:/#
```

After those commands, the fdisk prompt should exit and you will be back to the bash prompt of your terminal.

Enter `pvcreate /dev/sdb1` to create a LVM physical volume on the partition we just created.

You may be asking why we didn't format the partition with a file system but don't worry, that step comes later.

```
root@ubuntu:/# pvcreate /dev/sdb1
  Physical volume "/dev/sdb1" successfully created
```

Create volume Group

Now that we have a partition designated and physical volume created we need to create the volume group. Luckily this only takes one command.

```
vgcreate vgpool /dev/sdb1
```

```
root@ubuntu:/# vgcreate vgpool /dev/sdb1
  Volume group "vgpool" successfully created
```

Vgpool is the name of the new volume group we created. You can name it whatever you'd like but it is recommended to put vg at the front of the label so if you reference it later you will know it is a volume group.

Create logical volume

To create the logical volume that LVM will use:

```
lvcreate -L 3G -n lvstuff vgpool
```

```
root@ubuntu:/# lvcreate -L 3G -n lvstuff vgpool
  Logical volume "lvstuff" created
```

The -L command designates the size of the logical volume, in this case 3 GB, and the -n command names the volume. Vgpool is referenced so that the lvcreate command knows what volume to get the space from.

Format and Mount the Logical Volume

One final step is to format the new logical volume with a file system. If you want help choosing a Linux file system, read our how to that can help you choose the best file system for your needs.

```
mkfs -t ext3 /dev/vgpool/lvstuff
```

```
root@ubuntu:/# mkfs -t ext3 /dev/vgpool/lvstuff
mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
196608 inodes, 786432 blocks
39321 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=805306368
24 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912

Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 23 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

Create a mount point and then mount the volume somewhere you can use it.

```
mkdir /mnt/stuff
mount -t ext3 /dev/vgpool/lvstuff /mnt/stuff
```

```
root@ubuntu:/# mkdir /mnt/stuff
root@ubuntu:/# mount -t ext3 /dev/vgpool/lvstuff /mnt/stuff
```

Resizing a Logical Volume

One of the benefits of logical volumes is you can make your shares physically bigger or smaller without having to move everything to a bigger hard drive. Instead, you can add a new hard drive and extend your volume group on the fly. Or if you have a hard drive that isn't used you can remove it from the volume group to shrink your logical volume.

There are three basic tools for making physical volumes, volume groups, and logical volumes bigger or smaller.

Note: Each of these commands will need to be preceded by pv, vg, or lv depending on what you are working with.

- `resize` – can shrink or expand physical volumes and logical volumes but not volume groups
- `extend` – can make volume groups and logical volumes bigger but not smaller
- `reduce` – can make volume groups and logical volumes smaller but not bigger

Let's walk through an example of how to add a new hard drive to the logical volume "lvstuff" we just created.

Install and Format new Hard Drive

To install a new hard drive follow the steps above to create a new partition and add change it's partition type to LVM (8e). Then use pvcreate to create a physical volume that LVM can recognize.

Add New Hard Drive to Volume Group

To add the new hard drive to a volume group you just need to know what your new partition is, /dev/sdc1 in our case, and the name of the volume group you want to add it to.

This will add the new physical volume to the existing volume group.

```
vgextend vgpool /dev/sdc1
```

```
root@ubuntu:/# vgextend vgpool /dev/sdc1
  Volume group "vgpool" successfully extended
```

Extend Logical Volume

To resize the logical volume we need to say how much we want to extend by size instead of by device. In our example we just added a 8 GB hard drive to our 3 GB vgpool. To make that space usable we can use lvextend or lvresize.

```
lvextend -L8G /dev/vgpool/lvstuff
```

```
root@ubuntu:/# lvextend -L8G /dev/vgpool/lvstuff
  Extending logical volume lvstuff to 8.00 GiB
  Logical volume lvstuff successfully resized
```

While this command will work you will see that it will actually resize our logical volume to 8 GB instead of adding 8 GB to the existing volume like we wanted. To add the last 3 available gigabytes you need to use the following command.

```
lvextend -L+3G /dev/vgpool/lvstuff
```

```
root@ubuntu:/# lvextend -L+3G /dev/vgpool/lvstuff
  Extending logical volume lvstuff to 11.00 GiB
  Logical volume lvstuff successfully resized
```

Now our logical volume is 11 GB in size.

Extend File System

The logical volume is 11 GB but the file system on that volume is still only 3 GB. To make the file system use the entire 11 GB available you have to use the command resize2fs. Just point resize2fs to the 11 GB logical volume and it will do the magic for you.

```
resize2fs /dev/vgpool/lvstuff
```

```
root@ubuntu:/# resize2fs /dev/vgpool/lvstuff
resize2fs 1.41.12 (17-May-2010)
Filesystem at /dev/vgpool/lvstuff is mounted on /mnt/stuff; on-line resizing req
uired
old desc_blocks = 1, new_desc_blocks = 1
Performing an on-line resize of /dev/vgpool/lvstuff to 2883584 (4k) blocks.
The filesystem on /dev/vgpool/lvstuff is now 2883584 blocks long.
```

Note: If you are using a different file system besides ext3/4 please see your file systems resize tools.

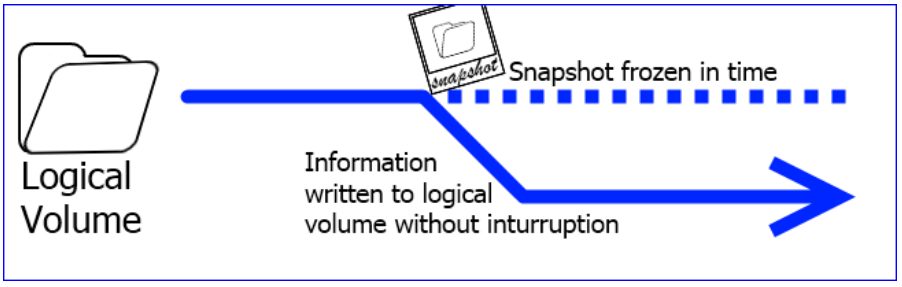
Shrink Logical Volume

If you wanted to remove a hard drive from a volume group you would need to follow the above steps in reverse order and use lvreduce and vgreduce instead.

1. resize file system (make sure to move files to a safe area of the hard drive before resizing)
2. reduce logical volume (instead of + to extend you can also use – to reduce by size)
3. remove hard drive from volume group with vgreduce

Backing up a Logical Volume

Snapshots is a feature that some newer advanced file systems come with but ext3/4 lacks the ability to do snapshots on the fly. One of the coolest things about LVM snapshots is your file system is never taken offline and you can have as many as you want without taking up extra hard drive space.



When LVM takes a snapshot, a picture is taken of exactly how the logical volume looks and that picture can be used to make a copy on a different hard drive. While a copy is being made, any new information that needs to be added to the logical volume is written to the disk just like normal, but changes are tracked so that the original picture never gets destroyed.

To create a snapshot we need to create a new logical volume with enough free space to hold any new information that will be written to the logical volume while we make a backup. If the drive is not actively being written to you can use a very small amount of storage. Once we are done with our backup we just remove the temporary logical volume and the original logical volume will continue on as normal.

Create New Snapshot

To create a snapshot of lvstuff use the lvcreate command like before but use the -s flag.

```
lvcreate -L512M -s -n lvstuffbackup /dev/vgpool/lvstuff
```

```
root@ubuntu:/# lvcreate -L512M -s -n lvstuffbackup /dev/vgpool/lvstuff
  Logical volume "lvstuffbackup" created
```

Here we created a logical volume with only 512 MB because the drive isn't being actively used. The 512 MB will store any new writes while we make our backup.

Mount New Snapshot

Just like before we need to create a mount point and mount the new snapshot so we can copy files from it.

```
mkdir /mnt/lvstuffbackup
mount /dev/vgpool/lvstuffbackup /mnt/lvstuffbackup
```

```
root@ubuntu:/# mkdir /mnt/lvstuffbackup
root@ubuntu:/# mount /dev/vgpool/lvstuffbackup /mnt/lvstuffbackup/
```

Copy Snapshot and Delete Logical Volume

All you have left to do is copy all of the files from /mnt/lvstuffbackup/ to an external hard drive or tar it up so it is all in one file.

Note: tar -c will create an archive and -f will say the location and file name of the archive. For help with the tar command use `man tar` *in the terminal.*

```
tar -cf /home/rothgar/Backup/lvstuff-ss /mnt/lvstuffbackup/
```

```
root@ubuntu:/# tar -cf /home/rothgar/Backup/lvstuff-ss /mnt/lvstuffbackup/
tar: Removing leading `/' from member names
```

Remember that while the backup is taking place all of the files that would be written to lvstuff are being tracked in the temporary logical volume we created earlier. Make sure you have enough free space while the backup is happening.

Once the backup finishes, unmount the volume and remove the temporary snapshot.

```
umount /mnt/lvstuffbackup
lvremove /dev/vgpool/lvstuffbackup/
```

```
root@ubuntu:/# umount /mnt/lvstuffbackup/
root@ubuntu:/# lvremove /dev/vgpool/lvstuffbackup
Do you really want to remove active logical volume lvstuffbackup? [y/n]: y
  Logical volume "lvstuffbackup" successfully removed
```

Deleting a Logical Volume

To delete a logical volume you need to first make sure the volume is unmounted, and then you can use lvremove to delete it. You can also remove a volume group once the logical volumes have been deleted and a physical volume after the volume group is deleted.

Here are all the commands using the volumes and groups we've created.

```
umount /mnt/lvstuff
lvremove /dev/vgpool/lvstuff
vgremove vgpool
pvremove /dev/sdb1 /dev/sdc1
```

```
root@ubuntu:/# umount /mnt/stuff
root@ubuntu:/# lvremove /dev/vgpool/lvstuff
Do you really want to remove active logical volume lvstuff? [y/n]: y
  Logical volume "lvstuff" successfully removed
root@ubuntu:/# vgremove vgpool
  Volume group "vgpool" successfully removed
root@ubuntu:/# pvremove /dev/sdb1 /dev/sdc1
  Labels on physical volume "/dev/sdb1" successfully wiped
  Labels on physical volume "/dev/sdc1" successfully wiped
```

That should cover most of what you need to know to use LVM. If you've got some experience on the topic, be sure to share your wisdom in the comments.

JOIN THE DISCUSSION

Justin is a Linux and HTPC enthusiast who loves to try new projects. He isn't scared of bricking a cell phone in the name of freedom.

• Published 02/7/11

SHOW ARCHIVED READER COMMENTS (8)