**Create LVM Partition in Linux**

**For scanning new LUN use below command:-**

$ls /sys/class/scsi\_host/ | while read host ; do echo "- - -" > /sys/class/scsi\_host/$host/scan ; done

How to create a LVM in Linux, these are the below steps to be followed.

1. Select or Identify the correct disks to be used for LVM.

2. Create a Physical Volumes(PV) on the disk.

3. Create the Volume Group(VG) on the Physical Volumes

4. Create Logical Volumes(LV) on the Volume Group

5. Create a filesystem for the logical vlume

we will use three disks of 5 GB each (/dev/sda5, /dev/sda6, and /dev/sda7) to create three physical volumes. You can either create the PVs directly on top of the device, or partition it first.

**Create a Physical Volumes(PV) Volume Groups(VG), and Logical Volumes(LV) on the disk.**

**Physical volume creation:-**

$ pvcreate /dev/sda5 /dev/sda6 /dev/sda7

You can list the newly created PVs with:

$pvs



and get detailed information about each PV with**:**

**$ pvdisplay /dev/sdX**

**(where X is b, c, or d)**



**Volume Groups(VG) Creation:-**

**To create a volume group named vg00 using /dev/sda5 and /dev/sda6 (we will save /dev/sda7 for later to illustrate the possibility of adding other devices to expand storage capacity when needed):**

**$** vgcreate vg00 /dev/sda5 /dev/sda6



Since vg00 is formed with two 5 GB disks, it will appear as a single 10 GB drive:



**Logical volume(LV) creation**:-

For example, let’s create two LVs named vol\_projects (5 GB) and vol\_backups (remaining space), which we can use later to store project documentation and system backups, respectively.

The -n option is used to indicate a name for the LV,whereas -L sets a fixed size and -l (lowercase L) is used to indicate a percentage of the remaining space in the container VG.

$ lvcreate -n vol\_project -L 5G vg00

$ lvcreate -n vol\_backup -l 100%FREE vg00



**To view information about a single LV, use lvdisplay with the VG and LV as parameters, as follows:**

# lvdisplay /dev/vg00/vol\_project



**We’ll use ext4 as an example here since it allows us both to increase and reduce the size of each LV (as opposed to xfs that only allows to increase the size):**

$mkfs.ext4 /dev/vg00/vol\_project

$mkfs.ext4 /dev/vg00/vol\_project



**Mounting the Partitions:-**

Create the directory and mount the FS as shown in below



**Permanently Mounting:-**

We have to give entires in /etc/fstab as shown in below.



**Extending LVM**:-

We dont require any down time for extending LVM.

First we need to check is there any free space in VG other wise we need to extend VG by PV and then VG to LV as shown in below.

In first SS there is no free space in vg00 so we need to extend vg00 by physcial volume.



In second SS vg00 extended by physcial volume (/dev/sda7) with 5GB.



Before extention of LV



After extention of LV:-

$lvextend -L +2G -r /dev/vg00/vol\_project

$lvextend -l +100%FREE -r /dev/vg00/vol\_backup



Remaining full space is given to /backup



**Reducing the LVM:-**

In below SS /backup is having 5.6GB we are reducing to 500MB.



For reducing the size we need application downtime because we should unmount the filesystem in fstab so we are unmounting file system as shown in below SS.





After unmounting we need to check file system errors and reduce the size by using below commands.

**check the file system for Errors using e2fsck command**:-

e2fsck /dev/vg00/vol\_backupk



Step:3 Reduce or Shrink the size of /backup to desire size.

$lvreduce -L -500M /dev/mapper/vg00-vol\_backup



**Update the reduce by using below command:-**



**After resize2fs use mount -a it will automatically mount the FS. Below SS after reducing the FS.**

