RAID:

RAID (Redundant Array of Independent Disks) in Linux is a technology that combines multiple physical drives into a single logical unit.

What is RAID 0 (Striping)?

- **Definition**: Splits data into chunks and distributes them across two or more disks.
- **Purpose**: Improves performance (I/O throughput).
- **Tradeoff**: No redundancy — failure of any disk causes total data loss.

Why RAID 0?

- Splits data into chunks and distributes ("stripes") them across two or more disks.
- **Boosts performance**—read/write speeds scale with the number of disks.
- **Lacks redundancy**: if any drive fails, you lose the entire array

To configure **RAID 0 (striping)** on Linux:

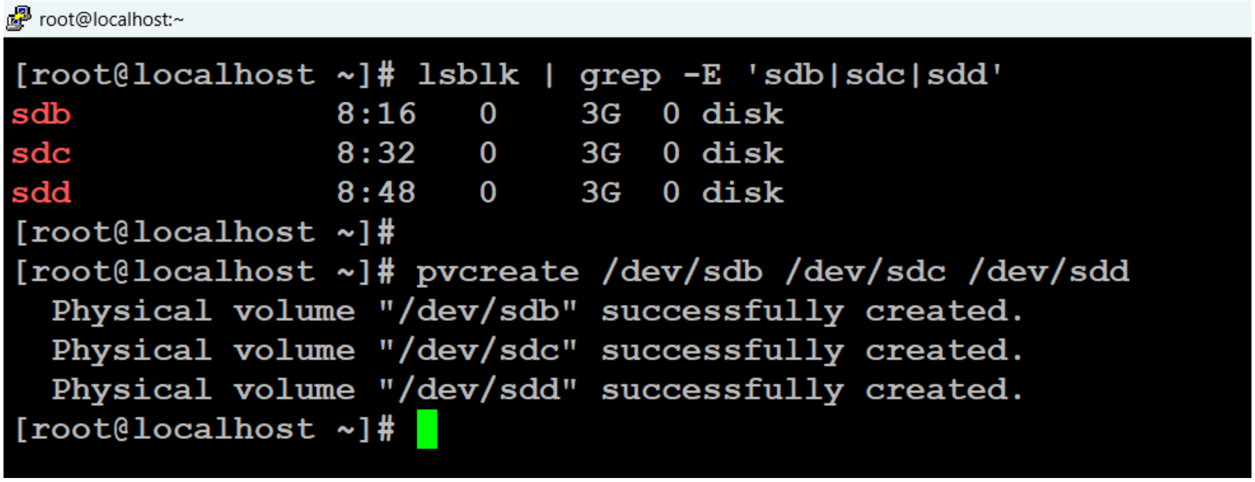
1. LVM (Logical Volume Manager)
2. MDADM (Multiple Device Administration)

RAID 0 Striping with LVM – Full Workflow

Step 1: Add New Disks & Initialize as PVs

Once you physically attach or virt- add your disks (e.g., /dev/sdb, /dev/sdc & /dev/sdd), format them as LVM PVs:

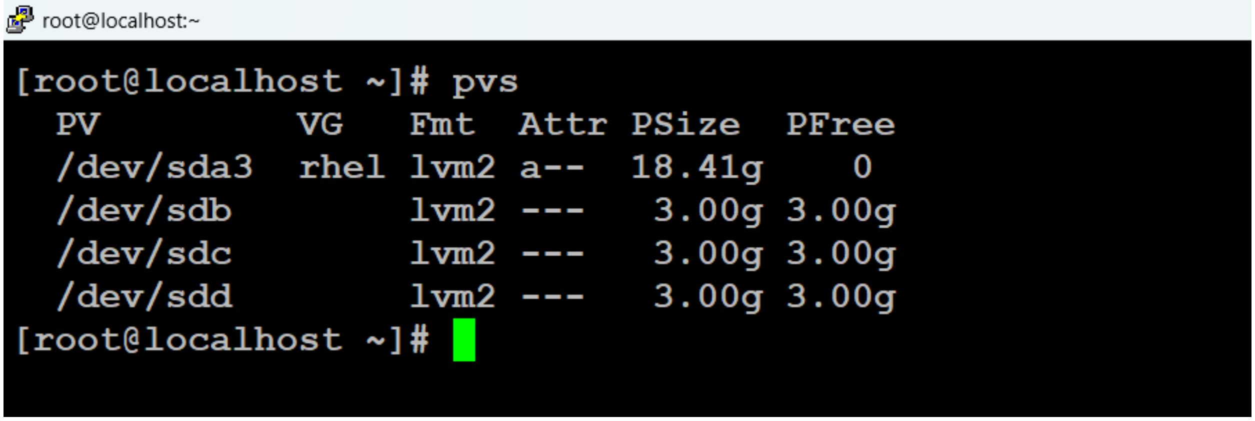
Command: pvcreate /dev/sdb /dev/sdc /dev/sdd



```
[root@localhost ~]# lsblk | grep -E 'sdb|sdc|sdd'
sdb                8:16   0    3G  0 disk
sdc                8:32   0    3G  0 disk
sdd                8:48   0    3G  0 disk
[root@localhost ~]#
[root@localhost ~]# pvcreate /dev/sdb /dev/sdc /dev/sdd
  Physical volume "/dev/sdb" successfully created.
  Physical volume "/dev/sdc" successfully created.
  Physical volume "/dev/sdd" successfully created.
[root@localhost ~]#
```

Creates three PVs ready for LVM usage.

Check with:

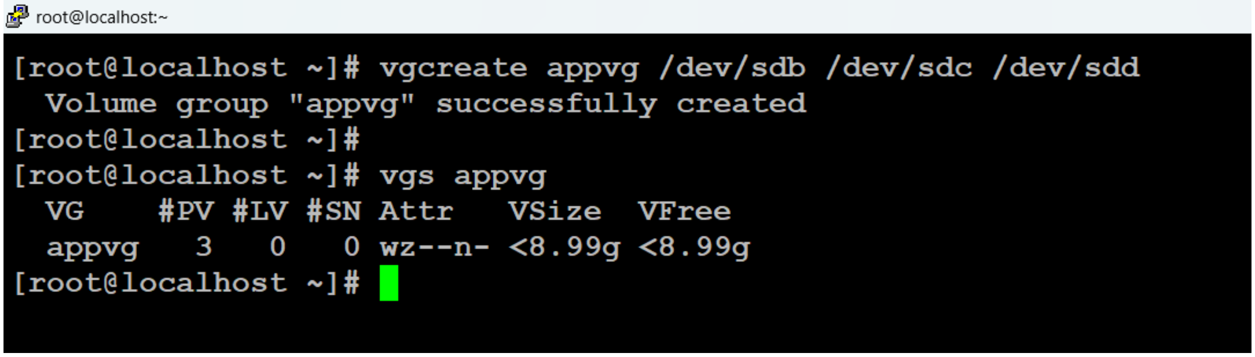


```
[root@localhost ~]# pvs
  PV         VG   Fmt  Attr PSize  PFree
  /dev/sda3  rhel lvm2 a--  18.41g    0
  /dev/sdb        lvm2 ---   3.00g 3.00g
  /dev/sdc        lvm2 ---   3.00g 3.00g
  /dev/sdd        lvm2 ---   3.00g 3.00g
[root@localhost ~]#
```

Step 2: Create a Volume Group (VG)

Combine the PVs into a volume group:

Command: vgcreate appvg /dev/sdb /dev/sdc /dev/sdd



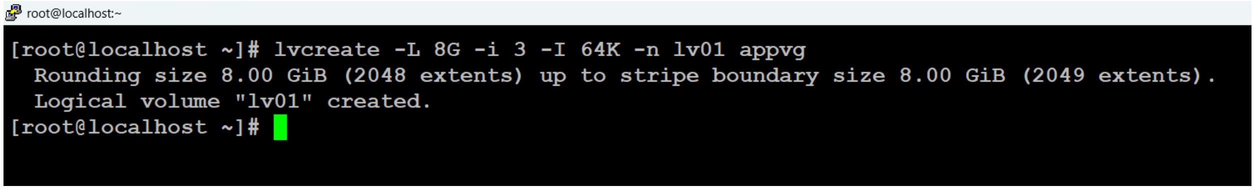
```
[root@localhost ~]# vgcreate appvg /dev/sdb /dev/sdc /dev/sdd
  Volume group "appvg" successfully created
[root@localhost ~]#
[root@localhost ~]# vgs appvg
  VG     #PV #LV #SN Attr   VSize  VFree
  appvg    3   0   0 wz--n- <8.99g <8.99g
[root@localhost ~]#
```

Combines the PVs into appvg

Step 3: Create a Striped (RAID-0) Logical Volume

To stripe across the 3 PVs:

Command: lvcreate -L 8G -i 3 -I 64K -n lv01 appvg

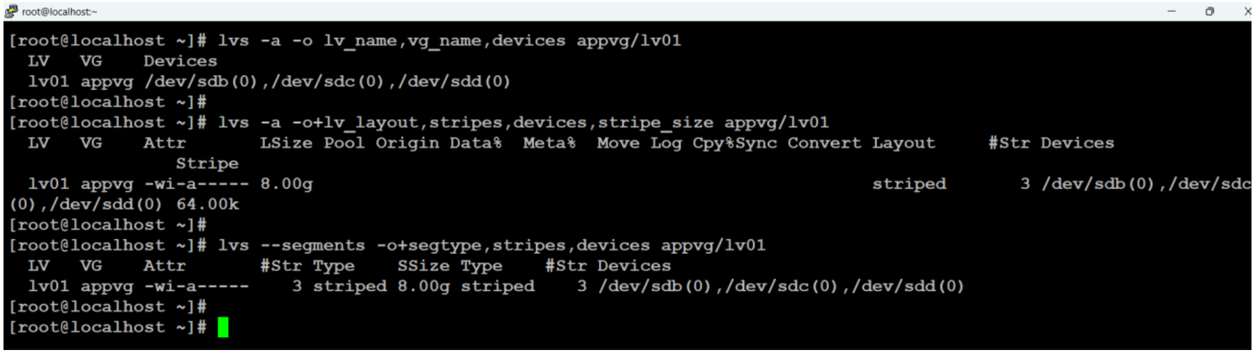


```
[root@localhost ~]# lvcreate -L 8G -i 3 -I 64K -n lv01 appvg
  Rounding size 8.00 GiB (2048 extents) up to stripe boundary size 8.00 GiB (2049 extents).
  Logical volume "lv01" created.
[root@localhost ~]#
```

- -i 3: uses 3 stripes (one per PV)
- -I 64K: stripe chunk size (adjustable)
- -L 8G: size of the LV

Maximum stripes cannot exceed the #PVs in VG

Check with:



```
[root@localhost ~]# lvs -a -o lv_name,vg_name,devices appvg/lv01
  LV   VG    Devices
  lv01 appvg /dev/sdb(0),/dev/sdc(0),/dev/sdd(0)
[root@localhost ~]#
[root@localhost ~]# lvs -a -o+lv_layout,stripes,devices,stripe_size appvg/lv01
  LV   VG    Attr       LSize Pool Origin Data%  Meta%  Move Log Cpy%Sync Convert Layout     #Str Devices
                Stripe
  lv01 appvg -wi-a----- 8.00g                                                     striped       3 /dev/sdb(0),/dev/sdc
(0),/dev/sdd(0) 64.00k
[root@localhost ~]#
[root@localhost ~]# lvs --segments -o+segtype,stripes,devices appvg/lv01
  LV   VG    Attr       #Str Type    SSize Type    #Str Devices
  lv01 appvg -wi-a-----    3 striped 8.00g striped    3 /dev/sdb(0),/dev/sdc(0),/dev/sdd(0)
[root@localhost ~]#
[root@localhost ~]#
```

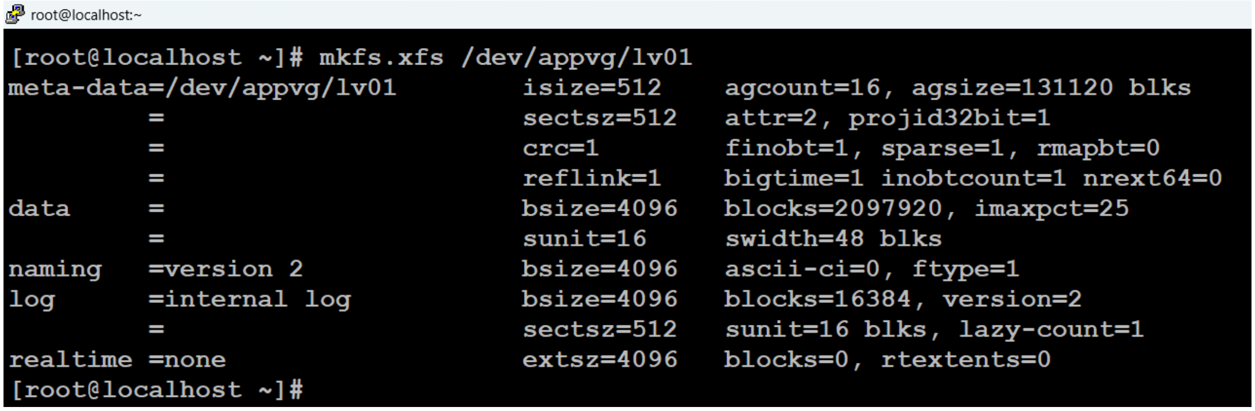
- **SegType:** striped
- **Stripes:** should show 3
- **Devices:** lists PVs used

Step 4: Format & Mount the LV

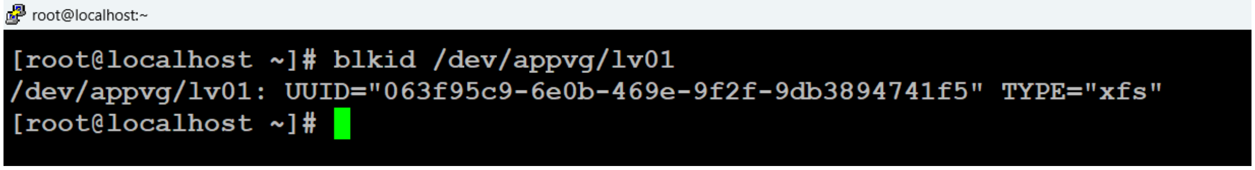
Format the new LV and mount it:

Commands:

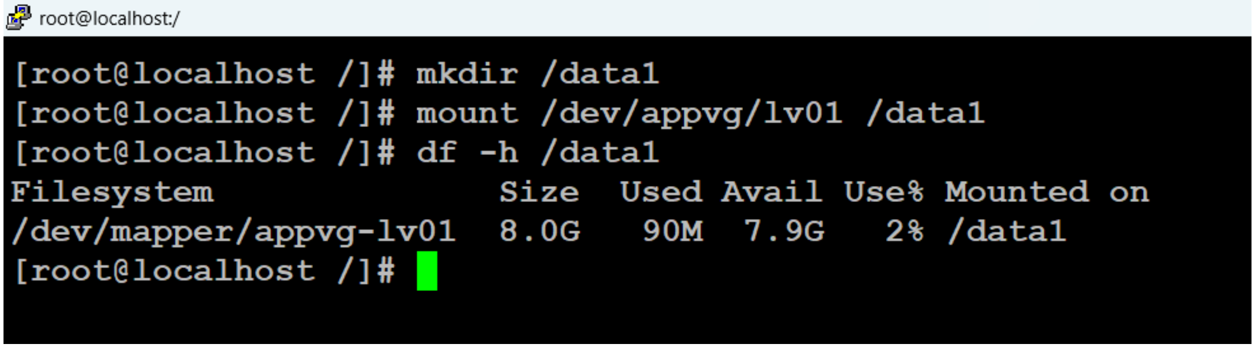
mkfs.xfs /dev/appvg/lv01
mkdir /data1
mount /dev/appvg/lv01 /data1



```
[root@localhost ~]# mkfs.xfs /dev/appvg/lv01
meta-data=/dev/appvg/lv01        isize=512    agcount=16, agsize=131120 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=1, sparse=1, rmapbt=0
         =                       reflink=1    bigtime=1 inobtcount=1 nrext64=0
data     =                       bsize=4096   blocks=2097920, imaxpct=25
         =                       sunit=16     swidth=48 blks
naming   =version 2              bsize=4096   ascii-ci=0, ftype=1
log      =internal log           bsize=4096   blocks=16384, version=2
         =                       sectsz=512   sunit=16 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
[root@localhost ~]#
```



```
[root@localhost ~]# blkid /dev/appvg/lv01
/dev/appvg/lv01: UUID="063f95c9-6e0b-469e-9f2f-9db3894741f5" TYPE="xfs"
[root@localhost ~]#
```

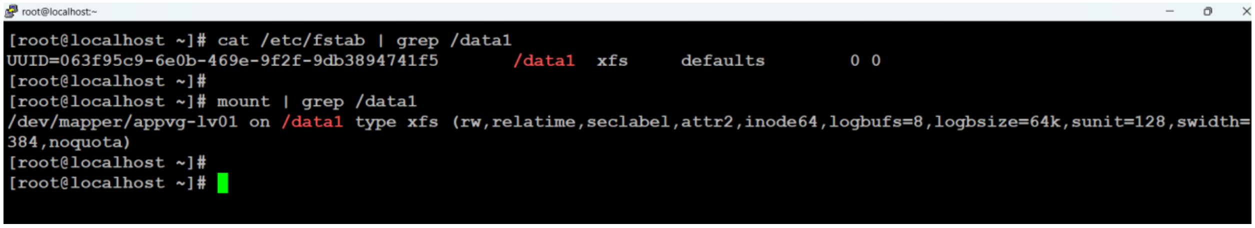


```
[root@localhost /]# mkdir /data1
[root@localhost /]# mount /dev/appvg/lv01 /data1
[root@localhost /]# df -h /data1
Filesystem              Size  Used Avail Use% Mounted on
/dev/mapper/appvg-lv01  8.0G   90M  7.9G   2% /data1
[root@localhost /]#
```

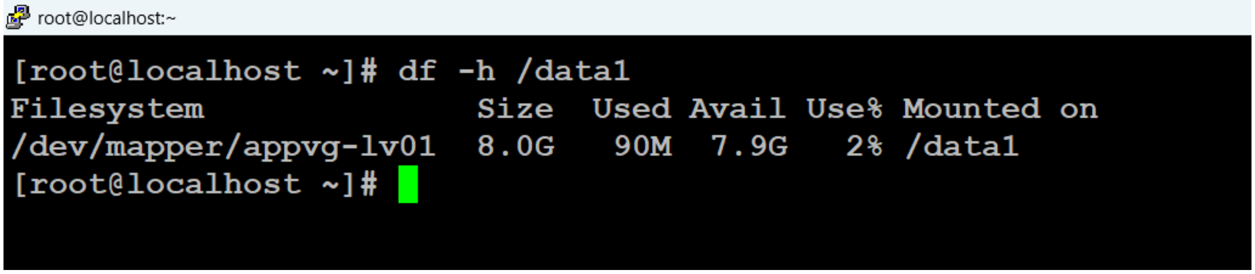
Step 5: Add entry to /etc/fstab for persistent mount:

Create a stable mount entry so that `/data1` mounts automatically at every boot.

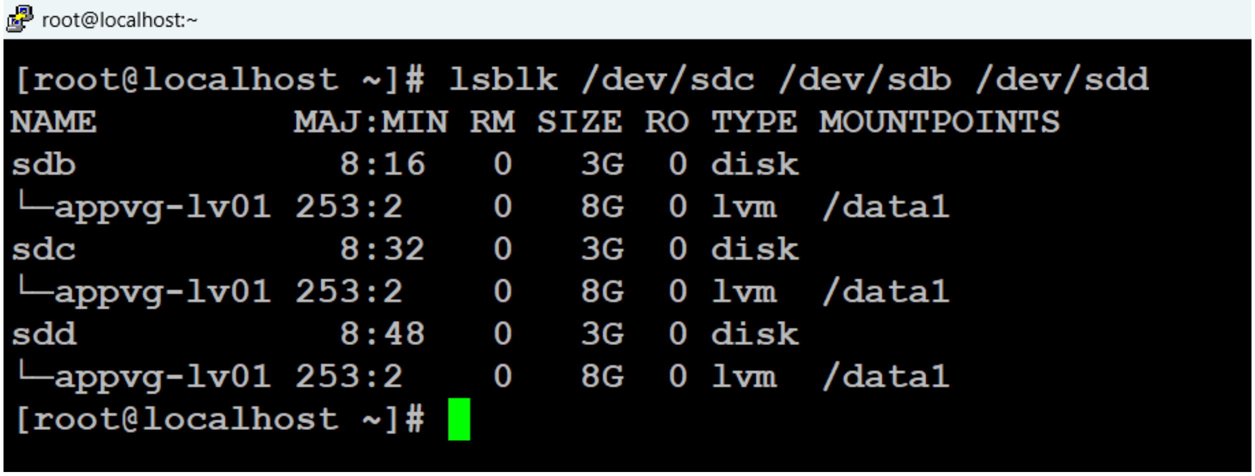
Command: vi /etc/fstab



```
[root@localhost ~]# cat /etc/fstab | grep /data1
UUID=063f95c9-6e0b-469e-9f2f-9db3894741f5       /data1  xfs     defaults        0 0
[root@localhost ~]#
[root@localhost ~]# mount | grep /data1
/dev/mapper/appvg-lv01 on /data1 type xfs (rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=64k,sunit=128,swidth=
384,noquota)
[root@localhost ~]#
[root@localhost ~]#
```



```
[root@localhost ~]# df -h /data1
Filesystem              Size  Used Avail Use% Mounted on
/dev/mapper/appvg-lv01  8.0G   90M  7.9G   2% /data1
[root@localhost ~]#
```



```
[root@localhost ~]# lsblk /dev/sdc /dev/sdb /dev/sdd
NAME          MAJ:MIN RM SIZE RO TYPE MOUNTPOINTS
sdb             8:16   0   3G  0 disk
└─appvg-lv01 253:2     0   8G  0 lvm  /data1
sdc             8:32   0   3G  0 disk
└─appvg-lv01 253:2     0   8G  0 lvm  /data1
sdd             8:48   0   3G  0 disk
└─appvg-lv01 253:2     0   8G  0 lvm  /data1
[root@localhost ~]#
```

Final Takeaway

RAID 0 gives you **speed** and **capacity efficiency**, but **no protection** against disk failure. When combined with a solid backup solution, it creates a balanced system: high performance with robust data safety