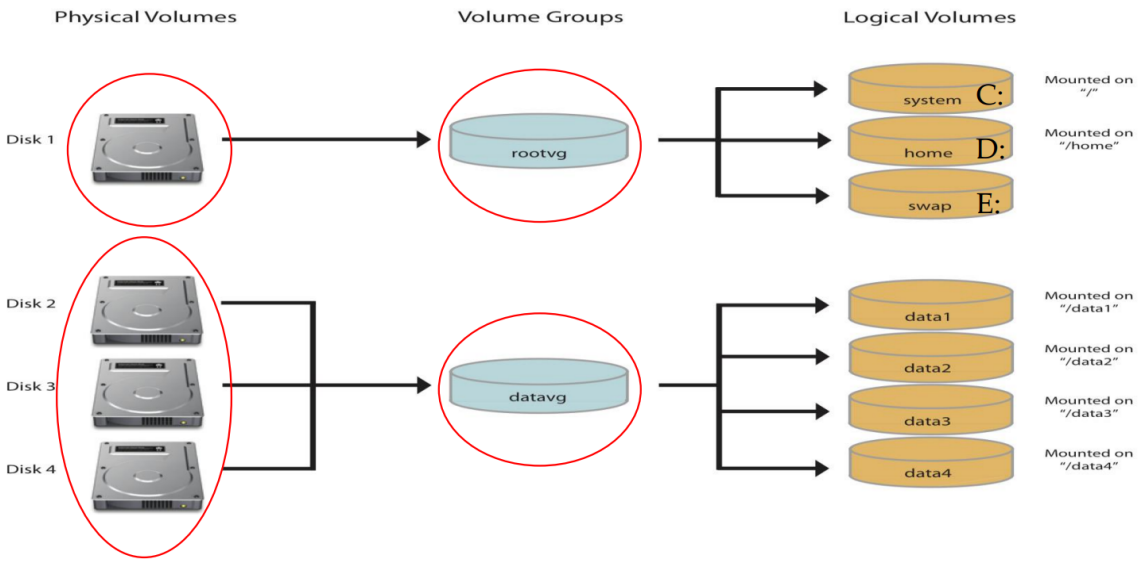
Manage logical volumes

Linux #redhat #filesystem #storage #LVM

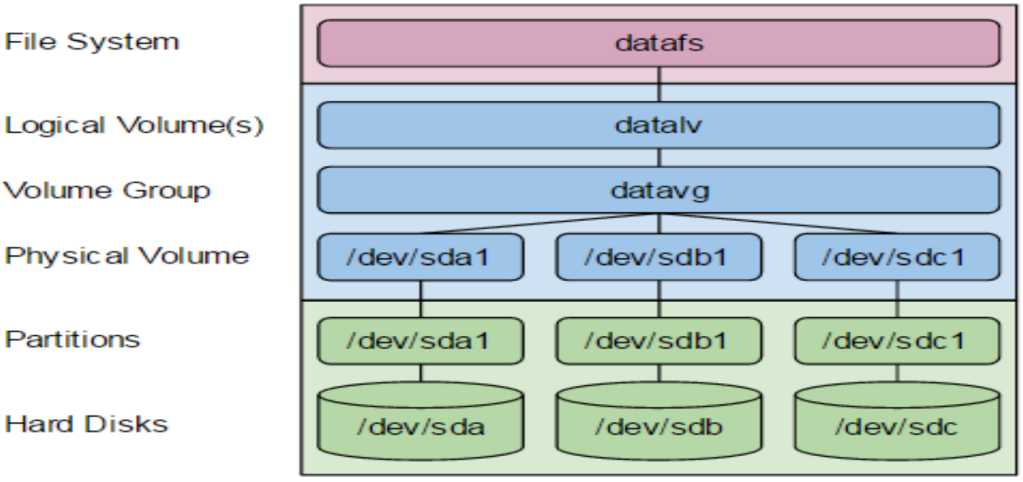
Logical Volume Management

- LVM allows disks to be combined
  - It is a software-based utility



- Instead of taking one disk and assign it to one partition or one mount point, you take all three disks combined together and make it a volume group, call it whatever you want it to call, if there were 3 disks of 100G each in this group, this group becomes a 300G volume. Then you create data1, data2, data3, data4 in different partitions.
- The main advantage of LVM is what if you run out 300G that you have created below, then you could add a fourth disk and without breaking your partitions or your filesystem, you could add that disk to your group, your volume group, which at this point is `datavg`, and extend the group size, so it becomes 400G if the disk added is 100G. Then you could associate that size to data1, if data1 disk space is run out of space.
- LVM is very useful in the corporate environment because you have to extend the disk space or partitions of your mount points all the time.
  - If the server runs out of space, if you have a physical server the only option you have is to pop the tray out and add another disk, but what if there is no capacity to add another disk? You are out of luck.
  - In a virtual environment you could add another disk or extend a disk in virtual machine. Once it's extended, you could extend your group, and then you could partition that or extend your data1, data2, etc. petitions.

Add Disk and create LVM Partition



- At the bottom of this picture you will see we have hard disks, sda, sdb, and sdc.
- For every hard disk we created a partition on top of it.
- If you are doing LVM on top of it, then you will have to create Physical Volume for each partition, then we create a volume group. That group will represent the entire physical volumes that will be coming out of those disks.
- Then you will create Logical Volume.
- Once we have the logical volume, which will be exactly the same as a standard volume, but of course this will be under LVM, then we are going to mount those logical volumes to our filesystem.

Steps to add and create LVM Partition

1. **Add disk physically or add a disk to your VM environment**. (More on Manage basic storage)
2. Once the disk was physically added, **boot your Linux machine and become root**
3. **Run the `fdisk -l` command to list the disks in your system**, here you will see the new inserted disk

Output from `fdisk -l`:

```
...
Disk /dev/sdc: 1 GiB, 1073741824 bytes, 2097152 sectors
Disk model: VBOX HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
...
```

- Note the dame `/dev/sdc` was assigned automatically as we already have an sda, and sdb disk.

4. **Create the partition**:
    - Run `fdisk /dev/sdc` to enter the disk with the `fdisk` utility
    - Enter `n` for "new" to create a new partition on the disk
    - Enter `p` for "primary partition"
    - Then leave the next prompted questions as defualt and hit Enter
    - In this case we want it to give the partition the entire size of the disk so we will also leave it default

Example using `fdisk` command:

```
[root@localhost ~]# fdisk /dev/sdc

Welcome to fdisk (util-linux 2.37.4).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0x2c41cb50.

Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1):
First sector (2048-2097151, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-2097151, default 2097151):

Created a new partition 1 of type 'Linux' and of size 1023 MiB.
```

5. To verify that you created your partition on this disk, while on the `fdisk` utility, you can **Enter `p` for "partitions" to list the partitions of the current `fdisk` disk.**
   - Enter `p`

Example using `p` command:

```
Command (m for help): p
Disk /dev/sdc: 1 GiB, 1073741824 bytes, 2097152 sectors
Disk model: VBOX HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x2c41cb50

Device     Boot Start     End Sectors  Size Id Type
/dev/sdc1        2048 2097151 2095104 1023M 83 Linux
```

- The partition has been created, but it has not been written to the table yet until we do `w` to "write", but at this time since we are doing LVM, we will need to type `t`

6. **Change the type of this partition by entering `t` for "type"** while on the selected disk on `fdisk` utility. Finally hit `w` to write changes to the table.
   - Enter `t` for "type"
   - Enter `L` to list Hex code or alias
   - Enter `8e` for Linux LVM
   - Enter `w` to "write" changes

Example using `t` command:

```
Command (m for help): t
Selected partition 1
Hex code or alias (type L to list all): L
```

```
00 Empty            24 NEC DOS          81 Minix / old Lin  bf Solaris
01 FAT12            27 Hidden NTFS Win  82 Linux swap / So  c1 DRDOS/sec (FAT-
02 XENIX root       39 Plan 9           83 Linux            c4 DRDOS/sec (FAT-
03 XENIX usr        3c PartitionMagic   84 OS/2 hidden or   c6 DRDOS/sec (FAT-
04 FAT16 <32M       40 Venix 80286      85 Linux extended   c7 Syrinx
05 Extended         41 PPC PReP Boot    86 NTFS volume set  da Non-FS data
06 FAT16            42 SFS              87 NTFS volume set  db CP/M / CTOS / .
07 HPFS/NTFS/exFAT  4d QNX4.x           88 Linux plaintext  de Dell Utility
08 AIX              4e QNX4.x 2nd part  8e Linux LVM        df BootIt
09 AIX bootable     4f QNX4.x 3rd part  93 Amoeba           e1 DOS access
0a OS/2 Boot Manag  50 OnTrack DM       94 Amoeba BBT       e3 DOS R/O
0b W95 FAT32        51 OnTrack DM6 Aux  9f BSD/OS           e4 SpeedStor
0c W95 FAT32 (LBA)  52 CP/M             a0 IBM Thinkpad hi  ea Linux extended
0e W95 FAT16 (LBA)  53 OnTrack DM6 Aux  a5 FreeBSD          eb BeOS fs
0f W95 Ext'd (LBA)  54 OnTrackDM6       a6 OpenBSD          ee GPT
10 OPUS             55 EZ-Drive         a7 NeXTSTEP         ef EFI (FAT-12/16/
11 Hidden FAT12     56 Golden Bow       a8 Darwin UFS       f0 Linux/PA-RISC b
12 Compaq diagnost  5c Priam Edisk      a9 NetBSD           f1 SpeedStor
14 Hidden FAT16 <3  61 SpeedStor        ab Darwin boot      f4 SpeedStor
16 Hidden FAT16     63 GNU HURD or Sys  af HFS / HFS+       f2 DOS secondary
17 Hidden HPFS/NTF  64 Novell Netware   b7 BSDI fs          fb VMware VMFS
18 AST SmartSleep   65 Novell Netware   b8 BSDI swap        fc VMware VMKCORE
1b Hidden W95 FAT3  70 DiskSecure Mult  bb Boot Wizard hid  fd Linux raid auto
1c Hidden W95 FAT3  75 PC/IX            bc Acronis FAT32 L  fe LANstep
1e Hidden W95 FAT1  80 Old Minix        be Solaris boot     ff BBT

Aliases:
   linux          - 83
   swap           - 82
   extended       - 05
   uefi           - EF
   raid           - FD
   lvm            - 8E
   linuxex        - 85
Hex code or alias (type L to list all): 8e
Changed type of partition 'Linux' to 'Linux LVM'.

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.
```

- The `t` command is to change a partition's system id
- After entering `t` we then enter `L` to list all Hex code or alias and look for the one that is associated with Linux LVM (`8e`)
- Now if we run `p` again to list the partitions of the disk, the partition `/dev/sdc1` that we just created will appear as Type Linux LVM
- Now our partition table is created.
  - You now can enter `w` to write partition table changes.

7. **Create physical volume:**
   - Run `pvcreate` followed by the name of the volume and the partition number
     - `pvcreate /dev/sdc1`
     - This will output: `Physical volume "/dev/sdc1" successfully created.`
   - To verify this, you can always run `pvdisplay`, which is to partition volume display

Example using `pvdisplay` command:

```
[root@localhost ~]# pvdisplay
```

Output:

```
  --- Physical volume ---
  PV Name               /dev/sda2
  VG Name               rhel
  PV Size               <19.00 GiB / not usable 3.00 MiB
  Allocatable           yes (but full)
  PE Size               4.00 MiB
  Total PE              4863
  Free PE               0
  Allocated PE          4863
  PV UUID               wrHyR0-8Pz0-34Lw-4xpp-Ztqn-Fv5m-f3TGWD

  "/dev/sdc1" is a new physical volume of "1023.00 MiB"
  --- NEW Physical volume ---
  PV Name               /dev/sdc1
  VG Name
  PV Size               1023.00 MiB
  Allocatable           NO
  PE Size               0
  Total PE              0
  Free PE               0
  Allocated PE          0
  PV UUID               BVN9hO-jckx-Sq7j-ewf6-De64-U29a-R5WShj
```

- It will tell you that it created the volume, with more information.

8. **Create volume group:**
   - Run `vgcreate` and the name of the volume group that you wanted to assign, finally add the name of the physical volume we previously created.
     - `vgcreate oracle_vg /dev/sdc1`
       - You could pick any name for this, in this case we picked 'oracle_vg' in order to identify this is a "volume group"
     - This will output: `Volume group "oracle_vg" successfully created`
   - You can verify this by running `vgdisplay` to list the volume groups with more information.

Example using `vgdisplay` command:

```
[root@localhost ~]# vgdisplay oracle_vg
```

- Note we only want to list information from the volume group that we just created 'oracle_vg'

Output:

```
  --- Volume group ---
  VG Name               oracle_vg
  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               1020.00 MiB
  PE Size               4.00 MiB
  Total PE              255
  Alloc PE / Size       0 / 0
  Free  PE / Size       255 / 1020.00 MiB
  VG UUID               uhqflf-4isP-H3Y8-hodl-Nss2-AMLQ-7kOffY
```

9. **Create a logical volume:**
   - Run the command `lvcreate`, then `-n` for "new", then the name you want to give to the logical volume (`oracle_lv`), then the size you want to give it (`--size 1G`), and finally the name of the volume group that it will be associated to (`oracle_vg`)
     - `lvcreate -n oracle_lv --size 1G oracle_vg`
   - Verify the action was completed by running `lvdisplay`

Example using `lvcreate` command:

```
[root@localhost ~]# lvcreate -n oracle_lv --size 1G oracle_vg
  Volume group "oracle_vg" has insufficient free space (255 extents): 256 required.
```

- Instead of doing 1G, we could do 1023 MB, which is one less Megabyte than a Gigabyte. (1G = 1024MB)

Example using `lvcreate` command:

```
[root@localhost ~]# lvcreate -n oracle_lv --size 1023M oracle_vg
  Rounding up size to full physical extent 1.00 GiB
  Volume group "oracle_vg" has insufficient free space (255 extents): 256 required.
```

- Note we got the same result so we will need to lower down the size of the logical volume

Example using `lvcreate` command:

```
[root@localhost ~]# lvcreate -n oracle_lv --size 1000M oracle_vg
  Logical volume "oracle_lv" created.
```

- Now we are just picking 1000MB because it does need some of the space to hold its LVM information.
- Now we created the logical volume

Example using `lvdisplay` command:

```
[root@localhost ~]# lvdisplay
```

- Note we cannot just check for one single logical volume, we need to list all of them.

Output:

```
  --- Logical volume ---
  LV Path                /dev/oracle_vg/oracle_lv
  LV Name                oracle_lv
  VG Name                oracle_vg
  LV UUID                9TTmCW-InfK-wqRy-ZTV4-ac4l-XQkQ-cVs41C
  LV Write Access        read/write
  LV Creation host, time linuxtest, 2024-07-03 13:16:04 -0600
  LV Status              available
  # open                 0
  LV Size                1000.00 MiB
  Current LE             250
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:2
...
```

10. **Format the disk and assign a filesystem to it:**
    - Run `mkfs.xfs /dev/oracle_vg/oracle_lv`
      - In the volume group, and then we have the logical volume that we just created
      - Note we are assigning xfs filesystem to this partition

Example using `mkfs.xfs` command:

```
[root@localhost ~]# mkfs.xfs /dev/oracle_vg/oracle_lv
```

Output:

```
meta-data=/dev/oracle_vg/oracle_lv isize=512    agcount=4, agsize=64000 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=256000, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0, ftype=1
log      =internal log           bsize=4096   blocks=16384, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
```

11. **Mount the logical volume to our directory** wherever that directory is:
    - Create a new directory with `mkdir`
      - `mkdir /oracle`
    - Mount LV to this directory
      - `mount /dev/oracle_vg/oracle_lv /oracle`
    - You can run `df -h` to verify this step.

Example using `mount` command:

```
[root@localhost ~]# mount /dev/oracle_vg/oracle_lv /oracle
```

```
[root@localhost ~]# df -h
```

Output:

```
...
/dev/mapper/oracle_vg-oracle_lv  936M   39M  898M   5% /oracle
```

- It now has assigned the volume to the `/oracle` directory.
- If you want you could also unmount it with the `umount` command.
- Remember you would have to modify `/etc/fstab` to let this disk mount while on boot. More on Manage basic storage (Please be careful with this step)
  - Add the line `/dev/mapper/oracle_vg-oracle_lv /oracle xfs defaults 0 0` to the `/etc/fstab` file to enable to mount the partition on boot. (Remember separations are done by tabs)

Add and Extend Disk Using LVM

- Imagine `/oracle` disk is completely full to 100%
  - `/oracle` = 1.0G
  - `/oracle` = Full
- Few Options:
  - Delete older files to free up disk space
  - Add new physical disk and mount it to `/oracle2`
    - If you have a physical system then you could add a new physical disk in this physical system if and only if you have the slot, an extra slot to add an extra disk to that physical system.
  - Create a new virtual disk and mount to `/oracle2`
    - All you have to do i go to your VM and add a new disk.
  - Extend `/oracle` through LVM
    - If you didn't set up `/oracle` to work with LVM you would be out of luck to extend this partition, but if you did, then you are in luck and you will be able to extend this partition using LVM
    - All you have to do is create a partition of a new disk and then attach that disk within the same group that you created for `/oracle` mount point.

Steps to extend disk using LVM

1. 1. **Add disk physically or add a disk to your VM environment**. (More on Manage basic storage)
2. Once the disk was physically added, **boot your Linux machine and become root**
3. **Run the `fdisk -l` command to list the disks in your system**, here you will see the new inserted disk. (replicated previously in this note)
4. **Partition that new disk**: (replicated previously in this note)
   - `fdisk /dev/sdd`
   - `n` to create a partition and select defaults by hitting Enter when prompted
   - `p` to verify partition
5. **Change the partition type:**
   - `t` for "type"
   - `8e` for Linux LVM
   - `p` to verify partition
   - `w` to write to partition table

- `fdisk -l /dev/sdd` to verify the partition was created and the type was changed

6. **Reboot the system** to make sure it is updated to within the system.
7. **Create a physical volume**:
   - Run `pvcreate /dev/sdd1`
     - We always have to create a physical volume and then add taht physical volume to the volume group.
   - Run `pvdisplay`
     - This will show us which group is associated with which disk.
     - You could also run `pvs`, it will also give you the same information in a briefly way.

Example using `pvs` command:

```
[root@localhost ~]# pvs
  PV         VG        Fmt  Attr PSize    PFree
  /dev/sda2  rhel      lvm2 a--   <19.00g       0
  /dev/sdc1  oracle_vg lvm2 a--  1020.00m   20.00m
  /dev/sdd1            lvm2 ---  1023.00m 1023.00m
```

- The `pvs` command displays information about physical volumes
- Note we want to add a new disk within the `oracle_vg` volume group
- Note the new physical volume is not part of any VG

8. **Extend the volume group:**
   - Run `vgextend oracle_vg /dev/sdd1`
     - Note we have to specify the volume group name (`oracle_vg`) and then follow that with the physical volume we just created (`/dev/sdd1`)

Example using `vgextend` command:

```
[root@localhost ~]# vgextend oracle_vg /dev/sdd1
  Volume group "oracle_vg" successfully extended
```

9. **Extend the logical group:**
   - Run `lvextend -L+1024M /dev/mapper/oracle_vg-oracle_lv`
     - Note that we have to specify the size, the option for this is `-L`
     - Note that we got the partition name (`dev/mapper/oracle_vg-oracle_lv`) by doing `df -h` (it has to be mounted)

Example using `lvextend` command:

```
[root@localhost ~]# lvextend -L+1024M /dev/mapper/oracle_vg-oracle_lv
  Size of logical volume oracle_vg/oracle_lv changed from 1000.00 MiB (250 extents) to <1.98 GiB (506
extents).
  Logical volume oracle_vg/oracle_lv successfully resized.
```

- The logical volume was successfuly extended

10. **Extend the filesystem:**
    - Run `xfs_growfs /dev/mapper/oracle_vg-oracle_lv`
      - Note that we got the partition name (`dev/mapper/oracle_vg-oracle_lv`) by doing `df -h` (it has to be mounted)

    - Note `xfs_growfs` stands for "grow filesystem"
- You can verify this action by looking at the new size of `dev/mapper/oracle_vg-oracle_lv` in the list made by `df -h`

Output from `df -h`:

```
Filesystem                      Size  Used Avail Use% Mounted on
devtmpfs                        4.0M     0  4.0M   0% /dev
tmpfs                           5.1G     0  5.1G   0% /dev/shm
tmpfs                           2.1G  9.2M  2.1G   1% /run
/dev/mapper/rhel-root            17G  7.2G  9.8G  43% /
/dev/sda1                       960M  411M  550M  43% /boot
/dev/sdb1                       2.0G   47M  1.9G   3% /data
/dev/mapper/oracle_vg-oracle_lv 936M   39M  898M   5% /oracle
tmpfs                           1.1G   52K  1.1G   1% /run/user/42
tmpfs                           1.1G   36K  1.1G   1% /run/user/1000
```

- Note the new size of the `dev/mapper/oracle_vg-oracle_lv` partition
    - It is now almost 2G which is the combination of 1G + 1G from our two disks.

Extending Swap on an LVM2 Logical Volume

from Extending Swap on an LVM2 Logical Volume

To extend an LVM2 swap logical volume (assuming `/dev/VolGroup00/LogVol01` is the volume you want to extend):

1. Disable swapping for the associated logical volume:

```
swapoff -v /dev/VolGroup00/LogVol01
```

2. Resize the LVM2 logical volume by 256 MB:

```
lvm lvresize /dev/VolGroup00/LogVol01 -L +256M
```

3. Format the new swap space:

```
mkswap /dev/VolGroup00/LogVol01
```

4. Enable the extended logical volume:

```
swapon -va
```

5. Test that the logical volume has been extended properly:

```
cat /proc/swaps
```