Table of Contents

F. Activate the Logical Volume
1. `swapon /dev/swap_vg/swap_lv`
G. Verify Swap Activation
1. `lsblk`
2. `free -m`
3. `cat /proc/swaps`

V. Disable Swap Space

A. `swapoff /dev/swap_vg/swap_lv`
B. Verify Status
1. `lsblk`
2. `free -m`
3. `cat /proc/swaps`

C. Re-activate Swap Space
1. `swapon -a`

1. Introduction to Logical Volume Management (LVM)

Logical Volume Management (LVM) provides a flexible and efficient way to manage disk space in Linux. Unlike traditional partitioning, LVM allows for the dynamic resizing of volumes, striping, mirroring, snapshots, and thin provisioning.

- **Key Concepts:**
  - **Physical Volume (PV):** A raw disk or partition prepared for LVM.
  - **Volume Group (VG):** A pool of storage from one or more PVs.
  - **Logical Volume (LV):** Units of storage carved out of VGs that can be used as a block device.

2. Key LVM Commands

2.1. Physical Volume (PV) Commands

1. pvcreate: Initializes a physical volume for LVM.
   - Example: pvcreate /dev/sdb1
2. pvdisplay: Displays attributes of a physical volume.
   - Example: pvdisplay /dev/sdb1
3. pvmove: Moves physical extents from one PV to another.
   - Example: pvmove /dev/sdb1 /dev/sdc1
4. pvremove: Removes a physical volume from LVM management.
   - Example: pvremove /dev/sdb1
5. pvresize: Resizes a physical volume.
   - Example: pvresize /dev/sdb1
6. pvs: Displays a summary of all physical volumes.
   - Example: pvs
7. pvscan: Scans all disks for physical volumes.
   - Example: pvscan

2.2. Volume Group (VG) Commands

1. vgcreate: Creates a volume group.
   - Example: vgcreate my_vg /dev/sdb1 /dev/sdc1
2. vgextend: Adds a PV to a VG.
   - Example: vgextend my_vg /dev/sdd1
3. vgreduce: Removes a PV from a VG.
   - Example: vgreduce my_vg /dev/sdd1
4. vgchange: Changes attributes of a VG.
   - Example: vgchange -a y my_vg
5. vgmerge: Merges two VGs.
   - Example: vgmerge destination_vg source_vg

6. vgsplit: Splits a VG into two smaller ones.
    - Example: vgsplit my_vg new_vg /dev/sdd1
7. vgscan: Scans for all VGs.
    - Example: vgscan
8. vgrename: Renames a VG.
    - Example: vgrename my_vg new_vg_name

2.3. Logical Volume (LV) Commands

1. lvcreate: Creates a new logical volume.
    - Example: lvcreate -L 10G -n my_lv my_vg
2. lvextend: Extends the size of a logical volume.
    - Example: lvextend -L +5G /dev/my_vg/my_lv
3. lvresize: Resizes a logical volume.
    - Example: lvresize --resizefs -L 20G /dev/my_vg/my_lv
4. lvremove: Removes a logical volume.
    - Example: lvremove /dev/my_vg/my_lv
5. lvconvert: Converts a logical volume (e.g., to a mirror or snapshot).
    - Example: lvconvert --snapshot --name my_snapshot /dev/my_vg/my_lv
6. lvs: Displays information about all logical volumes.
    - Example: lvs
7. lvscan: Scans for all logical volumes.
    - Example: lvscan

3. Advanced LVM Operations

3.1. Logical Volume Snapshot Feature and Restore

1. **Creating a Snapshot**
   Use the lvcreate command with -s and -n:
    - Example: lvcreate -L 1G -s -n snap_lv /dev/vg_name/lv_name
    - View snapshots: lvs
2. **Activating the Snapshot**
   Use lvchange to activate a snapshot volume:
    - Example: lvchange -ay /dev/vg_name/snap_lv
3. **Restoring from a Snapshot**
   Use lvconvert --merge to restore:
    - Example: lvconvert --merge /dev/vg_name/snap_lv

   Use Case Example:
   Before updating /dev/vg_data/db_lv, take a snapshot:

    - lvcreate -L 2G -s -n db_lv_snap /dev/vg_data/db_lv

   If needed, restore:

    - Activate snapshot: lvchange -ay /dev/vg_data/db_lv_snap
    - Merge snapshot: lvconvert --merge /dev/vg_data/db_lv_snap

3.2. Remove LVM in Linux

1. Remove all LVs:
    - lvremove /dev/my_vg/my_lv
2. Remove the VG:

    - vgremove my_vg
3. Remove the PV:
    - pvremove /dev/sdb1

3.3. Restoring a Deleted LVM from the Archive Folder

1. **Locate Archive Files**
   List archive files:
    - ls /etc/lvm/archive/
2. **Identify the Correct File**
   Inspect the file:
    - cat /etc/lvm/archive/vg_name_YYYY-MM-DD_HH:MM:SS.vg
3. **Test Restoration**
   Use vgcfgrestore --test to simulate:
    - vgcfgrestore --test -f /etc/lvm/archive/vg_name_YYYY-MM-DD_HH:MM:SS.vg vg_name
4. **Restore Volume Group Metadata**
   Proceed with actual restoration:
    - vgcfgrestore -f /etc/lvm/archive/vg_name_YYYY-MM-DD_HH:MM:SS.vg vg_name
5. **Scan for Logical Volumes**
    - lvscan
6. **Activate Volume Group**
    - vgchange -ay vg_name
7. **Verify Logical Volumes**
    - lvs

4. LVM in RHEL 8: Attributes and Details

4.1. Displaying LVM Attributes

1. **Display LV attributes**:
    - lvs -o lv_name,lv_size,lv_attr
2. **Display VG attributes**:
    - vgs -o vg_name,vg_size,vg_attr

4.2. Managing LVM Metadata

1. **Backup LVM Metadata**:
    - vgcfgbackup my_vg
2. **Restore LVM Metadata**:
    - vgcfgrestore my_vg

Step-by-Step Guide to Merge LVM Volume Groups Using vgmerge

Step 1: Verify Volume Groups

Start by listing all the available volume groups to identify the **source VG** (the one you want to merge) and the **target VG** (the one you want to merge into):

```
vgdisplay
```

This will give you the names and details of the volume groups on your system.

Step 2: Change the Status of the Source VG (if necessary)

If the source volume group is active, you may need to deactivate it before merging.
Use `vgchange` to deactivate it:

```
vgchange -a n <source_vg_name>
```

For example, if your source volume group is named `vg_source`:

```
vgchange -a n vg_source
```

Step 3: Merge Volume Groups

Now, use the `vgmerge` command to merge the **source VG** into the **target VG**.

```
vgmerge <target_vg_name> <source_vg_name>
```

For example, if the target VG is named `vg_target` and the source VG is named `vg_source`, you would run:

```
vgmerge vg_target vg_source
```

This command will move all physical volumes and logical volumes from `vg_source` into `vg_target`.

Step 4: Activate the Merged VG

After the merge, you'll need to reactivate the merged volume group:

```
vgchange -a y <target_vg_name>
```

For example:

```
vgchange -a y vg_target
```

Step 5: Verify the Merge

You can verify that the volume groups have been merged successfully by using the following commands:

```
vgdisplay
lvdisplay
pvs
```

These commands will show that all the physical volumes (PVs) and logical volumes (LVs) from the source VG are now part of the target VG.

Example Workflow

```
# List all volume groups
vgdisplay

# Deactivate the source VG
vgchange -a n vg_source

# Merge the source VG into the target VG
vgmerge vg_target vg_source

# Reactivate the merged VG
vgchange -a y vg_target

# Verify the merge
```

```
vgdisplay
lvdisplay
pvs
```

Step-by-Step Guide to Split a Volume Group Using vgsplit

Step 1: List Volume Groups and Physical Volumes

Before splitting, identify the existing volume group and the physical volumes you want to split. List all the volume groups and physical volumes using the following commands:

```
vgdisplay
pvs
```

This will show the current setup of your VGs and PVs.

Step 2: Prepare for Splitting

Decide which physical volume(s) and logical volume(s) you want to move to the new volume group. You can get details on which PVs and LVs belong to which VG using lvdisplay.

```
lvdisplay <volume_group_name>
```

Step 3: Split the Volume Group

To split a volume group, use the vgsplit command. You will specify the source VG, the new VG name, and the physical volume(s) to move into the new VG.

```
vgsplit <source_vg_name> <new_vg_name> <physical_volume>
```

For example, if you want to split the vg_main volume group, moving /dev/sdb1 to a new VG called vg_new, you would run:

```
vgsplit vg_main vg_new /dev/sdb1
```

This command creates a new volume group named vg_new with the specified physical volume /dev/sdb1 while keeping the remaining physical volumes in vg_main.

Step 4: Verify the Split

After splitting, you can verify that the new VG has been created and that the physical volumes have been moved by running:

```
vgdisplay
pvs
lvdisplay <new_vg_name>
```

These commands will display the details of the split volume groups and their associated physical and logical volumes.

Step 5: Activate the New Volume Group (if needed)

If the new volume group is inactive, you can activate it using vgchange:

```
vgchange -a y <new_vg_name>
```

For example:

```
vgchange -a y vg_new
```

Example Workflow

```
# List volume groups and physical volumes
vgdisplay
pvs

# Split the volume group, moving a physical volume to the new group
vgsplit vg_main vg_new /dev/sdb1

# Verify the new volume group and physical volumes
vgdisplay
pvs
lvdisplay vg_new

# Activate the new volume group if needed
vgchange -a y vg_new
```

Why Logical Volume Migration is Required?

- **Replace a faulty disk**
- **Upgrade to a larger disk**

Steps to Migrate Logical Volumes in RHEL

1. Check the Existing Setup

Verify the current LVM configuration:

```
# df -hP /data/
# pvs
# vgs
# lvs
```

Check which physical device is used by the logical volume:

```
# lvs -o +devices /dev/vg01_data/lv_data
```

2. Add a New Physical Volume

Create a physical volume on the new disk and extend the volume group:

```
# pvcreate /dev/sdc1
# vgextend vg01_data /dev/sdc1
```

3. Migrate the Logical Volume

Move data from the old disk to the new one using `pvmove`:

```
# pvmove -n /dev/vg01_data/lv_data /dev/sdb1 /dev/sdc1
```

Monitor progress:

```
# watch -n 0.5 lvs -a -o +devices
```

4. Remove the Old Physical Volume

Once the migration is complete, remove the old disk from the volume group and clean up:

```
# vgreduce vg01_data /dev/sdb1
# pvremove /dev/sdb1
```

5. Verify the Changes

Check the new setup:

```
# pvs
# lvs -o +devices /dev/vg01_data/lv_data
```

This approach makes it easy to replace a disk while keeping your system online.

Migrating an LVM data disk from one machine to another in **RHEL 8** involves a series of steps to ensure that the logical volumes, volume groups, and physical volumes are properly moved and recognized on the target system. Below is a step-by-step guide:

Steps to Migrate LVM Data Disk from One Machine to Another

1. Identify the LVM Structure on the Source Machine

Before beginning the migration, gather information about the LVM structure on the source machine.

- Use the following commands to list the current LVM configuration:

```
# List all volume groups
vgs

# List all logical volumes
lvs

# List all physical volumes
pvs
```

2. Prepare the Disk for Migration

If the LVM data disk is on a physical disk that you plan to move to the new machine, ensure that it's not mounted and properly prepared for migration.

- **Unmount any filesystems** on the logical volumes:

```
umount /dev/<volume_group>/<logical_volume>
```

- **Deactivate the volume group** (optional but recommended):

```
vgchange -an <volume_group>
```

3. Export the Volume Group (Optional)

While not strictly necessary, exporting the volume group ensures that the system on the source machine won't accidentally activate the volume group after migration.

```
vgexport <volume_group>
```

This makes the volume group inactive and unimportable on the source machine, but it's safe for import on the target machine.

4. Physically Transfer the Disk

- If you're using a **physical disk**, simply disconnect it from the source machine and connect it to the target machine.
- If you're using a **virtual disk**, copy the disk image or attach the virtual disk to the target machine using your virtualization platform (like VMware, KVM, etc.).

5. Scan for the New Disk on the Target Machine

Once the disk is physically or virtually attached to the target machine, scan for it to ensure it's recognized.

- If it's a physical disk, you can force the kernel to rescan the SCSI bus:
- `echo "- - -" > /sys/class/scsi_host/hostX/scan`
  (Replace `hostX` with the correct host number for your SCSI controller.)

- Check for the new disk:
- `fdisk -l`

6. Import the Volume Group

On the target machine, import the volume group so that it becomes available.

- First, scan for the physical volumes:
- `pvscan`
- Then, import the volume group:
- `vgimport <volume_group>`
- Activate the volume group:
- `vgchange -ay <volume_group>`

7. Mount the Logical Volumes

Once the volume group is active, you can mount the logical volumes to access the data.

- List the logical volumes to verify:
- `lvs`
- Mount the desired logical volumes:
- `mount /dev/<volume_group>/<logical_volume> /mnt`

8. Optional: Persistent Mounting

If you want the logical volumes to be automatically mounted at boot time, add entries to `/etc/fstab`:

```
/dev/<volume_group>/<logical_volume> /mnt ext4 defaults 0 0
```

9. Verify Data Integrity

After mounting the logical volumes, check that the data is accessible and the files are intact.

```
ls /mnt
```

Example Scenario

1. On the source machine:
   - You have a volume group named `vg_data` and a logical volume named `lv_backup`.
   - Unmount the logical volume and deactivate the volume group.
2. `umount /dev/vg_data/lv_backup`
3. `vgchange -an vg_data`
4. Physically move the disk to the target machine.
5. On the target machine:

- Scan for the new disk, import the volume group, and activate it:

```
6. pvscan
7. vgimport vg_data
8. vgchange -ay vg_data
```

9. Mount the logical volume on the target machine:

```
10. mount /dev/vg_data/lv_backup /mnt
```

This method ensures that your LVM data disk is safely migrated to the new machine without data loss.

LVM Thin Provisioning Setup

1. List all block devices to check current available disks

```
lsblk  # Identify available disks and partitions
```

2. Start fdisk to create a new partition on /dev/sdb

```
fdisk /dev/sdb <<EOF
n     # New partition
p     # Primary partition
1     # Partition number 1
      # Default first sector
+20G  # Set partition size to 20GB
w     # Write changes and exit
EOF
```

3. Update the kernel partition table

```
partprobe  # Notify the OS of partition table changes
```

4. Verify the partition creation

```
lsblk  # Confirm the new partition /dev/sdb1
```

5. Initialize the new partition as a physical volume for LVM

```
pvcreate /dev/sdb1  # Prepare /dev/sdb1 for LVM
```

6. Create a volume group with a custom extent size

```
vgcreate -s 32M vg_thin /dev/sdb1  # Create a volume group
```

7. Verify the volume group creation

```
vgs
vgdisplay vg_thin
```

8. Create a thin pool in the volume group

```
lvcreate -L 15G --thinpool thin_pool vg_thin  # Create thin pool
```

9. Verify the thin pool creation

```
lvs
lsblk
```

10. Create thin-provisioned logical volumes

```
lvcreate -V 5G --thin -n thin_vol_client1 vg_thin/thin_pool
lvcreate -V 5G --thin -n thin_vol_client2 vg_thin/thin_pool
lvcreate -V 5G --thin -n thin_vol_client3 vg_thin/thin_pool
```

11. Confirm thin volume creation

```
lvs  # Verify thin volumes
```

12. Create mount points

```
mkdir -p /mnt/client1 /mnt/client2 /mnt/client3
```

13. Format the thin-provisioned volumes

```
mkfs.ext4 /dev/vg_thin/thin_vol_client1
mkfs.ext4 /dev/vg_thin/thin_vol_client2
mkfs.ext4 /dev/vg_thin/thin_vol_client3
```

14. Mount the volumes

```
mount /dev/vg_thin/thin_vol_client1 /mnt/client1
mount /dev/vg_thin/thin_vol_client2 /mnt/client2
mount /dev/vg_thin/thin_vol_client3 /mnt/client3
```

15. Check mounted filesystems

```
df -h  # Check mounted thin volumes
```

16. Create 1GB test files in each volume

```
fallocate -l 1G /mnt/client1/test.img
fallocate -l 1G /mnt/client2/test.img
fallocate -l 1G /mnt/client3/test.img
```

17. Check thin pool usage

```
lvs  # Monitor thin pool usage
```

18. Create an additional thin-provisioned volume

```
lvcreate -V 5G --thin -n thin_vol_client4 vg_thin/thin_pool
```

19. Extend the thin pool

```
lvextend -L +15G /dev/vg_thin/thin_pool
lsblk  # Verify thin pool extension
```

20. Create another partition for further expansion

```
fdisk /dev/sdb <<EOF
n     # New partition
p     # Primary partition
2     # Partition number 2
      # Default first and last sector
t     # Change partition type
8e    # Set type to Linux LVM
w     # Write changes and exit
EOF
```

21. Notify the OS of partition changes

```
partprobe
```

22. Initialize the new partition as a physical volume

```
pvcreate /dev/sdb2
```

23. Verify physical volumes

```
pvs  # Confirm /dev/sdb2 creation
```

24. Extend the volume group

```
vgextend vg_thin /dev/sdb2
```

25. Extend the thin pool further

```
lvextend -L +15G /dev/vg_thin/thin_pool
```

26. Verify volume group and logical volumes

```
vgs
lvs  # Confi
```

Chapter 9: Create & Manage Swap Space

9.1. Understanding Swap Space

Swap space is a designated area on disk that serves as an overflow for your system's RAM. When your system runs out of physical memory, it uses swap space to keep applications running. Properly managing swap space is crucial for system performance, especially in environments with limited RAM.

Benefits of Swap Space:

- **Memory Management**: Provides extra virtual memory for processes when RAM is full.
- **System Stability**: Helps prevent out-of-memory errors and system crashes.
- **Hibernation**: Allows the system to save the current state to disk during hibernation.

9.2. Creating Swap Space on Red Hat

1. Check OS Release Information

To ensure you are on the correct machine, check the OS version:

```
cat /etc/os-release
```

2. List Block Devices

Identify available disks and their partitions:

```
lsblk
```

3. Create Swap Space

a. Create a New Partition

1. Open `fdisk` for the desired disk (e.g., `/dev/sda`):

```
fdisk /dev/sda
```

3. Create a new partition:
    - Type `n` for a new partition.
    - Type `p` for a primary partition.
    - Press `Enter` to accept the default partition number.
    - Press `Enter` to accept the default first sector.
    - Specify the size (e.g., `+2G` for a 2GB swap partition).
4. Change the partition type:
    - Type `t` to change the partition type.
    - Enter `82` to set the type to Linux swap.
5. Write the changes:
    - Type `w` to write changes and exit.

b. Verify Partition Creation

Check the new partition:

```
lsblk
```

c. Notify the OS of Partition Changes

```
partprobe
```

d. Set Up the Swap Space

1. Create the swap area:

```
mkswap /dev/sda1
```

3. Verify the setup:

```
lsblk -f
free -m
cat /proc/swaps
```

7. Enable the swap space:

```
swapon /dev/sda1
```

9. Confirm it's active:

```
free -m
top
cat /proc/swaps
```

4. Make Swap Space Permanent

Edit the `/etc/fstab` file to make the swap space persistent across reboots:

```
vim /etc/fstab
```

Add the following line (replace `UUID=` with the actual UUID of the swap partition):

```
UUID=YOUR_SWAP_UUID swap swap defaults 0 0
```

To find the UUID, you can run:

```
blkid /dev/sda1
```

5. Check Free Memory

```
free -g
free -m
```

9.3. Creating Swap Space with LVM on CentOS

1. Log into the CentOS machine and check the block devices:

```
lsblk
```

3. Open `fdisk` for the desired disk:

```
fdisk /dev/sda
```

5. Create a new LVM partition:
    - Type `n` for a new partition.
    - Type `p` for a primary partition.
    - Press `Enter` to accept the default partition number.
    - Press `Enter` to accept the default first sector.
    - Specify the size for the swap partition (e.g., `+2G`).
6. Change the partition type to LVM:
    - Type `t` and select the partition.
    - Enter `8e` for Linux LVM.
7. Write the changes and exit:

```
8. w
```

1. Initialize Physical Volume for LVM

```
pvcreate /dev/sda1
```

2. Create a Volume Group

```
vgcreate swap_vg /dev/sda1
```

3. Create a Logical Volume for Swap

```
lvcreate -L 5000M -n swap_lv swap_vg
```

4. Set Up the Swap Space

```
mkswap /dev/swap_vg/swap_lv
swapon /dev/swap_vg/swap_lv
```

5. Verify the Swap Space

```
lsblk
free -m
cat /proc/swaps
```

6. Make LVM Swap Permanent

Edit `/etc/fstab` to include the new swap space:

```
vim /etc/fstab
```

Add the following line:

```
/dev/swap_vg/swap_lv swap swap defaults 0 0
```

7. Check the Swap Space Again

```
free -m
cat /proc/swaps
```

8. Turn Off Swap (if needed)

To disable the swap space:

```
swapoff /dev/swap_vg/swap_lv
```

Then check:

```
lsblk
free -m
cat /proc/swaps
```

9. Re-enable Swap

```
swapon -a
```