**Storage Management**

1. [List, create, delete, and modify physical storage partitions](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/StorageManagement.md#list-create-delete-and-modify-physical-storage-partitions)
2. [Manage and configure LVM storage](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/StorageManagement.md#manage-and-configure-lvm-storage)
3. [Create and configure encrypted storage](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/StorageManagement.md#create-and-configure-encrypted-storage)
4. [Configure systems to mount file systems at or during boot](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/StorageManagement.md#configure-systems-to-mount-file-systems-at-or-during-boot)
5. [Configure and manage swap space](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/StorageManagement.md#configure-and-manage-swap-space)
6. [Create and manage RAID devices](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/StorageManagement.md#create-and-manage-raid-devices)
7. [Configure systems to mount file systems on demand](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/StorageManagement.md#configure-systems-to-mount-file-systems-on-demand)
8. [Create, manage and diagnose advanced file system permissions](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/StorageManagement.md#create-manage-and-diagnose-advanced-file-system-permissions)
9. [Setup user and group disk quotas for filesystems](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/StorageManagement.md#setup-user-and-group-disk-quotas-for-filesystems)
10. [Create and configure file systems](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/StorageManagement.md#create-and-configure-file-systems)

**List, create, delete, and modify physical storage partitions**

* lsblk -> lists all available disk devices plus available partitions.
  + Use -f to show filesystem type.
* parted -> is another tool doing the same. Used for scripting.
* fdisk -> it is used to manage disk partition in MBR modality.
  + E.g. fdisk /dev/sda

This will open an interactive menu that will permit to show current status of partitions or create a delete new partitions.

* gdisk -> it is used to manage disk partition in GPT modality.
  + E.g. gdisk /dev/sda
* Destroy all MBR partition on a disk.
  + gdisk /dev/sda -> x (expert) -> z (zap)
* Convert MBR to GPT.
  + gdisk /dev/sda -> W -> Y
  + GPT held a copy of the partition table at the end of the disk.

**Manage and configure LVM storage**

* Before creating a Logical Volume you first need to created a physical volume and after that a volume group.
* A physical volume is a partition that can be part of volume group. Inside a volume group can be created logical volume.
* The advantage of logical volume is that their size can be managed easly.
* If more space is needed a volume group can be extended as well.
* lvmdiskscan - can be used to view on which partitions are LVM enabled.

Physical Volume

* pvcreate /dev/sdb1 -> To create a physical volume with partition sbd1.
* pvs -> Lists available physical volumes.
* pvdisplay /dev/sdb1 -> Shows info of a physical volume.
* pvmove /dev/sdb1 /dev/sdc10 -> Move the Volume to new disk.

Volume Group

* vgcreate vgname /dev/sdb1 -> To create a volume group called *vgname* and add the sdb1 physical volume to it.
* vgs -> Lists available volume groups.
* vgdisplay vgname -> Shows info of a volume group.
* vgextend vgname /dev/sdc3 -> Extends a volume group adding a new physical volume /dev/sdc3
* vgreduce vgname /dev/sdc3 -> Remove physical volume from a volume group.

Logical volume

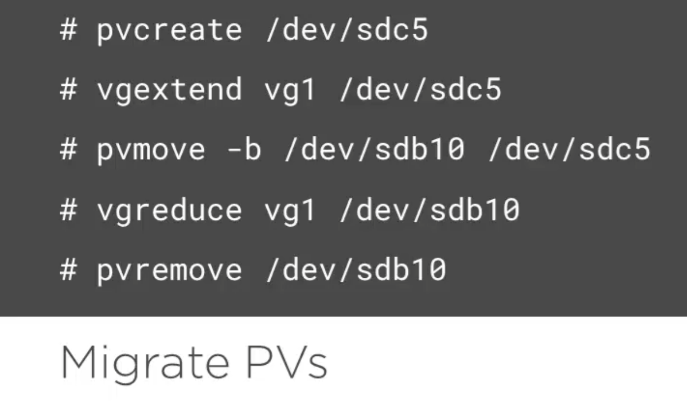
* lvcreate -n volumename -L 10G vgname

To create a logical volume called *volumename* of size 10GB on volume group *vgname*.

* lvcreate -n volumename -l 100%FREE vgname

To create a logical volume called *volumename* with all available space on volume group *vgname*.

* lvs -> List available logical volumes.
* lvdisplay -> Shows info of all logical volumes.
* lvdisplay vgname/volumename -> Shows info of a logical volume *volumename* contained in *vgname* volume group.
* Before using a logical volume, file system must be created on it with: mkfs -t ext4 /dev/vgname/volumename
* Add entry to /etc/fstab
  + blkid /dev/vgname/volumename -> shows the UUID of a formatted volume group.
  + You can use: /dev/mapper/vgname-volumename or /dev/vgname/volumename it's the same.
* lvextend -L +1G -r vgname/volumename  -> Extends the logical volume *volumename* of one giga and resize.
  + -r is used to resize file system.
  + For xfs use xfs\_growfs /lvm to resize the filesystem.
  + To resize ExtN use -r or resize2fs /dev/vgname/volumename.
  + The size of xfs filesystem cannot be reduced!
* lvextend -l +100%FREE -r /dev/mapper/vgname-volumename - It is important to include the minus (-) or plus (+) signs while resizing a logical volume. Otherwise, you’re setting a fixed size for the LV instead of resizing it.
* lvreduce -L -1G -r vgname/volumename  -> Reduce the logical volume *volumename* of one giga and resize.
* sudo lvresize -L +5G --resizefs vgname/volumename -> You can use resize command also. -r == --resizefs
* lvcreate -L 30M -s -n backup /dev/vgname/volumename -> Create LVM Snapshot.
  + mount /dev/vgname/volumename/backup /mnt -o nouuid,ro -> Mount the backup somewhere.
* Migrate PVs:

[](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/Migrate_PVs.png)

**Create and configure encrypted storage**

* To use encrypted storage a kernel module must be loaded
  + sudo modprobe dm\_crypt -> Loads kernel module dm\_crypt.
  + echo dm\_crypt >> /etc/modules-load.d/dm\_crypt.conf -> Load dm\_crypt module automatically when system will be restarted.
  + lsmod -> Lists all loaded kernel modules.
* yum -y install cryptsetup -> Install software used to manage encrypted storage.
* shred -v /dev/vg1/lv1 -> Overwrite a file to hide its contents.

Encrypt

* cryptsetup luksFormat /dev/vgname/volumename -> Encrypts a logical volume *volumename* contained in *vgname* volume group.
  + A password must be provided.
  + When confirmation will be required insert a capital YES.
  + **NOTE**: this command can be used with physical volume as well.
* cryptsetup luksDump /dev/vgname/volumename -> Show header information of a LUKS device.
* cryptsetup luksOpen /dev/vgname/volumename namenewdevice

It open encrypted volume and associate it to a new device called *namenewdevice*

* + Password must be provided
* mkfs.ext4 /dev/mapper/namenewdevice -> It creates a file system in *namenewdevice*.
  + Now the new device can be mounted.

Close device

* Unmount device
* cryptsetup close namenewdevice -> close *namenewdevice*

Automount

* echo "passwd" >> /root/key -> Insert a string that will be used as authentication key to open device.
* chmod 400 /root/key -> Reduces permission on key file.
* cryptsetup luksAddKey /dev/mapper/namenewdevice /root/key -> Add key to encrypted device called *namenewdevice*
* Edit /etc/crypttab and add below row:
  + namenewdevice /dev/vgname/volumename /root/key
* Add below row to /etc/fstab
  + /dev/mapper/namenewdevice /mnt/mountpoint ext4 defaults 0 0
* Reboot system or reload system manager
  + systemctl daemon-reload
* The new encrypted volume will be mounted on /mnt/mountpoint

**Configure systems to mount file systems at or during boot**

* Edit /etc/fstab adding a row similar to:
  + /dev/sdb1 /mnt/mountpoint ext4 defaults 0 0
    - Mount device sdb1 to mountpoint.
    - Device is formatted using ext4 filesystem.
    - Default mount options are used
    - 0 0 -> Dump (bkp) and fsck.
      * First 0 means no backup required
      * Second 0 means no fsck required in case of not correct umount. To enable fsck insert 2 because number indicate the check order, and 1 is given to operating system disk and two do data disks.
* mount -> Shows mounted volumes.
* mount -a -> Reloads /etc/fstab.
* mount -t type -o options device dir
  + It mounts a *device* formatted with file system *type* on directory *dir* using a list of options.
  + Mount options can be found in man mount. There are general options for all FS and specific to FS. List of general mount options:
    - async -> I/O asincrono
    - auto -> Can be mounted using mount -a
    - default -> Equal to this list of options: async,auto,dev,exec,nouser,rw,suid
    - loop -> To mount an ISO image
    - noexec -> no exec
    - nouser -> A user cannot mount this volume
    - remount -> Mount volume also if it is already mounted
    - ro -> Read only
    - rw -> Read an write
    - relatime -> Modify file access time (atime) if file is changed or one time a day. Alternative, to reduce disk traffic, noatime can be used. This is useful with SSD to avoid not useful write.

***SMB protocol***

* yum -y install samba samba-client cifs-utils -> it installs software need to manage CIFS/SMB protocol
* Samba configuration file:  /etc/samba/smb.conf
* smbclient -L targetIP

It lists all SMB shared directory available on a target IP

* + root password must be provided
* mount -t cifs -o username=smbuser,password=1234pwd //192.168.0.10/share /media/samba

It mounts a directory *share*, shared by server 192.168.0.10 on samba directory. User and password to authentication are provided

* Permanent configuration
  + echo "username=smbuser" >> /media/smb/.smbconf # can be any file name, just change the perm.
  + echo "password=1234pwd" >> /media/smb/.smbconf
  + chmod 600 /media/smb/.smbconf
  + In /etc/fstab insert:
    - //192.168.0.10/share /media/samba cifs credentials=/media/samba/.smbcredentials,defaults 0 0

***NFS protocol***

* yum -y install nfs-utils -> it install software to manage NFS protocol.
  1. systemctl enable nfs-server --now
  2. Shared files are configure in /etc/exports like this:
  3. /exports\_nfs/docs 192.168.233.0/24(ro,no\_subtree\_check)

# If no\_root\_squash option is used, remote root users are able to change any file on the shared file system

* 1. Run the following: exportfs -rv
* showmount -e targetIP

It lists all NFS shared directory available on a target IP

* mount -t nfs -o defaults 192.168.0.10:/srv/nfs /media/nfs

It mounts a directory *nfs*, shared by server 192.168.0.10 on nfs directory

* Permanent configuration
  1. In /etc/fstab insert:
     + 192.168.0.10:/srv/nfs /media/nfs nfs defaults 0 0
  2. To user NFSv3 insert:
     + 192.168.0.10:/srv/nfs /media/nfs nfs defaults,vers=3 0 0

Example: NFS create nfs share and give it ro insecure access to network xx.xx.xx.xx & give rw secure access to example.com

* 1. /nfs/share xx.xx.xx.xx(ro,insecure) example.com(rw,secure)

For more info: man exports

* **Note** Use Samba in a mixed environment and NFS whenever cross-platform compatibility is not necessary.

**Configure and manage swap space**

* To use a device as swap space:
  + mkswap /dev/sdb3
  + swapon -v /deb/sdb3
  + In /etc/fstab insert:
    - /dev/sdb3 swap swap defaults 0 0
  + swapon --show -> Display swap usage summary or: cat /proc/swaps
  + swapoff /dev/sdb3 -> Disable the swap.
  + swapon -p 10 /dev/sd3 -> Starts the swap and give it priority of 10. The higher priority is used first. Can be configured directly in */etc/fstab* replacing *defaults* with *sw,pri=10*
* You can also use file as swap space. The commands are the same.

**Create and manage RAID devices**

Concepts:

* Parity disk. It is used to provide fault tolerance.
* The spare device. It doesn't take part of the RAID(Redundant Array of Independent Disks) and it is used only in case of a disk fault. In this case spare enter in the RAID and the content of lost disk is reconstructed and saved on it.
* yum -y install mdadm -> Installs software to manage RAID devices.
* RAID 0 - Striped - No spare
  + mdadm --create --verbose /dev/md0 --level=stripe --raid-devices=2 /dev/sdb1 /dev/sdc1
* RAID 1 - Mirror
  + mdadm --create --verbose /dev/md0 --level=1 --raid-devices=2 /dev/sdb1 /dev/sdc1
* RAID 5 - (1 parity + 1 spare)
  + mdadm --create --verbose /dev/md0 --level=5 --raid-devices=3 /dev/sdb1 /dev/sdc1 /dev/sdd1 --spare-devices=1 /dev/sde1
* RAID 6 - (2 parity + 1 spare)
  + mdadm --create --verbose /dev/md0 --level=6 --raid-devices=4 /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde --spare-devices=1 /dev/sdf1
* RAID 10 - (Stripe + Mirror + 1 spare)
  + mdadm --create --verbose /dev/md0 --level=10 --raid-devices=4 /dev/sd[b-e]1 --spare-devices=1 /dev/sdf1
* mdadm --detail /dev/md0 -> Shows status of RAID device.
* To use device md0, format it and use as a classical device.
* cat /proc/mdstat -> Give general info.

Monitoring RAID devices:

* mdadm --assemble --scan
* mdadm --detail --scan >> /etc/mdadm.conf -> Write configuration to the configuration file. This file helps to reinitialize RAID device after reboot.
* echo "MAILADDR root" >> /etc/mdadm.conf
* systemctl start mdmonitor
* systemctl enable mdmonitor

***Add disk:***

* mdadm /dev/md0 --add /dev/sbc2
* mdadm --grow --raid-devices=4 /dev/md0 - It adds a spare disk and after it grows array.

***Remove disk:***

* mdadm /dev/md0 --fail /dev/sdc1 --remove /dev/sdc1

mdadm --grow /dev/md0 --raid-devices=2

It mark disk as failed and remove it. After that the size of array must be adjusted.

***Delete RAID:***

* Unmount device
* mdadm --stop /dev/md0
* mdadm --zero-superblock /dev/sbc2 -> It cleans the partition that can be resused later.

References:

* <https://raid.wiki.kernel.org/index.php/A_guide_to_mdadm>

**Configure systems to mount file systems on demand**

* yum -y install autofs installs software need to manage automount.

Automount NFS directory

* Edit /etc/auto.master and insert:
  + /misc /etc/auto.misc
* Edit /etc/auto.misc and insert:
  + nfs -fstype=nfs 192.168.0.10:/srv/nfs
  + sten-automount -fstype=ext4 :/dev/sdb1
* systemctl start autofs

**Create, manage and diagnose advanced file system permissions**

**ACL - Access control list**

* They must be supported by the filesystem or build-in into the kernel. Check the kernel with the follwoing: grep ACL /boot/config-$(uname -r)
* With some old filesystem a mount option (e.g. *acl*) must be provided to enable ACL.
* Check the *default* mount option for ext4(/dev/sdb1) -> tune2fs -l /dev/sdb1 | grep -i default
* getfacl file -> shows ACL applied to a file.
* setfacl -R -m g:sales:rx file -> set ACL on file.
  + -R -> recursive, if file is a directory, ACL will be applied to all file inside it.
  + -m -> modify
  + g:sales:rx -> group sales can read and execute
    - g group
    - u user
    - o other
* setfacl -m u:dummy:- file -> remove all permissions of user dummy.
* setfacl -m d:g:sales:rx directory -> Set a default ACL to a directory. In this way all files created inside it will have same ACL as default.

The default ACL is a specific type of permission assigned to a directory, that doesn’t change the permissions of the directory itself, but makes so that specified ACLs are set by default on all the files created inside of it.

* If an ACL is applied, when ls -la is executed an + is inserted after other permissions. The "." in the end shows that ACL is supported.
* setfacl -x u:test:w test -> remove ACL.
* setfacl -x u:test: test -> Or try this way.
* setfacl -b file -> removes all ACL.

**Extended attributes**

* Extended attributes are file properties.
* With some old filesystem a mount option (e.g. *user\_xattr*) must be provided to enable extended attributes.
* Only root user can remove an attribute!
* chattr +i file -> Add *immutable* attribute to a file. It cannot be deleted or removed.
* chattr -i file -> Remove *immutable* attribute from a file.
* chattr +a file -> The file can only be opened in append mode for writing.
* chattr +A file -> When a file with this attribute set is open, its atime(last time the file was accessed/opened) record is not changed.
* lsattr file -> shows file's extended attributes.
* To find more attribute have a look here: man chattr

**Setup user and group disk quotas for filesystems**

***ext quota:***

* **Quota**: space that can be used by an user on one specific filesystem.
  + NOTE: To limit space in a directory it is better create a specific mount point with a specific partition.
* yum -y install quota -> installs software need to manage quota.
* ***usrquota,grpquota*** mount options must be inserted for filesystem to which enable quota (e.g. editing /etc/fstab)
* After that options are inserted, remount partition to enable them.
* After remount execute quotacheck -mavug that check used blocks and inserted them in a tracking file.
  + Two files will be created:
    - aquota.group
    - aquota.user
* quotaon -a -> Start quota system.
  + Alternative:
  + quotaon -vu /mnt/mountpoint -> It starts only quota user for specific mountpoint.
  + quotaon -vg /mnt/mountpoint -> It starts only quota group for specific mountpoint.
* quota -vu user -> shows user's quota.
* The quota is specified in blocks of 1K size and in number of inode that is the number of files that can be created.
* Hard limit: maxim value allowed.
* Soft limit: a limit that can be exceeded for a *grace period*. Default *grace period* is one week.
* When grace period is reached, soft limit become and hard limit.
* setquota -u hriska 20000 25000 0 0 /dev/sdd2 - Set block limit quota for user hriska.
* edquota -t -> Edit the grace period. Is an unique value for the whole system.
* edquota -u user -> Edit user's quota.
  + In each column can be insert a value for soft and hard limit for blocks and inode.
  + **NOTE**: Normally soft and hard limits are configured equal to avoid confusion.
* repquota -aug -> It shows an overview of current quota for each users.
* Soft limit=900 and the hard limit=1000 blocks (**1024 bytes/block \* 1000 blocks = 1024000 bytes = 1 MB**) of disk space usage.

***xfs quotas:***

* Valid quota mount options are:
  + uquota/uqnoenforce: User quotas
  + gquota/gqnoenforce: Group quotas
  + pquota/pqnoenforce: Project quota
* xfs\_quota
  + -c is command -x is expert mode.
  + xfs\_quota -c "quota delme1" -> Show quota for user delme1.
  + xfs\_quota -xc "report -h" /mnt/quota -> Enter expert mode and show quota for /mnt/quota.
  + xfs\_quota -xc "limit -u bsoft=30m bhard=40m delme1" /mnt/quota -> Set block quota for user delme1.

**Create and configure file systems**

* mkfs.ext4 /dev/sdb1 -> Creates an filesystem ext4 on sdb1 partition.
* mkfs -t ext4 -L DATA /dev/sdb5 -> Creates the filesystem and label it.
* tune2fs -L MY\_DATA /dev/sdb5 -> Change the label of the *ext* filesystem. Can also be used to change other params like: filesystem\_check count and count, etc.
* dumpe2fs /dev/sdb5 -> Give information for *ext* filesystem.
* fsck.ext4 /dev/sdb1 -> Checks the integrity of sdb1 filesystem.
* mount -o remount,noexec /dev/sdb1 /data/ext4 -> Add new mount option - *noexec*.
* umount /data/ext4 -> Unmount the partition.
* For ***xfs*** use the following command. A lot of more options. Used in Enterprises:
  + mkfs.xfs -b size=1k -l size=10m /dev/sdb3
  + xfs\_db -x /dev/sdb3 -> Open interactive menu.
  + xfs\_admin -L STEN\_XFS\_ww /dev/sten/mirrored -> Change label. xfs\_admin uses the xfs\_db(8) command to modify various parameters of a filesystem. Nice tool!