Linux Commands

General Commands

to add numbered lines in the vim editor use **set number**, edit the .vimrc file and add that line

ls -al /bin/sh → to see what shell your system use

visudo → vi editor used to edit the sudoers file, it checks for errors

sudo !! → if you forget to do sudo for the previous command

tail → to see the last 10 lines of a file

less → to read files (scrolling)

more → to read files (displays one page)

which → locate the executable file of a command

whereis → locate the binary, source, and manual page files for a command

type → to see the type of a command

who → to see who is currently logged in

stat <file name> → display information about a file

uniq → to remove any duplicates

uptime → how long the system has been online

man -k <keyword> → find a command by a keyword

Managing Users and Groups

getent : to retrieve information from ***/etc/passwd***, ***/etc/shadow*** and ***/etc/group***

```
Syntax : getent database [key]

Example: getent passwd Mike
```

passwd : to set a password to a user

```
passwd [username]
```

gpasswd : to administer group related stuff (use -a to add a user to a group)

```
gpasswd [option] group
```

```
-a user: Add a user to a group.
-d user: Remove a user from a group.
```

useradd : to add a user (to see the defaults use -D) *Note: when you create a user a copy of /etc/skel will be placed in the directory of the new user*

```
useradd [options] username
```

userdel : to delete a user

```
userdel [options] username

-r: Remove the user's home directory and mail spool.
```

usermod : to modify a user

```
usermod [options] username

-l new_name: Change the username.
-s /path/to/shell: Change the user's login shell.

sudo usermod -l new_username old_username

sudo usermod -s /bin/bash username
```

groupadd : to create groups

```
groupadd groupname
```

groupmod : to modify a group (use -n to change the name)

```
sudo groupmod -n new_groupname old_groupname
```

groupdel : to delete a group

```
sudo groupdel groupname
```

chage : to set password expiration date and policies

```
chage [options] username

-l: List password aging information for a user.

#Set a password expiration date:-

sudo chage -E 2024-12-31 username
```

groups : to see the groups that a certain user belongs to.

id : displays UID, GID and groups of a certain user.

chsh : change a user's default shell. *Note: you can also use **usermod -s** to change the shell*

```
sudo chsh -s /bin/zsh username
```

File Access & Permissions

Umask values

- **0** : read, write and execute
- **1** : read and write
- **2** : read and execute
- **3** : read only
- **4** : write and execute
- **5** : write only
- **6** : execute only
- **7** : no permissions

Read=4, write=2, execute=1

Plus and Minus signs for modifying, while Equal sign for overwriting. Ex. Chmod u=rw file.txt

chmod → to change permissions

```
chmod 660 file.txt
chmod u+rw,g+rw,o-rwx file.txt
```

chown → change file owner and group

```
sudo chown username:groupname file.txt
```

chgrp → used to change group ownership

```
chgrp [options] group file
```

newgrp → used to change your primary group

```
newgrp groupname
```

advanced permissions

getfacl → get file access control lists

```
getfacl file
```

setfacl → set file access control lists

```
setfacl -m u:username:rw file.txt
setfacl -s d:u:username:rw,g:groupname:r dir

-s: Overwrite the current ACL.
-m: Modify the existing ACL
```

Disk Partitions and File Systems

lsblk → list block devices

blkid → print block devices attributes

mkswap : create linux swap area

```
sudo mkswap /dev/sda3
```

swapon/swapoff : to enable/disable a swap area to be used

```
sudo swapon /dev/sda3
sudo swapoff /dev/sda3
```

mount : to mount a filesystem to location *Note: if you want them to be mounted automatically when the system boots, you will need to add them to the* ***/etc/fstab***

umount : to unmount a filesystem

Mounting with systemd

- **Create a file in /etc/system/system:**

```
The file name should match the path of the mount point. For example,
if the mount point is /mnt/storage, the file should be named mnt-
storage.mount
```

- **File structure:**

```
The following should be added to the file:


#general information about the unit
[Unit]
Description=Backup mount point


#information used mounting
```

```
[Mount]
What=/dev/sda3 #its better to use the UUID (/dev/disk/by-uuid/<UUID>)
Where=/mnt/Storage1
Type=ext4
Options=defaults

[Install]
WantedBy=multi-user.target #when it should be mounted.
```

- **Tell system to reread unit files:**

```
Systemctl daemon-reload
```

- **Start and check the status of the mount:**

```
Systemctl start mnt-storage.mount

Systemctl status mnt-storage.mount
```

- **Automatic mount at boot time:**

```
Systemctl enable mnt-storage.mount
```

Choose and Create a Partition Table

fdisk : used to create MBR Partition style

```
fdisk [options] device
```

gdisk : used to create GPT partition style

```
gdisk [options] device
```

parted : used for both MBR and GPT

```
parted [options] device
```

Choose and Build a File System

ls usr/sbin/mkfs* : to list all the aliases of mkfs, which will tell you what filesystems are supported

mkfs : to build a filesystem

```
mkfs -t fstype device
```

e2label : to change the label of a partition on ext filesystem

```
e2label device new_label
```

xfs_admin -L : to change the label of a partition on xfs filesystem

```
xfs_admin -L new_label device
```

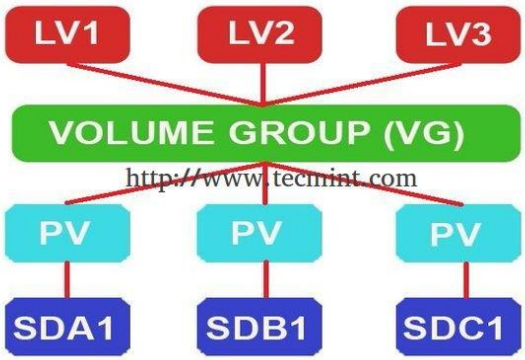
resize2fs : used to resize ext2/ext3/ext4 filesystem

```
sudo resize2fs /dev/sda1 20G
```

e2fsck : check the filesystem. *Note: -f to force check even if clean*

```
sudo e2fsck -f /dev/sda1
```

Logical Volume Manager (LVM)



Physical Volumes

pvcreate : create a physical volume on a disk so it's recognized as belonging to LVM

(pvcreate /dev/disk1 /dev/disk2.....)

pvdisplay : display information about physical volumes

pvs : same as pvdisplay but with less information

pvscan : scan block devices for physical volumes

pvremove : remove physical volume from lvm

```
sudo pvremove /dev/sdb
```

Volume Groups

vgcreate : create a volume group

```
vgcreate <volume group name> /dev/disk1 /dev/disk2….
```

vgdisplay : display information about volume groups

vgs : same as vgdisplay but with less information

vgextend : extend a volume group by adding more physical volumes

```
sudo vgextend <volume group name> /dev/sdd.....
```

vgremove : delete a volume group

```
vgremove [options] volume_group_name
```

Logical Volumes

lvcreate : create a Logical volume) *Note: after you create the logical volumes they are treated as normal disk so you can add a file system to them and add them to **/etc/fstab***

```
lvcreate -L 500G <volume group name> -n <logical volume name>
```

lvdisplay : display information about logical volumes

lvs : same as lvdisplay but with less information

lvresize : resize a logical volume [+][-]

```
sudo lvresize -L +/-50G /dev/my_vg/my_lv
```

lvreduce: reduce the size of a logical volume

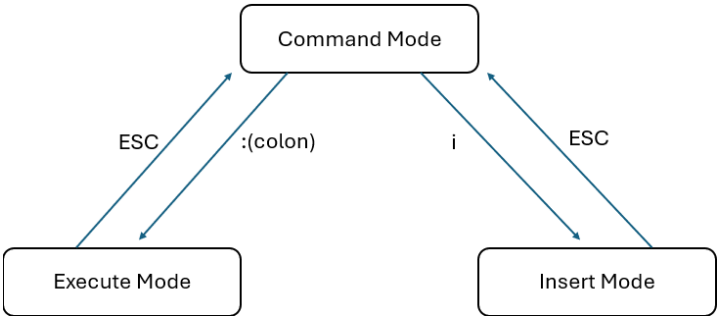
```
sudo lvreduce -L 8G /dev/my_vg/media_lv #this will be the new
size
```

lvextend: expand a logical volume

```
sudo lvextend -L +5G /dev/my_vg/media_lv
```

lvremove : delete a logical volume

Vi/Vim editor



Vi +/< pattern > filename : to open a file and places the cursor first occurance of a string

Command Mode

I : insert before line

A : insert after line

o : add new line after current line

O : add new line before current line

yy : to copy a line

p : to paste

dd : To delete a line

ZZ : save and exit

r: save and exit

/<anything> : to search for something (forwards)

? <anything> : to search for something (backwards)

Execute Mode

:w : To save

:q! : exit without saving

:d : delete a line

:p : paste

:e filename : to switch to another file

:set number : to add numbered lines

:< line number > : to jump to a specific line

:< first line >,< last line >d : to delete a range of lines

:< first line >,< last line >y : to copy a range of lines

:< first line >,< last line > s/< old word >/< new word > : to find and replace within a specific range

:%s/< old word >/< new word >/< flag >: to find and replace through the whole document

Flags:

| g | Flag - Replace all occurrences of pattern |

| c | Flag - Confirm replaces. |

| & | Repeat last :s command |

Searching and Manipulating Files

Wildcard	Description
[]	List of possible values
-	Range of values
.	Any single character
*	Any number of characters
^	Beginning of line
$	End of line
|	Or
()	Sub-expression or slice
\	Escape character

fgrep/grep -F : Fixed string search

egrep/grep -E : search using regular expressions (wildcard)

printf : format and print data

wc : print the number of lines, characters, words, and the byte count.

sort: sort the content of a text file. *Note: use **-r** for reverse sorting, **-k** to specify the column number, and **-t** for the delimiter*

cut: extract sections from the provided input. *Note: use **-f** to specify the column/field number, and **-d** for the delimiter*

diff: comparing files line by line

locate/find → used to look for files *Note: you can use 2>/dev/null with **find** to redirect and throw away any error messages*

sed : for filtering and transforming text. *Note: mostly used in scripting and automation*

awk : pattern scanning, manipulating data, and generating reports. *Note: mostly used in scripting and automation*

Examples of find

- **Find files by name :**

```
find / -name filename.txt 2>/dev/null
```

- **Find files by type :**

```
find /home/hussain/ -type f -name "*.sh" 2>/dev/null
```

- **Find directories by type :**

```
find / -type d -name "project" 2>/dev/null
```

- **Find Files Modified in the Last 7 Days:**

```
find /home/hussain/ -type f -mtime -7 2>/dev/null
```

- **Find Files Larger Than 100MB:**

```
find / -type f -size +100M 2>/dev/null
```

- **Find Files and Execute a Command:**

```
find /home/hussain/ -type f -name "*.log" -exec rm {} \; 2>/dev/null


#NOTE: whenever find locates a file it will substitute {} with the
path of that file. '\;' indicates the end of -exec.
```

Examples of egrep/grep -E

- **Match Lines that contain either of Two patterns:**

```
echo -e "cat\nbat\nrat\nmat" | grep -E "cat|rat"
```

cat

```
rat
```

- **Matching Lines That Start with a Specific Pattern:**

```
echo -e "start\nstop\nstand\nrest" | grep -E "^st"


start
stop
stand
```

- **Matching Lines That End with a Specific Pattern:**

```
echo -e "running\njumping\nswimming\ncycling" | grep -E "ing$"


running
jumping
swimming
```

- **Matching Lines That Do Not Contain a Specific Pattern:**

```
echo -e "foo\nbar\nbaz\nqux" | grep -Ev "ba"
foo
qux
```

- **Matching Lines with a Specific Number of Characters:**

```
echo -e "a\nab\nabc\nabcd" | grep -E "^.{2}$"
ab
```

- **Matching Lines That Contain a Number:**

```
echo -e "word1\nword\nword2\nword3" | grep -E "[0-9]"
word1
word2
word3
```

- **Matching Lines with One or More Occurrences of a Character:**

```
echo -e "a\nab\nabb\nabbb" | grep -E "ab+"


ab
abb
```

```
abbb
```

- **Matching a Pattern Repeated a Specific Number of Times:**

```
echo -e "a\nab\nabb\nabbb" | grep -E "ab{2}"


abb
```

- **Using Grouping to Match Complex Patterns:**

```
echo -e "catdog\ncatcatdogdog\ncatdogdog" | grep -E "(cat|dog){2}"
catcatdogdog
```

- **Matching a Word Boundary with \b:**

```
echo -e "word\nwordplay\nsword\nplayword" | grep -E "\bword\b"


word
```

- **Matching Lines with an Optional Character:**

```
echo -e "gray\ngrey\ngreen" | grep -E "gr[ae]y"
gray
grey
```

Archiving and Compressing

tar : to create an archive, which means combining multiple files into one file without reducing their size

```
tar [options] archive_name.tar file1 file2 directory/
```

gzip/guzip : compress and decompress using GNU zip . *Note: commonly used as it provides a balance between speed and compression ration*

xz : to compress and decompress using xz tool . *Note: best compression ratio between **gzip** and **bzip2**, but is the slowest*

tar options

-c : Create a new archive

-v : Verbose (optional, shows progress).

-f : Specify the name of the archive file.

-x : Extract the files from the archive.

-z : compress and decompress using gzip . *Note: end with **.tar.gz***

-J : compress and decompress using xz. *Note: end with **.tar.xz***

-p: preserve permissions.

--same-owner: to keep the ownership of the files and directory untouched.

--no-same-owner: it will make you the owner.

-t : List the contents of the archive.

Backing up

dd

```
sudo dd if=/dev/sda1 of=/mnt/Storage1/boot.img status=progress
```

bs= : to specify how much data to grab each time.

conv=: ***sync*** to compare the copied data with the source and if there are no errors it will copy the second block of data. ***noerror*** *if any error occure just continue*

count=: to specify the number of blocks to copy.

To restore just flip the if and the of. However, if it was zipped you will need to unzip it first, and don't forget to unmount the partition or the disk.

rsync

```
rsync [azurAPgoe] source destination


#Note: -a= archive, -z= compress, -u= skip newer files on the
destination, -r= recursive, -A= copy all permissions (unix && FACL ) -
P= show Progress, -g= preserve groups, -o= preserve ownership , -e=
execute another command first
```

- **Copying files locally and preserve permissions:**

```
rsync -azurAP /home/Mike/Documents /mnt/Storage1
```

- **Copying files to a remote location:**

```
rsync -azurP -e ssh /home/Mike/Documents Mike@192.168.100.51:
/mnt/Storage1
```

- **Including and excluding files:**

```
rsync -azurP /home/Mike/Documents /mnt/Storage1 --include "*.pdf" --exclude ".*"
```

- **To test and visually see what rsync will do:**

```
rsync -azurP /home/Mike/Documents /mnt/Storage1 --dry-run
```

Scheduling Tasks Using Cron

```
* * * * * /path/to/command
| | | | |
| | | | └──── Day of the week (0-7) (0 and 7 both represent Sunday)
| | | └─────── Month (1-12)
| | └──────────── Day of the month (1-31)
| └─────────────── Hour (0-23)
└────────────────── Minute (0-59)
```

Ex:

```
#Run a script every day at 2:30 AM:
0 0 * * 0 /path/to/backup.sh
# Run a command every 5 minutes:
*/5 * * * * /path/to/command
```

crontab -l : View the current crontab

crontab -e : Edit/create the crontab:

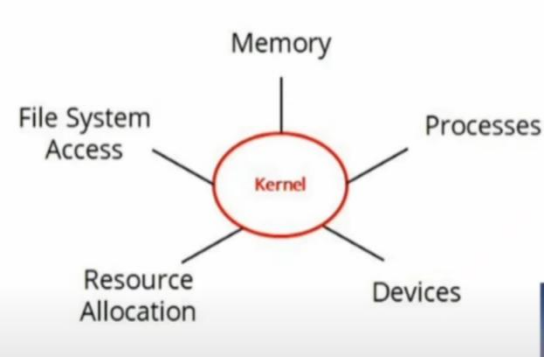
crontab -r : remove a crontab:

Special Directories

You can place executable scripts in these directories, and they will be run at the appropriate intervals.

- /etc/cron.hourly/
- /etc/cron.daily/
- /etc/cron.weekly/
- /etc/cron.monthly/

Boot Process and Kernal



uname : displays system information

```
uname [options]
-a: Display all available system information.
-r: Display the kernel release.
-s: Display the kernel name
```

lsmod : displays the loaded kernel modules

modinfo: displays information about a kernel module

```
modinfo [options] module_name
modinfo ext4    # Display information about the 'ext4' module

-d: Display the module's description.
-p: Display parameters of the module
```

insmod : install a module to the kernel. *Note: it does not handle dependencies, it is recommended to use modprobe*

rmmod: remove a module from the kernel

modprobe : activate and deactivate modules

```
modprobe [options] module_name

sudo modprobe -a ext4    # Load the 'ext4' module into the kernel
sudo modprobe -r ext4 # Remove the 'ext4' module from the kernel
```

sysctl : displays kernel parameters

Managing Services

ps aux : display all process with the user attached to each one (snapshot)

pstree : show process tree (parent and child process)

pmap : show every thing that a certain process is using (libraries.....)

top/htop : display all process with the user attached to them (live)

lsof : list open files. *Note: remember everything in Linux is a file, so you can use it to know what process uses a particular port, or to know what process is using a particular directory, or to list opened files for a particular user*

jobs : to see what applications you put in the background. *Note: you can an application in background using **CTRL+Z** or using **bg** command and to bring back to the foreground you can use the **%+jobnumber** or the **fg** command*

kill/killall : to terminate a process

pgreg :used to grep the PID of an application, It is the same as ***ps aux | grep <process>***

nice: start an application with a certain priority

```
nice -n 10 my_application
```

renice: to change the priority of a process or an application. (already running)

```
sudo renice 5 -p 1234

# Change the priority of process with PID 1234 to a nice value of 5


sudo renice -10 -u username

# Increase the priority of all processes belonging to 'username'
```

systemctl : used to manage services

systemd-analyze: reveals performance statistics for boot times, traces system services, and verifies unit files

systemctl options

systemctl status : to check the status of a service, showing if it's running, stopped, or in a failed state, along with recent log messages.

systemctl enable : to configure the system a service automatically at boot.

systemctl disable : to prevent a service from starting automatically at boot. *Note: in some cases, even if you disable a service some other service might trigger this service to start and it will start again, so if you want to prevent from loading even if other services depend on it use **systemctl mask.***

systemctl start : to start a service making it active.

systemctl stop : to stop a service making it inactive.

systemctl restart : to **r**estart a service, useful for applying configuration changes or refreshing the service.

systemctl reboot/poweroff : to safely reboot or shutdown the system.

systemctl list-unit-files : lists all the installed unit files

systemd-analyze options

systemd-analyze time : display the time taken for each major stage of the boot process

systemd-analyze blame : lists all the system services that were started, sorted by the time they took to initialize. *Note: useful for identifying which services are slowing down the boot process*

systemd-analyze plot : generates an SVG graphic that visualizes the boot process, showing when each service started and how long it took. *Note: redirect the output to a **.svg** file and then you can view the output in any browser or image viewer*

systemd-analyze security : to assess the overall security of the services

Managing and Configuring Hardware

lsusb : list connected usb devices

lspci: list PCI buses in the system and devices connected to them.

dmesg : display the log messages in the kernel ring buffer.

dmidecode : retrieve hardware related information in human readable format

Networking

nmcli : controlling the network manager from the CLI

Ip addr/ifconfig : show ip address configuration.

ip route/route : to see your default gateway.

nslookup : to get the ip of a particular domain name.

dhclient : turn on the dhcp service.

traceroute/tracepath : trace the hops (routers) between you and a specific address like www.google.com

ss : session statistics. *Note: e.g. **ss -an,** to display all session and disable name lookup.*

hostnamectl set-hostname <newname> :to change your computer name

telnet : remote connection (not secure)

ssh : secure remote connection

nc : connect on a specific port

```
nc www.test.com 80
```

how to use nmcli

General Syntax

```
nmcli OBJECT {COMMAND | help}
```

1. General Network Information

- **Status of NetworkManager (Displays the status of the NetworkManager service):**

```
nmcli general status
```

2. Device Management

- **List All Network Devices:**

```
nmcli device status
```

- **Bring a Device Up/Down:**

```
nmcli device connect/disconnect <device_name>
```

- **Show Device Information:**

```
nmcli device show <device_name>
```

3. Connection Management

- **List All Connections (Active and Inactive):**

```
nmcli connection show
```

- **List Active Connections:**

```
nmcli connection show --active
```

- **Add a New Wi-Fi Connection (SSID + Password):**

```
nmcli device wifi connect <SSID> password <PASSWORD> name
<connection_name>
```

- **Reload Connections:**

```
nmcli connection reload
```

4. Wi-Fi Specific Commands

- **Scan for Wi-Fi Networks:**

```
nmcli device wifi rescan
```

- **List Available Wi-Fi Networks:**

```
nmcli device wifi list
```

- **Connect to a Wi-Fi Network (SSID + Password):**

```
nmcli device wifi connect <SSID> password <PASSWORD>
```

5. Basic Commands in Edit Mode

```
nmcli connection edit [connection_name_or_uuid]
```

help: Get help while in the editor

print: View all settings for the connection

- **Set a static IP:**

```
set ipv4.addresses 192.168.1.50/24
```

- **Set a gateway:**

```
set ipv4.gateway 192.168.1.1
```

- **Set DNS:**

```
set ipv4.dns 8.8.8.8
```

- **Switch IPv4 method to manual:**

```
set ipv4.method manual
```

- **Save the changes and exit:**

```
save
quit
```

Network Firewall (Firewalld)

Firewalld Tutorial

https://youtu.be/jgErVHBz7XI?si=ohEllMYwXvvjNcxz

- **Check the status of firewalld:**

```
firewall-cmd --state
```

- **List available zones:**

```
firewall-cmd --get-zones
```

- **To see what the default zone is:**

```
firewall-cmd --get-default-zones
```

- **List active zones with interfaces attached to them:**

```
firewall-cmd --get-active-zones
```

- **List available zones:**

```
firewall-cmd --get-zones
```

- **To create a new zone:**

```
firewall-cmd --permanent --new-zone=testZone

#Note: use --permanent to write the configuration to disk
```

- **To reload the firewall:**

```
firewall-cmd --reload
```

- **To change the zone of an interface:**

```
firewall-cmd --zone=<new zone> --change-interface=<interface
name> --permanent

#Note: Normally, when using --permanent you need to reload
firewalld to apply the config to runtime, but in the case of
changing the zone firewalld updates both permanent config and
runtime.
```

- **List all supported services:**

```
firewall-cmd --get-services
```

- **To allow a service in a specific zone:**

```
firewall-cmd --permanent --zone=<zone name> --add-
service=<service zone>


#Note: if you are modifying the default zone, then you don't have
to specify the zone.

#Note: This is being written to disk, so you will have to reload
the firewall and that will interrupt the network traffic. To
avoid that run the command again without --permanent to  modify
what the config running in the ram
```

- **To allow a port in a specific zone:**

```
firewall-cmd --permanent --zone=<zone name>  --add-port=<port
number>/tcp


#or you can add a range


firewall-cmd --permanent --zone=<zone name>  --add-port=10000-
20000/tcp
```

- **List allowed services in a specific zone:**

```
firewall-cmd --zone=<zone name> --list-services


#Note: this will show you what is running in the ram, you can add
--permanent to see what is written on the disk.
```

- **List allowed ports in a specific zone:**

```
firewall-cmd --zone=<zone name> --list-ports


#Note: this will show you what is running in the ram, you can add
--permanent to see what is written on the disk.
```

- **List applied rules in a specific zone:**

```
firewall-cmd –zone=<zone name> --list-rich-rules
```

- **Display the manual page for rich language format:**

```
man firewalld.richlanguage
```

Security

encrypting disks with luks

It is a good idea to use shred to make sure that the previous data is not recoverable and securely wiped.

```
sudo shred -v --iterations=3 /dev/sdX
```

encryption using luks

```
1. sudo cryptsetup luksFormat <device>
```

this will format the disk as an encrypted disk, so make sure you back up your data and it is always a good idea to unmount the drive.

```
2. sudo cryptsetup luksOpen /dev/sda <assign a name to it>
```

this will create a mapper entry for the encrypted disk using the name you assign: /dev/mapper/my_encrypted_vol. you will also use it to open the volume once you encrypted

```
3. sudo mkfs.ext4 /dev/mapper/my_encrypted_vol

4. sudo mount /dev/mapper/my_encrypted_volume /mnt/my_mount_point
```

format the encrypted disk with a file system of your choice and mount it.

```
5. sudo umount /mnt/my_mount_point
6. sudo cryptsetup luksClose my_encrypted_volume
```

after you are done first unmount it and then close it using the above command, or if you shutdown the system it will automatically unmount it and closed it

audit tool

a tool for monitoring and tracking system events for security and compliance purposes. It allows administrators to track access to files, monitor system calls, and detect suspicious behavior in the system.

Key Components of audit

- **auditd**: The main audit daemon is responsible for writing audit events to the log.
- **Audit rules**: These are the rules that define which events are captured (e.g., system calls, file access, etc.). . *Note: usually stored in **/etc/audit/audit.rules***
- **Audit logs**: The logs generated by auditd are typically found in /var/log/audit/audit.log.

```
sudo auditctl -w /etc/passwd -p wa -k passwd_changes
```

- **-w** : Watch the file /etc/passwd.
- **-p** : Monitor write and attribute changes.
- **-k** : Add a key to identify the audit rule.

auditctl -l : list audit rules

ausearch -k : view the logs of a specific file

changing ssh keys

it's recommended to regenerate the ssh keys after the first boot, because during the first boot of a system it might not have enough entropy(randomness) to generate a strong key so the key will be less secure and predictable. It is the default for RedHat-based distros

1. Navigate to /etc/ssh and delete the ssh keys
2. Regenerate the keys using ssh-keygen. Make sure you create all of them and use the same file names as the previous keys

```
sudo ssh-keygen -t rsa -f /etc/ssh/ssh_host_rsa_key
```

3. When you are finished generating the keys, restart the ssh daemon (sshd)

SELinux

Security-Enhanced Linux (SELinux) is a security architecture for Linux systems that enforces access control policies to govern how processes and users can interact with files, devices, and other processes, so it adds an additional layer of security

Enforcing Mode: SELinux policy is enforced, and violations are blocked and logged.

Permissive Mode: SELinux policy is not enforced, but violations are logged. This mode is often used for troubleshooting.

Disabled: SELinux is turned off entirely.

```
sestatus
getenforce
```

This command will give you information about SELinux, getenforce will give you only the current mode, while sestatus will give you more information such current mode and mode configured in config file..... *Note: to see the log files generated by SELinux, you can navigate to* ***/var/log/audit/audit.log***

```
sudo setenforce permissive/enforcing
```

Temporarily change SELinux, you can use the number instead of the words. Permissive is 0 and enforcing is 1

```
ls -Z /var/www/html
```

to see the SELinux context of the files in this directory

```
ps auxZ
```

display process with their SELinux context

```
sudo chcon -t httpd_sys_content_t /var/www/html/index.html
```

Used to temporarily change the context of a file

```
sudo restorecon /var/www/html/index.html
```

this will revert a file to its default context, so any changes made by chcon will go

```
sudo semanage fcontext -a -t httpd_sys_content_t '/mydata(/.*)?'
```

To change a context and make it persistence use the above command. This will change the default context, so after running this command, run restorecon on the same file to change it to the new context.

```
sudo semanage port -a -t http_port_t -p tcp 8080
```

to add a port to the allowed ports for httpd

AppArmor

Similar to SELinux, AppArmour Provides mandatory access control. Unlike SELinux which uses subjects (e.g. processes) and Objects (e.g. file, port, dir), AppArmour implements a profile-based approach (more user-friendly) where every application protected by AppArmour has a profile, which is a file

Enforcing Mode: In this mode, AppArmor enforces the rules defined in the profile. Any action that violates the policy will be blocked and logged.

Complain (Permissive) Mode: In this mode, violations are logged but not blocked. This is useful for debugging and developing new profiles.

```
sudo aa-status
```

display the current status, loaded profiles, and what mode they are put in.

```
sudo aa-unconfined
```

list process running without AppArmor profile

```
sudo aa-genprof /usr/bin/apache2
```

To generate a new profile for an application, once you run the command it start monitoring the application activity to generate the profile.

```
sudo aa-logprof
```

used to update existing profiles. Once you run the command it will analzye the logs generated while the profile was in complain mode and helps you by suggesting changes to the corresponding profiles

```
sudo aa-autodep
```

a quick, initial and basic profile generated by anlayzing the application, and it will include the permission for necessary dependencies. A good starting point, if don't want to go through the process of **aa-genprof** immediately.

```
sudo aa-enforce /etc/apparmor.d/usr.bin.apache2
```

put a profile into enforcing mode

```
sudo aa-complain /etc/apparmor.d/usr.bin.apache2
```

put a profile into complain mode

```
sudo aa-disable /etc/apparmor.d/usr.bin.apache2
```

disable a profile.

```
sudo aa-remove-unknown
```

removes unknown/unused AppArmor profiles.

Installing and Managing Software

apt : used to manage packages on Debian-based distros. *Note: it combines the functionality of apt-get and apt-cache*

yum/dnf : the package manager for redhat-based distros. *Note: yum is being gradually replaced by dnf, so dnf is the future*

rpm : redhat package manager, this is the old package manager that redhat-based ditsro used to use. *Note: you shouldn't use it as it has been replaced by yum/dnf, however it is still used to import GPG key*

apt options

apt list : used to list packages. *Note: used if you know what you are looking for*

apt search : used to search for packages by keywords. *Note: used if you don't know what you are looking for, so you can provide a keyword, and it will search in the packages and their description for a match*

apt install : used to install packages.

apt-key add : to add a key or a hash, so it can be used to verify packages.

apt update : to update the apt cache.

apt upgrade : to upgrade the software packages installed.

apt full-upgrade :upgrade packages and install new packages or remove existing ones as needed.

apt remove : to remove/uninstall a software.

apt autoremove : Removes dependency libraries and packages that are no longer required.

apt clean : Clears the APT cache.

apt autoclean : Cleans up partially downloaded packages.

yum/dnf options

yum list: to list available packages with a specific word.

yum search : to search in repos for a package by a keyword.

yum info: to see information about a package .

yum install : to install packages.

yum remove : to remove/uninstall a software.

dnf autoremove : look for any unused dependencies and remove them.

yum update : to update the cache, install the updates, and upgrade the OS. *Note: unlike apt where you have three different commands; one for updating the cache, one for installing the updates, and one for upgrading the distro, **yum update** combines the three in one command*

yum provides */xeyes : to find the package name fo xeyes

rpm --import : to add a key or a hash, so it can be used to verify packages.

Git

- **Install git:**

```
sudo apt install git
sudo dnf/yum install git
```

- **save git configuration on the system level:**

```
git config --system
```

- **save git configuration for a user:**

```
git config --global
#Note: you need to be logged in as that user
```

- **save git configuration just for the project:**

```
git config --local
#Note: you can remove --local as it's the default
```

- **basic configuration (name, email, the text editor you want to use and turn on colors):**

```
git config --global user.name "Hussain Ahmed"
```

```
git config --global user.email "husmohdali@gmail.com"

git config --global core.editor "vim"

git config --global color.ui true


#Note: this is a user-level config, so you need --global
```

- **list configuration:**

```
git config --list
```

- **initialize a directory to be tracked by git:**

```
git init

#Note: you need to be inside the directory where you want to
initialize the repository. Also, you can do ls -al to double check
that the .git dir has been created
```

- **check git status (which files aren't being tracked, any changes have been made):**

```
git status
```

- **start tracking a file (push to staging area):**

```
git add test1.txt

#Note: it's like you telling git to pay attention to this file, but it
doesn't know if changes have been made.


#Note: you can use . to specifiy everything in the current directory.


To execlude certain files from being tracked

     1. Create a .gitignore

     2. name the files inside the .gitignore
```

- **Commit changes (push to repository):**

```
#Note: it is like telling git that I made changes to the file keep
track of them


git commit -m "initial commit"


#Note: -m to add a short description fo the changes. You can use -M
and it will open the editor so you can add a detailed description.


git commit -a


#Note: commits all changes and open the editor to write a description
```

- **See your commits:**

```
git log
```

- **Check what chages have been made:**

```
git diff test.txt


#Note: this will allow you to see the differences between the current
version of the file and the last save point or the last vesion you
committed
```

- **Roll back to the last save point:**

```
git checkout test.txt
```

- **Push a local repository to a remote repository :**

```
#Specify the remote repository

git remote add <name> <url>

#Note: in the name you can add anything, but it is a conventional rule
to use origin as the name.


#upload/push the local repository. Syntax: git push -u <name> <branch
name>


git push -u origin main
```

- **Download/clone a remote repository :**

```
git clone <url of remote repository>
```

- **Create a branch:**

```
git branch <name of new branch>
```

- **List created branches:**

```
git branch
```

- **Switch to another branch:**

```
git checkout <branch name>
```

- **merge branches:**

```
git merge <brach name>
```

Capacity Planning

iotop : to see disk I/O

Sar

SAR or System Activity Reporter is a service that runs in the background and collects performance data and logs it, so then you can go and read it later. It tracks CPU, memory, disk I/O and network I/O

- **install SAR:**

```
sudo apt install sysstat
```

- **enable SAR:**

```
edit /etc/default/sysstat and make sure its enabled

Ex:-

ENABLED="true"
```

- **Change how often SAR run:**

```
When you installed SAR, it created a cron job in /etc/cron.d, so
navigate to this dir and edit the cron job inside sysstat file
```

- **Display collected performance data :**

```
sar [urbn]
-u: CPU
-r: RAM
-b: disk I/O
-n: network I/O
```