

Linux Foundation Course (LFS101)

Basic Linux Command: (COMMAND OPTIONS ARGUMENTS)

Command	Flag	Description
sudo (MORE)	-l : List command with root / sudo permission	Executes a command with root / admin privileges Allows you to execute the command as a superuser / admin SUDO SETUP (MORE)
sudo apt update		To update system
sudo apt upgrade		To upgrade the packages

Managing Services and Startup

sudo systemctl stop gdm sudo telinit 3	FN + ALT + F2	To switch GUI from CLI
sudo systemctl start gdm sudo telinit 5		To switch CLI from GUI
sudo systemctl status service-name		Shows if service is running
sudo systemctl enable --now service-name		Starts and enable the service to run at boot
sudo systemctl --failed		List services that have failed to start

Package Installation

sudo apt install package_name		To install a package
sudo apt remove package_name		To remove a package
package_name --version		To check existency of package
which package_name		Locate application in the system
whereis package_name		Locate application in the system

Service Control

service bluetooth stop		Turn-ON bluetooth
service bluetooth start		Turn-OFF bluetooth
nmcli radio wifi on		Turn-ON WiFi
nmcli radio wifi off		Turn-OFF WiFi
nmcli device wifi connect "SSID_NAME" password "PASSWORD"		To connect specific wifi

Linux Basic and Documentation

cat /proc/version uname -r neofetch		Info about the linux system
help	cmd --help cmd -h	Provides information about on built-in commands
man	-f / whatis. -k / apropos. -a	Display the manual page
Info		GNU Info
echo		To show output
exit		To close the terminal
history	- h : Save history - c : Clear	To show history of the cmd ^ : Up Arrow v: Down Arrow CTRL - R (Reverse search) + ENTER !! : Execute prev CMD ! : History substitution !\$: last ARG !n : nth CMD !string: Most Recent CMD
whatis CMD		About command
Environment Variable		
echo \$<var_name>		Shows the value of a specific variable
export VARIABLE=VALUE		Export a new variable value
set grep HIST	HISFILE HISFILESIZE HISSIZE HISCONTROL HISIGNORE	History ENV var
Basic Navigation		
pwd	-L : Symbolic Link -P : Target Link	Print Working Directory \$PWD variable holds path Types of Path (Absolute & Relative)
tree	-d : directory -a	Show directory structure
ls dir dirs	- l : long listing format - a : hidden file - h : human readable -ld: check permission of directory - t : Sort by time	List directory contents ls <file_path>

	<ul style="list-style-type: none"> - 1: list in new line - i : inode -R: Recursive file 	
cd pushd popd	<ul style="list-style-type: none"> . current .. previous - previous / root ~ user home directory 	Change directory
Directory Management		
mkdir folder_n1 folder_n2 mkdir -p parent/child mkdir folder_n1{1,2,3...}	<ul style="list-style-type: none"> - p : make parent d - v : log for create d - m : Set permission 	Make new directory
TEMPDIR=\$(mktemp -d /tmp/tempfile.xxx....x)		Make temporary directory
touch file_name , file_name , ... touch file_name echo (data) > file_name echo (data) >> file_name echo \$null >> file_name	<ul style="list-style-type: none"> - a : change access time - m : change modification time - c -r : copy both access and modification time - t : custom timestamp 	Create an empty file Used to set / update the access, Change and modify times of files
TEMP = \$(mktemp /tmp/tempfile.xxxx....x)		Make temporary file
stat file_name		Verification of file details
namei		To check the file path through layer of folder's path
cp <source> <destination>cd ~	<ul style="list-style-type: none"> - r : recursive - i : interactive - f : force - p : preserve 	Copy files / directories cp folder_name folder_name -r cp file_name folder_name
mv <source> <destination>	<ul style="list-style-type: none"> - u : move only if source is newer - i : interactive - f : force 	To Move / rename files / directories mv <file_name> <new_file_name> mv <folder_name> <folder_name> (rename) mv <file_name> <new_folder_name> (move)
rm	<ul style="list-style-type: none"> - r : recursive - f : force deletion - i : interactive deletion 	Remove file or directory rm folder_name -r rm file_name
rmdir	<ul style="list-style-type: none"> - p : remove directory and ancestors - v : show message on terminal 	To remove an empty directory

In	-s: soft link creation	hard link creation
Trash Management		
trash-list		List the files
trash-empty		Empty trash
gio trash file_name		Move to trash files
Text View and Edit		
cat	- n : number of o/p line	To display content of the file (U-B)
cat file_name cat file_name file_name cat file_name file_name > new cat file1 >> existing_file cat > file (Overwrite) cat >> file (Append) cat << EOF > file > > > EOF	- b : add number to the test line - E : add \$ at the end of each line	Create an insert content Cat >> file.txt cat > file.txt + text + Ctrl D copy content of one file to another cat file_name > file_new_name
tac		To display content of the file (B-U)
less	-N : number of o/p line	To display content terminal wise
more		
head	- n <number>	Display first few line of file (Default 10)
tail	- n <number> - f	Display last few line of file (Default 10)
nl		Number the lines of our text data
diff [opt] <file1> <file2> cmp [opt] <file1> <file2> patch	-c: show 3 result -r: -i: ignore case -w: ignore white space -q:	Compare files and directory GUI (diffuse, vimdiff, meld)
nano / pico / emacs / ed / gedit	- B : backup previous file	Text editor
vim		Text editor
nautilus		Open file Manager
open		To open file into GUI
code .		Open file in VS code
sort <file_name>	- r: Reverse order - k n : - u: unique line	Sort the lines in the files
uniq	-c : count duplicate	Remove duplicate consecutive lines in text file

paste	- d: separator - s:	Concatenate data from different file
join		Concatenation of data from different file based on common filed
split (More)		Break file into equal size segment
tr		To translate char into other char
tee		To save output stream data into file
wc file_name (More)	- l : No. of lines - c: No. of bytes - w: No. of words	Count information in a file Line Words Char <file_name>
cut		To extract specific columns
Search and Filter		
grep (Global Regular Expression Print) (More)	- i : ignore case - r : recursive - n : number of lines - v : Not match	To search pattern into file grep -i " " file1.txt grep -r " " file1.txt grep -n " " file1.txt
find (More)	- name : search by name - iname - delete : delete file - type (d/f/l) - exec - ok - ctime / atime / mtime - cmin / amin / nmin - size (+/-)	Search for files in a directory find <path> -delete ls
locate (More) updatedb		Fast file searching Path are stored here
strings file_name		To extract all printable char strings found in the file or files given as arg
sed (stream editor)		To search, filter and replace operation on our data To modify the contents of file or input stream (Placing the contents into a new file / output stream)
sed -e CMD <file_name>		Specify editing commands at the command line, process input from a file, and put the output on standard out
sed -f scriptfile <file_name>		Specify a script file containing sed commands, operate on file, and put output on standard out

sed s/word1/word2 file		Filter standard input, putting output on standard out
sed s/pattern/replace_string file (One)		Substitute first string occurrence in every line
sed s/pattern/replace_string/g file (All)		Substitute all string occurrences in every line
sed 1,3s/pattern/replace_string/g file (Range)		Substitute all string occurrences in a range of lines
sed -i s/pattern/replace_string/g file (Save)		Save changes for string substitution in the same file
awk		To extract and then print specific contents of a file
awk "CMD" file		Specify a command directly at the command line
awk -f scriptfile file		Specify a file that contains the script to be executed
awk '{print \$0}' file		print entire file
awk -F: '{print \$1}' file		Print first field of every line separated by a colon
awk -F: '{print \$1 \$7}' file		Print first and seventh field of every line
Wildcards	*: String of Char ?: Single Char [set]: Any ![set]: Not Any : . : Single char	

File I/O Management

CMD < input-file		Change Input Source (Take From File)
CMD > output-file CMD 1> output-file		Change Output Stream (Give To File)
CMD < input-file > output-file		
CMD 2 > error-file		Change Error Stream (Give To File)
CMD > all-output-file > 2&1 CMD >& all-output-file		Send error and output to same file
lsof		List open files

File Compression

zip (.zip)	- r : recursive -e : encryption -u : update -d : delete -m: move	To compress a directory zip <flag> arch_name.zip folder_name
unzip (.zip)	-l : list content of file without zip -d : unzip to specific directory	To expand a directory unzip arch_name.zip unzip <flag> arch_name.zip unzip arch_name.zip -d path
tar (.tar / .tar.gz / .tar.bz2)	- cvf : Create (.tar) - xvf : Extract (.tar) - tvf : list content without extract - rvf : append file to tar file - C: extract tar to specific directory -czvf: Create (.tar.gz) -xzvf: Extract (.tar.gz) -cjvf: Create (.tar.bz2) -xjvf: Extract (.tar.bz2) -cJvf: Create (.tar.xz) -xJvf: Extract (.tar.xz)	Archive files and directories tar -cvf name.tar file_path / file tar -xvf name.tar file_path / file tar -czvf archive.tar.gz file_path / file tar -xzvf archive.tar.gz tar -cjvf archive.tar.bz2 file_path / file tar -xjvf archive.tar.bz2
gzip (.gz)	(Single file required)	gzip file_name gzip * gzip -r <directory_name>
gunzip (.gz)	(Single file required)	gunzip file_name
bzip2 (.bz2)		bzip2 * bunzip2 *.bz2
xz		xz *
zcat / xzcat / bzcat		To view a compressed file
zless / xzless / bzless		To page through a compressed file
zmore / xzmore		To page through a compressed file
zgrep / xzgrep		To search inside a compressed file
zdiff / xzdiff / bzdiff		To compare two compressed file
System ON / OFF		
reboot shutdown -r	- f : force	To reboot the system
Sudo shutdown <flag> now	- r : restart - h: shutdown	To restart/ shutdown PC

Sudo poweroff sudo halt reboot -p		To shutdown PC
Shutdown -h <time> "msg"		To shutdown at specific time
login		
last -n <number>		Show last login to system
Process Management (MORE)		
echo \$\$	PID (Process ID)	To show current process id
echo \$PPID	PPID (Parent Process ID)	To show parent process id of current id
	RUID (Real UID)	Used to identify the user who started the process
	EUID (Effective UID)	Determines the access rights of the user
	RGID (Real Group ID)	Identifies the group that started the process
	EGID (Effective Group ID)	Determines the access rights of the group
pgrep		Shows the PID of a process through its name
pidof	- s : return one pid	Finds PID of running process based on its name
pstree [options] [PID]	-a: Show command line arguments -p: Show PIDs -c: Disable compaction -n: Sort process -u: Show UID -h : -g :	To display process tree
ps -elf	- f : full-format result - e : Display all process	Static snap short of running process
ps (process status)	- aux : show all process with detail information - axo : show specific info - l : priority and nice - f : PPID - u : username	Static snap short of running process
renice (+/-) n <PID>		Change Priority of the process nice value (Priority)
top (More)	- H : threads	Dynamic real time view of system process

w		
uptime (More)		To find out how long the system is active (running)
htop / atop / btop		Colorful UI
jobs	- l	List background and suspended process
Oneko &	Fun command (& used for background process)	
Fg %[job_id]		Send back to ahead
Bg %[job_id]		Send ahead to back
Kill [option] pid kill -SIGKILL <pid> kill -9 <pid>	- 9 : force kill - s : Specify signal	To terminate process
at		To execute any non-interactive command at specific time
cron	/etc/crontab	Time-based scheduling utility program
crontab	-e	To open crontab editor to edit existing jobs / to create new jobs
anacron	/etc/ancrontab	MOdern Linux Distro
sleep		Delays execution for a specific period

Fun Command

cowsay	Fun command	
nyancat	Fun command	
sl	Fun command	
lolcat	Fun command	
figlet	Fun command	
cmatrix	Fun command	
cal	Show calender	

System Information and Monitoring

uname	-a	System details (Kernel version, machine type, OS name)
hostnamectl		System information
lscpu [options] (More)	- b: - c: - p: info in parsable format - a / --all: display both online and offline CPU	CPU information

	<ul style="list-style-type: none"> -online: online CPU offline: offline CPU - x: - y: -bytes: Info in bytes -e: info in human readable format -json -caches -hex 	
lsblk	- l : list partition	To show disk, partition and mount point
sudo fdisk	- l : list partition	To show disk partition
free	<ul style="list-style-type: none"> - h : human-readable format - m : machine-readable format 	Show memory usage
Real-Time Monitoring		
vmstat n		Display system stats updated every n sec
<u>iostat -xz 1</u>		Shows how busy your disks are and if any are overloaded.
iftop		Display real-time network usage
Check Hardware and Devices		
lspci	Graphics card, network	Lists all PCI devices
lsusb		Lists USB devices
sudo dmidecode -t system		Shows hardware details like manufacturer, model and BIOS
View Logs and System Events		
journalctl -xe		Shows recent system errors and warnings
journalctl -u servicename		Displays logs related to particular service
dmesg -T		Shows kernel message
Checking Disk and File Usage		
<u>df</u>	<ul style="list-style-type: none"> - i : display inode usage - k : display disk use - h : disk usage in the human-readable format - T: 	<ul style="list-style-type: none"> Display inode usage of filesystems Display used and available disk space Shows filesystem usage and type
cat /proc/meminfo		To display memory info

du	- sh :	Display disk space usage
User Management (More)		
cat /etc/passwd cat /etc/shadow <i>getent passwd</i>		To view user
<u>su</u> - su - <user_n>		
sudo vi pw		To edit passwd file
whoami		Returns username of the current user
who	-a:	To list the current logged-on users
alias		
unalias		
id <user_name> id		Check user information
sudo finger [user]		Check user information
last		Show last login in system
groups <user_name>		Check Group Memberships
sudo <u>passwd</u> [user_name]	-l : Lock User Account -u : Unlock User Account	Set / Reset user password
sudo <u>useradd</u> <user_name> sudo <u>adduser</u> <user_name>	-M :Without / directory -c : Comment to user -s : Terminal -d : Directory_Path -u : UID -g : GID -e : Expiry Date -p : Unencrypted Pass	To create user (n_us):(us.pass.holder):UID:GID::us_home_dir:shell_to_user
sudo <u>usermod</u> <CHG> <user_n>	-u <new_id>-<user_n> -c <Comment> <user_n> -g <new_id><user_n> -U: unlock account -L: lock account -I <new_log><user_n> -d <new_path><user_n> -e <Date> <user_n>	To rename user_name
sudo userdel -r <user_name> sudo deluser -r <user_name>		To delete user
su [user_name] ssh <user_name>@localhost		Switch to another user
exit		Back to home
Group Management		

<code>cat /etc/group cat /etc/gshadow getent group</code>		To view group (n_grp):(grp.pass.holder):GID:Mem_grp
<code>sudo vigr</code>		Edit group file
<code>sudo groupadd <user_name> sudo addgroup <user_name></code>		To create group
<code>sudo groupmod -n <N_> <O_></code>		To rename group_name
<code>sudo groupdel <user_name></code>		To delete group
<code>sudo usermod -aG <g_n> <u_n> sudo adduser <u_n> <g_n> sudo gpasswd -a <u_n> <g_n></code>	-a : append mode	To add existing user in a group
<code>sudo usermod -G <g_n> <u_n></code>		To add new user in a group
<code>sudo usermod -g <new_pg> <u_n></code>		Change primary group of a user
<code>sudo gpasswd -d <u_n> <g_n> sudo deluser <u_n> <g_n></code>		To delete user from group
<code>sudo chage -l <user_n></code>		See change in password

File Management

<u>chmod</u>	(u/g/o) (+/-/=) (r/w/x)	Change file permissions <code>chmod [permission] [file]</code>
<u>chown</u>		Change file owner and group <code>chown [user] [file]</code>
<u>chgrp</u>		Change group ownership of file <code>chgrp [group] [file]</code>
<code>mkdir /mnt/<mount_point_name></code>		Create directory as the mount point with mount_point_name
<code>sudo mount <partition> <mount_point_directory></code>		Mount
<code>sudo umount <mount_point_directory></code>		Unmount

Networking Command

<u>ifconfig</u>	- a show all interface	Configure network interfaces
<u>iwconfig</u>		Configure wireless network interface
<u>ip</u> ip addr show	- a : show all address - link : display link layer info	Display or configure network interface
<u>ping</u> <hostname>	- c : count of packet	Send ICMP ECHO_request to network host

<u>route</u> ip route show	-n :	Show current routing table
route add -net address ip route add		Add static route
route del -net address ip route del		Delete static route
traceroute <address>		To inspect the route
ethtool		Queries network interface and can also set various parameter such as the speed
netstat	-r	To display all active connections and routing tables
nmap	-sP : ping scan - O : OS detection -sn	Network exploration and security auditing
tcpdump		Dumps network traffic for analysis
iptraf		Monitors network traffic in text mode
mtr		COmbines functionality of ping and traceroute
dig		Tests DNS workings
nslookup		Information about DNS server
host	- p : specific server port - t : type of DNS	DNS lookup utility
whois	- b : brief IP ranges	Display LinQuery domain registration information
wget		To download files and downloads
curl <URL> curl -o saved.html <URL>	- O : save name with same name	To transfer data from or to a server To obtain information about a URL
mapscii		To show world Map on Terminal

Short Cut

Ctrl + D	Close terminal
Ctrl + Alt + T	Open terminal
Ctrl + L	Clears the screen
Ctrl + C	Kills the current process
Ctrl + Shfit + +	Zoom out

Ctrl + -	Zoom in
Alt + Tab	To change current task
Ctrl + r	Search and execute command
Window + L	Lock the screen
Ctrl + 1 / Ctrl + 2	Toggle from icon to list and list to icon
Ctrl + H	Show hidden files
Ctrl + F	Search Box
Ctrl + S	Temporarily halt output to the terminal window
Ctrl + Q	Resume output to the terminal window
Ctrl + Z	Puts the current process into suspended background and back to prompt

Linux

- UNIX based operating system / Multiuser / Multitasking / Free / Open-Source
 - High performance / Strong security and stability / Cross platform / Flexible
 - Built-in networking and service processes (Daemons)
 - File system, User interface, System utilities, Application programs
 - Combines wide range of open-source tools and components
-

Linux History

Linux Community

Basic Terms

Kernel

- Brain / core of the Linux operating system used to control hardware
- Make interaction between hardware and application
- Manage hardware resource, Control system processes, Handle memory management
- Ex. Linux Kernel

System Libraries and Utilities

- Used to run programs and executing system calls effectively
- Provides standard function and APIs
- Ex. GNU C Library

Distribution

- Collection of programs combine with linux kernel to make linux based OS
- Ex. Red Hat Enterprise Linux, Fedora, Ubuntu and Gentoo

Boot Loader

- Program that boots the operating system
- Ex. GRUB, ISOLINUX

Service

- Program that runs as a background process
- Ex. httpd, nfsd, ntpd, ftpd and named

File System

- Method for storing and organizing files
- Ex. ext3, ext4, FAT, XFS, NTFS and Btrfs

X-Window System

- Standard toolkit and protocol to built GUI for linux system

Desktop Environment

- Graphical user interface on top of the operating system
- Ex. GNOME, KDE, XFCE and Fluxbox

Command Line

- Interface for typing commands on top of the operating system

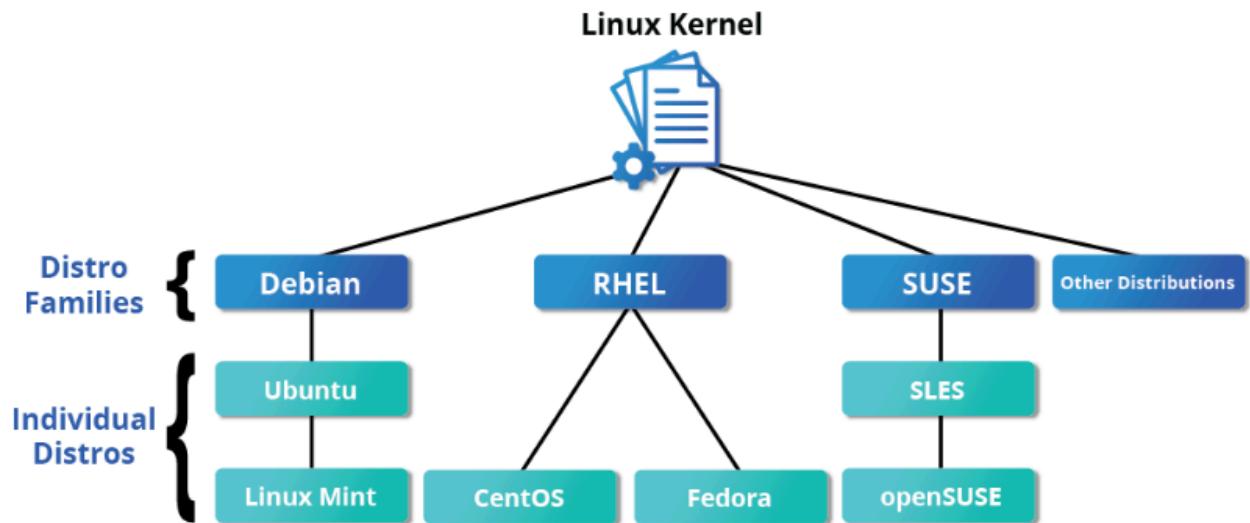
Shell

- Command line interpreter that interprets the command line input and instructs the operating system to perform any necessary tasks and commands
 - Ex. bash, tcsh and zsh
-

Linux Distribution

- Linux is combined with a collection of software packages and utilities, which are together called Linux distributions.

- [Linux distribution](#) is an operating system that is made up of a collection of software packages and utilities based on Linux kernel or you can say distribution contains the Linux kernel and supporting libraries and software



Red Hat Family System ([CentOS](#), [Fedora](#))

- Ex. CentOS, CentOS Stream, Fedora and Oracle Linux.
- dnf or RPM-based package manager
- RHEL is widely used by enterprises that host their own systems

Fedora

- Close to RHEL contains more software than Red Hat's enterprise version
- Large community is working for development of Fedora
- Fedora serves as an upstream testing platform for RHEL.

CentOS / CentOS stream

- Used for various activity, demonstrations and labs
- No cost to end user and longer release cycle
- CentOS is a close clone of RHEL
- Most popular Linux distribution in enterprise environments
- Quick update receives before RHEL for CentOS stream and CentOS receives them after

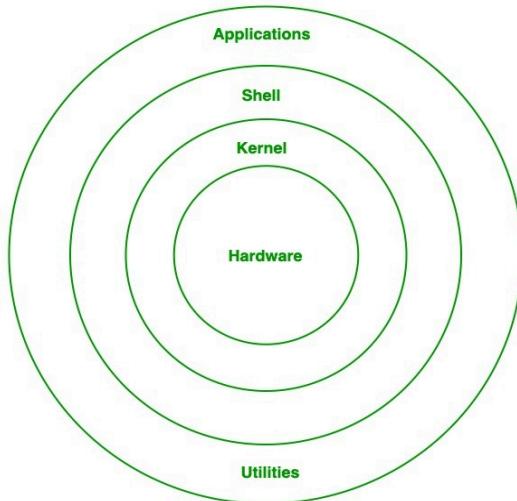
SUSE Family Systems ([openSUSE](#))

- Ex. openSUSE (Reference distribution for the SUSE family)
- RPM- based zypper package manager
- SUSE Linux Enterprise Server (SLES) is upstream for openSUSE.
- SUSE Linux Enterprise Server (SLES) has no cost to end users
- YaST (Yet Another Setup Tool) application for system administration purposes.
- SLES is widely used in retail and many other sectors

Debian Family Systems (Ubuntu, LInux Mint, Kali)

- Ex. Ubuntu > Linux Mint
- DPKG-based APT package manager (apt, apt-get, apt-cache)
- Debian distribution is upstream for several other distributions
- Common use on server, cloud deployment and desktop computers
- Community based open-source project
- Stability / Security / Popular / Long Term Support (LTS) / Easy to Use /GNOME
- Free / Open-Source / Community-driven / High reliable

Architecture of Linux



Kernel

- Core part of the operating system.
- Manages system resources and communication with hardware.
- Types: Monolithic, Microkernel, Hybrid, Exokernel.
- Provides low-level services like memory, process and device management.
- Acts as a bridge between hardware and system software.

System Library / Shared Library

- Collection of pre-written code used by programs.
- Provides standard functions (e.g., file handling, math, I/O).
- Helps in implementing OS functionality without rewriting code.
- Reduces program size by sharing common routines.

- Interfaces between application programs and the kernel.

Shell

- User interface for interacting with the OS.
- Accepts user commands and translates them for the kernel.
- Can be command-line (e.g., Bash) or graphical.
- Provides scripting capability for task automation.
- Acts as a bridge between the user and system.

Hardware Layer

- Physical components of the computer system.
- Includes CPU, RAM, storage devices, and I/O peripherals.
- Executes machine-level instructions.
- Provides the foundation on which OS and software run.
- Directly managed by the kernel for resource allocation.

System Utility

- Tools to manage and configure the computer system.
- Includes software installation, user management, and networking.
- Monitors and optimizes system performance.
- Provides security via permissions and access control.
- Helps in maintaining and troubleshooting the system.

Computer Startup: (Booting process)

- Whenever any computing device is in dummy state (without power) its Operating system remain stored in secondary storage (computer architecture – storage unit – two types – 1) Primary (Ex. RAM, ROM) 2) Secondary (hard disk, SSD) after device gets started the operating system must be in the main memory of the device and our computing device is in active state (with power)



- This task is done by the hardware and firmware (specific type of software that is embedded into hardware device to control operation/ pieces of software that get all the main parts of your PC / software that provides low-level control of computing device hardware / Types: Embedded Firmware, BIOS/UEFI Firmware etc.) in the CPU / processor in the computer system

BOOTING PROCESS:

- It is a process of initializing the system / switching on the computer (start with button press/software command) and loading the operating system from a secondary memory to the main memory, initializes its components and prepares to execute user application
- BIOS and Setup Program, The Power-On-Self-Test (POST), The OS loads, System configuration, system Utility and user authentication
- Process of returning a computer from a state of sleep does not involve booting all data gets restored by state of hibernation.

Boot Loader/ Bootstrap loader

- It is a program which resides in the MBR and it's function is to load the OS kernel into the memory
- Program with functionality to load the operating system from bootable device into main memory
- Load other software for the operating system to start
- Primary function to provide the user with a choice of operating system to boot into
- there are two type of boot loader 1) GRUB 2) LILO
- Window (Windows Boot Manager/Bootmgr.exe), Linux System (GRUB), MAC (boot.efi)

SEQUENCE OF BOOTING:

- It contains set of operations that computer performs when it is switched on

Boot Devices:

- It is the device from which the OS is loaded
 - BIOS supports booting from various devices
 - Ex. Hard disk drive / Optical drive / floppy drive / NIC / USB
-

Boot Sequence:

1) BIOS (Basic Input output system)

- BIOS is a **firmware** stored on a **ROM chip** on the motherboard
- It allows access to and setup of computer system at **most basic hardware level**
- **UEFI (Unified Extensible Firmware Interface)** is the modern replacement for BIOS.

- Firmware program to testing, initiating and detecting computer hardware component (Keyboard, Hard drives, Mouse, Screen, Processor and Memory)
- It contains **start-up programs** such as Device detection, Hardware initialization and Preparing the system to load the operating system
- It is the **first program executed** in the boot process and runs directly from **ROM**.
- **POST** (Power on Self Test: firmware checks the system hardware components and ensure that they are correctly in working condition before passing control to the boot loader / to check that device of computer is functionally stable / functionality test for Processors, Memory, Storage, keyboard, System timer)
- If **POST test gets failed** then it show error message on screen
- Check for bootable device (Pendrive / USB / CD / DVD / Network / Hard disk) and allow the user to configure a boot order in BIOS setup
- BIOS looks for a **boot sector (MBR – Master Boot Record)** at the beginning of the disk to get bootloader
- Search for boot loader to find **kernel image** on the disk and loads the operating system into **main memory (RAM)** to start system (FOUND IN MBR)
- Once control is passed to the bootloader, BIOS's role is completed.

2) MBR (Master Boot Loader)/ EFI system partition

- First sector of bootable device with size of 512 bytes
- It contains partition table (446 byte), boot loader code (64 byte) which is responsible for loading the operating system into memory and starting its execution and Error checking (2 bytes)
- boot loader is usually stored on one of the system's storage devices, such as a hard disk or SSD drive, either in the boot sector (for traditional BIOS/MBR systems) or the EFI partition (for more recent (Unified) Extensible Firmware Interface or EFI/UEFI systems).
- It contains IMP info for the boot loader to locate and load the operating system
- Once boot loader program is detected (found in MBR) then it will perform its task and further control will pass to it.

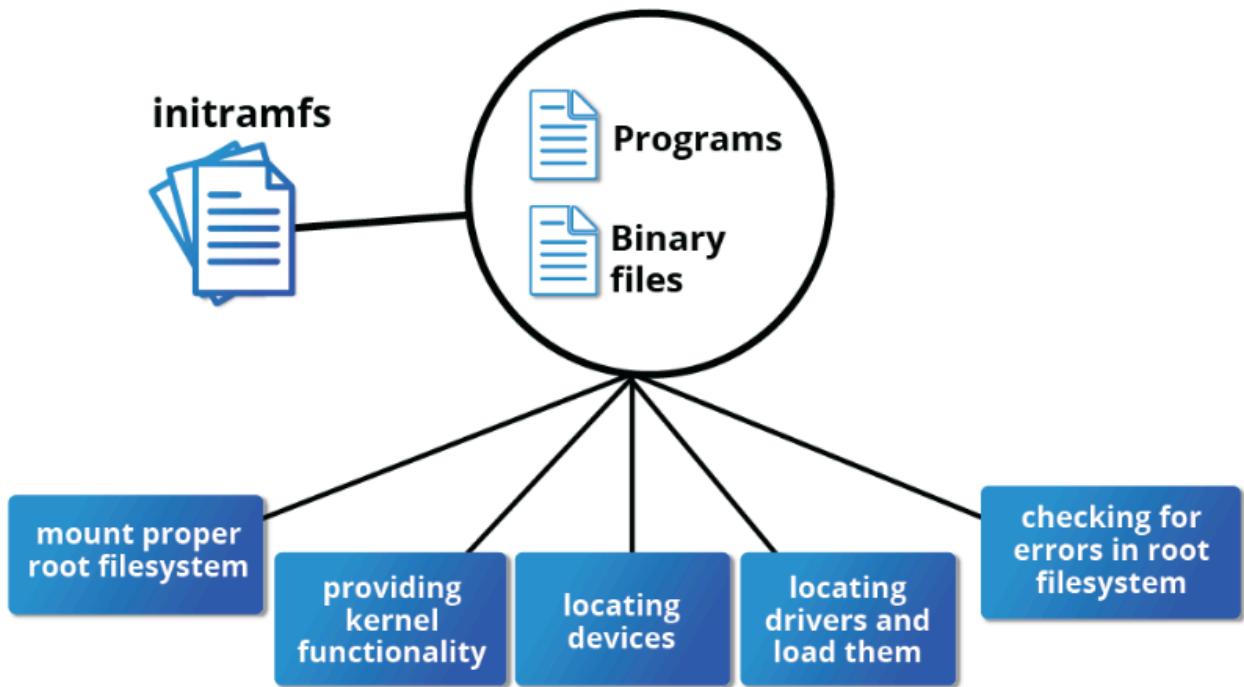
3) Boot Loader ([MORE](#))

- Linux boot loaders can present a user interface for choosing alternative options for booting Linux and even other operating systems that might be installed.
- Boot loader is responsible for loading the kernel image and the initial RAM disk or filesystem (which contains some critical files and device drivers needed to start the system) into memory.
- Displays a simple GUI menu to select available OS or kernel images (default last image is selected)
- It allows the user to choose a kernel and proceed with the booting process

- Once you selected the kernel image, it locates that kernel image
- Provides boot instructions including kernel location, root file system, and kernel parameters
- **GRUB** (Grand UNified Boot Loader), **ISOLINUX** (For booting from removable media), **DAS U-Boot** (For booting on embedded devices) are the most widely used on modern Linux system
- The grub configuration file is present at /boot/grub/grub.cfg directory
- Used to boot various operating systems windows / Linux
- The LILO configuration file is present at /etc/lilo.conf directory
- Used to boot Linux-based operating system

4) KERNEL [\(MORE\)](#)

- The boot loader loads the **kernel** along with **initrd (initial RAM Disk) / initramfs (initial RAM file system)** into memory to begin system initialization.
- The kernel starts initializing and configuring the system hardware, system memory and attached devices (Processors, I/O subsystems, storage devices)
- **initramfs** filesystem image contains a **temporary minimal filesystem** with essential programs, binaries and kernel modules.
- It provides **filesystem support** and **device drivers** required to access the real root filesystem, especially for mass storage controllers.
- The kernel uses the **udev system** from initramfs to detect available hardware devices and load the appropriate drivers dynamically.
- Using the tools in initramfs, the kernel **locates, checks, and mounts the real root filesystem** at the correct mount point.
- The **mount process** informs the OS that the root filesystem is ready and integrates it into the filesystem hierarchy.
- If special hardware drivers are required before accessing storage, they must be present in the **initramfs image**.
- After the root filesystem is successfully mounted, **initramfs is cleared from RAM** to free memory.
- Then, kernel executes **/sbin/init** from the real root filesystem.
- Control is finally passed to the **init system**, which completes initialization and continues the normal boot process



5) INIT (MORE)

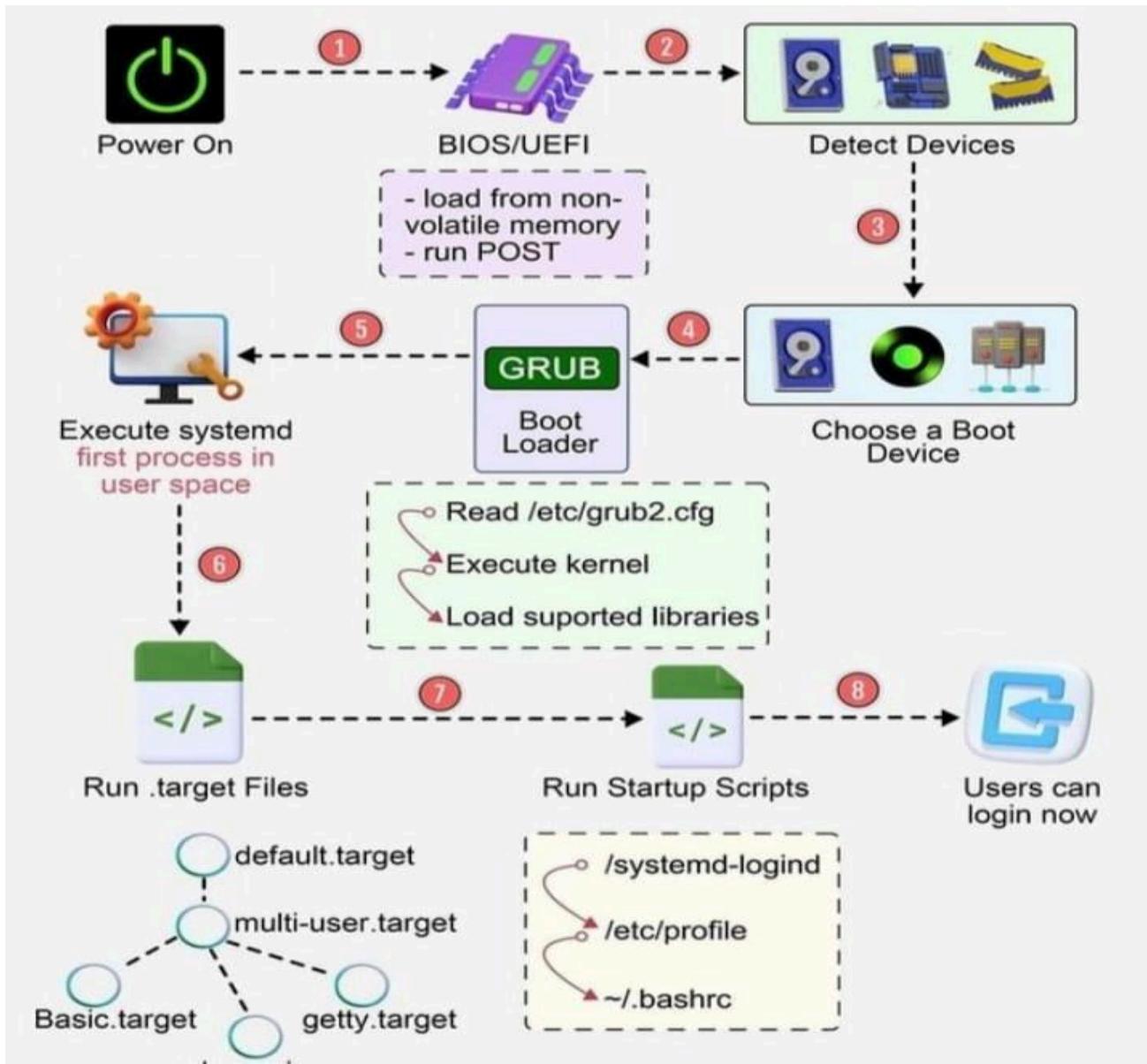
- **/sbin/init runs as Process ID (PID) 1**, making it the **parent of all user-space processes**.
- Init is responsible for **starting all required system services and processes** to bring the system to a usable state.
- Most processes in the system **trace their origin back to init**, except **kernel processes**, which are started directly by the kernel
- Init manages the **entire system lifecycle**: startup, continuous operation, and clean shutdown.
- It monitors non-kernel processes, **cleans up after terminated processes**, and **restarts services** when needed (e.g., login services).
- Init launches **text-mode or graphical login prompts** for user authentication (username/password).
- It determines the **system startup mode** (runlevel or target), which controls which services and processes are started.
- Traditionally, initialization followed a **sequential startup model (SysVinit)** using startup scripts grouped by runlevels.
- Modern systems use **event-driven and parallel startup mechanisms**, allowing faster boot times.

- The primary Linux initialization systems are **SysVinit** (traditional, sequential), **Upstart** (event-based, now deprecated) and **systemd** (modern, dominant init system)
- In systemd-based systems, **/sbin/init** is a **symbolic link** to **/lib/systemd/systemd**.
- systemd replaces complex shell scripts with **unit configuration files** that define dependencies and execution rules.
- systemd improves performance by **starting multiple services in parallel**, utilizing multi-core CPUs.
- A single command, **systemctl**, is used to manage services (start, stop, restart, enable, disable, status).
- systemd has been adopted by all major Linux distributions, simplifying system administration across platforms.

6) RUNLEVELS

- Runlevels define **different operational states** of the Linux operating system.
- Each runlevel determines **which system services and processes are started or stopped** during boot.
- Traditional SysVinit systems use **runlevels numbered from 0 to 6**.
- Each runlevel corresponds to a **specific mode of operation** (shutdown, single-user, multi-user, reboot, etc.).
- During system startup, init switches the system to the **configured default runlevel**.
- Within each runlevel, services can be configured to **start, stop, or remain inactive**.
- Modern systemd-based systems replace runlevels with **targets**, while still maintaining backward compatibility.
- Runlevels (or targets) allow flexible system configuration for **maintenance, networking, multi-user, or graphical environments**.

Run Level	Mode	Action
0	Halt	Shuts down system
1	Single-User Mode	Does not configure network interfaces, start daemons, or allow non-root logins
2	Multi-User Mode	Does not configure network interfaces or start daemons.
3	Multi-User Mode with Networking	Starts the system normally.
4	Undefined	Not used/User-definable
5	X11	As runlevel 3 + display manager(X)
6	Reboot	Reboots the system



Summary:

- The BIOS stage is the first stage of the boot process, and it involves detecting and initializing hardware devices. It then looks for the MBR on the boot device.
- The boot loader is responsible for loading the operating system kernel into memory and passing control to the kernel. GRUB is the most widely used boot loader on modern Linux systems, due to its flexibility, reliability, and compatibility.
- The kernel initialization stage involves initializing hardware devices, setting up system memory, mounting the root file system, and starting the init process.

- The init system initializes system services and processes, such as daemons, and prepares the system for user interaction and application execution.

TYPES OF BOOTING:

Based on Starting:

1) Cold booting (start/soft)

- When computer starts for the first time/starts from shut-down state and switch on the power button to start the system
- System reads all the instructions from the ROM (BIOS) and Operating system automatically gets loaded into the system
- Requires more time
- ram is clear to zero

2) Warm booting (restart/hard)

- When computer systems come to no response/hang state and then system needs restart
 - Required when install a new software/ to set software/hardware configuration changes / system is behaving abnormally / not responding well
 - less time is required
 - RAM is not clear to zero
-

Linux File System

- File system is method of storing and organizing arbitrary collections of data in a human-readable form

File System

- Method of storing and accessing files

Types of File System

- Conventional Disk Filesystem (ext3,ext4,XFS,JFS,NTFS,vfat,exfat)

- Flash Storage Filesystem (ubifs,jffs2,yaffs)
- Database Filesystem
- Special purpose Filesystem (procfs, sysfs,tmpfs,fuse)

Partitions

- Dedicated subsection of physical storage media / Physical contiguous portion of hard disk / Logical part of the disk
- It is a container having file system

	WINDOWS	LINUX
Partition	Disk1	<code>/dev/sda1</code>
Filesystem type	NTFS/VFAT	EXT3/EXT4/XFS/BTRFS...
Mounting parameters	DriveLetter	MountPoint
Base folder (where OS is stored)	C:\	/

Filesystem Hierarchy Standard (FHS)

- Linux system stores files according to standard layout called the FHS
- In Linux, everything is treated as a file (including devices and processes).
- All files and directories exist under a single root directory /.
- FHS defines the structure and purpose of each directory.
- Specifies what types of files go into which directories (e.g., /bin, /etc, /home).
- Derived from the UNIX filesystem design principles.



Standard File Streams / Descriptors

Standard Input (standard in / stdin / 0) - Keyboard

Standard Output (standard out / stdout / 1) - Terminal

Standard Error (stderr / 2) - Terminal

I/O Redirection

Graphical User Interface (GUI)

Graphical Desktop / X Window System

- Loading the graphical desktop is one of the final steps in the boot process of a linux desktop which is called as X window System / X
- **Display Manager** (Keep track of the display being provided and loads the X server / Handle graphical login and start the appropriate desktop environment after a user logs in)
- Display management, Loads Graphical Desktop (X or Wayland) and Manage graphical logins are the major role of display manager
- Ex. GNOME-gdm, KDE-kdm
- **Session Manager** (Start and maintain the components of graphical session)
- **Window manager** (Controls the placement and movement of windows, window title-bars and controls)
- Display Manager, Session Manager, Window Manager and Utilities provides a seamless desktop environment

Desktop Environment

GNOME

- Popular / Easy to Use / Default desktop environment / Menu-based navigation
- Different look for different Distro (RHEL, Fedora, CentOS, SUSE, Debian)

KDE

- Common desktop environment / Widely used
-

Command Line Interface (CLI)

Advantage

- No extra load from graphical interface (faster and lightweight).
 - Almost all tasks can be done using commands.
 - Scripts can automate repeated tasks.
 - Remote systems can be accessed easily.
 - Applications can be started directly using commands.
 - Command line works the same across all Linux distributions.
-

Aspect	CLI (Command Line Interface)	GUI (Graphical User Interface)
Definition	Text-based interface where users type commands	Visual interface using windows, icons, and menus
Usage	Requires remembering commands and syntax	Easy to use with minimal memorization
Speed	Faster for repetitive and automated tasks	Slower for repetitive tasks

Hard Link

- A **hard link** is an additional name for an existing file.

Key Points:

- Both the original file and the hard link **share the same inode number**.
- They point to the **same data on disk**.
- Deleting one name **does not delete the file data** as long as another hard link exists.
- Changes made through one name **appear in the other**.
- Hard links **cannot span different filesystems**.
- Hard links **cannot be created for directories** (to prevent filesystem loops).

Advantages:

- Saves disk space.

- File remains safe until all hard links are removed.

Disadvantages:

- Can cause confusion if one name is deleted.
- Difficult to manage across filesystems.

Soft Link / Symbolic Link / Symlinks Link

- A **soft link** is a special file that points to another file by its **path name**.

Key Points:

- Has a **different inode number** from the original file.
- Acts like a shortcut or reference.
- Can link to **directories**.
- Can span **different filesystems or disks**.
- If the original file is deleted, the link becomes a **dangling (broken) link**.

Advantages:

- Very flexible.
- Easy to move or redirect.
- Commonly used for shortcuts.

Disadvantages:

- Breaks if the target file is removed or renamed.
-

Environment Variables

- Quality that have specific values which may be utilized by the command shell or other utilities and applications

PS1 Variable

- Special shell variable that defines the primary command prompt
- PS1 (Prompt String 1)
- It controls how your **command prompt looks**
- Used mainly in **Bash shell**

Symbol	Meaning
\u	Username
\h	Hostname (short)
\H	Full hostname
\w	Current working directory (full path)
\W	Current directory name only
\\$	Shows # for root, \$ for normal user
\t	Current time (24-hour)
\d	Current date
\n	New line

Package Management Systems on Linux

- Packages contains the files and other instructions needed to make one software component work well and cooperate with the other components that comprise the entire system.
- Low-level package manager (Dpkg, RPM) used to unpack individual packages, running scripts and getting the software installed correctly
- High-level package manager (apt, dnf,zypper) used to groups of packages, downloads packages from the vendor and figures out dependency

OPERATION	RPM	DEB
Install package	<code>rpm -i foo.rpm</code>	<code>dpkg --install foo.deb</code>
Install package, dependencies	<code>dnf install foo</code>	<code>apt install foo</code>
Remove package	<code>rpm -e foo.rpm</code>	<code>dpkg --remove foo.deb</code>
Remove package, dependencies	<code>dnf remove foo</code>	<code>apt autoremove foo</code>
Update package	<code>rpm -U foo.rpm</code>	<code>dpkg --install foo.deb</code>
Update package, dependencies	<code>dnf update foo</code>	<code>apt install foo</code>
Update entire system	<code>dnf update</code>	<code>apt dist-upgrade</code>
Show all installed packages	<code>rpm -qa or dnf list installed</code>	<code>dpkg --list</code>
Get information on package	<code>rpm -qil foo</code>	<code>dpkg --listfiles foo</code>
Show packages named foo	<code>dnf list "foo"</code>	<code>apt-cache search foo</code>
Show all available packages	<code>dnf list</code>	<code>apt-cache dumpavail foo</code>
What package is file part of?	<code>rpm -qf file</code>	<code>dpkg --search file</code>

Application:

- Servers and Hosting
- Development
- Desktop and Personal Use
- Cybersecurity
- Embedded Systems
- Supercomputers

Installation:

Linux Installation on Virtual Machine

Dual-Booting:

Download ISO > Create Bootable USB > Boot from USB > Install Linux > Complete installation

STEP 1: Disable secure boot (disable** / **enable**)**

Right click on Window icon > setting > system > recovery > advance startup > (**BLUE SCR**) > Troubleshoot > advance option > UEFI Firmware setting > Restart > (BIOS) > Secure Boot > disable > F10 (Save) > Restart > bitlocker key > (if there is a problem then move to BIOS and enable key will enter into the window) > enter bitlocker key > enter into window

STEP 2: Turn off bitlocker

Open Powershell > CMD: manage-bde -status > Other CMD (Chatgpt) / Click on GUI

STEP 3: Create a partition (Disk Partition)

Right click on Window icon > Disk management > Choose drive > right click > shrink volume > give space in mb > shrink > (X GB unallocated space created > right click > new simple volume > next > next >> Vol label (rename)

STEP 4: Create bootable device

Browser > ubuntu download > download ubuntu desktop > Browser > Rufus > rufus 4.4.exe > coonect pendive (8GB) > open rufus > yes > boot selection (select > ubuntu) > choose distro > start > ok > yes > warning > ok

STEP 5: Boot with bootable device

Right click on Window icon > setting > system > recovery > advance startup > (**BLUE SCR**) > use a device > choose USB device > click > GRUB > try and install ubuntu > ubuntu installation page > select language> INSTALL UBUNTU > continue > minimal installation > continue > **option one, two, three** >install now > continue > add credential > install > restart >

GRUB Customization:

Platform: Gnome.org

Download > Extract > cd to that directory

sudo mkdir -p /boot/grub/themes
sudo cp -r /path/to/your/theme /boot/grub/themes/
sudo nano /etc/default/grub
GRUB_THEME="/boot/grub/themes/your-theme-directory/theme.txt"
sudo update-grub
sudo reboot

GNOME-tweaks

Platform: [Gnome.org](https://gnome.org)

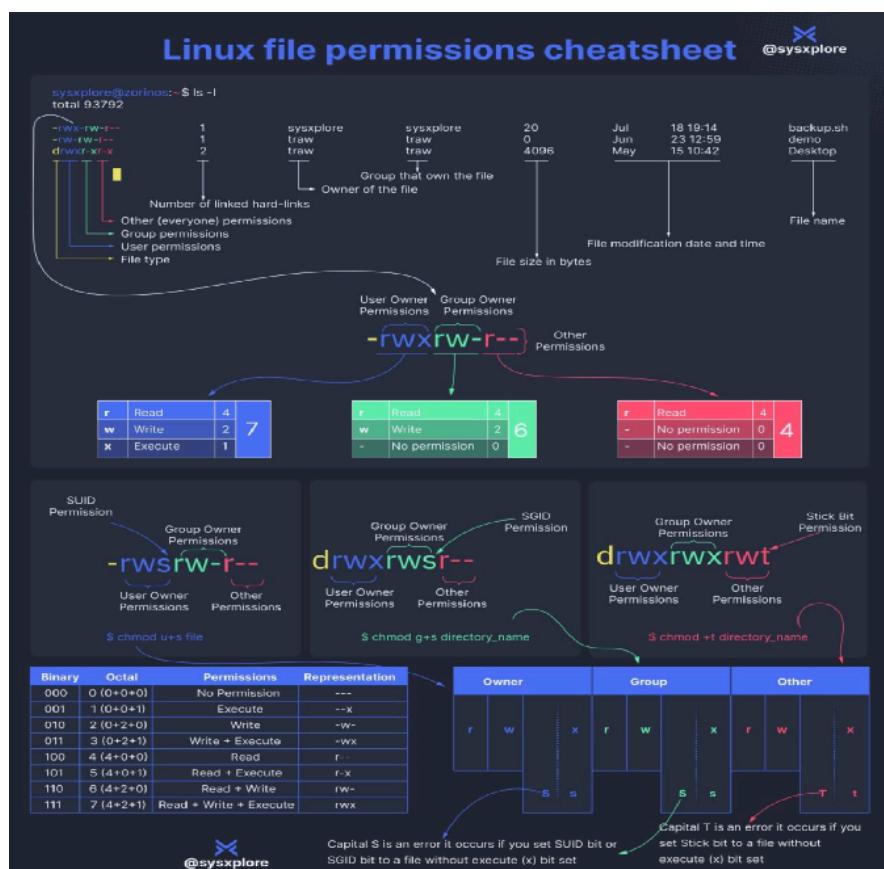
```
sudo apt install gnome-tweaks  
gnome-tweaks  
sudo apt install gnome-shell-extension-manager
```

FILE Permission:

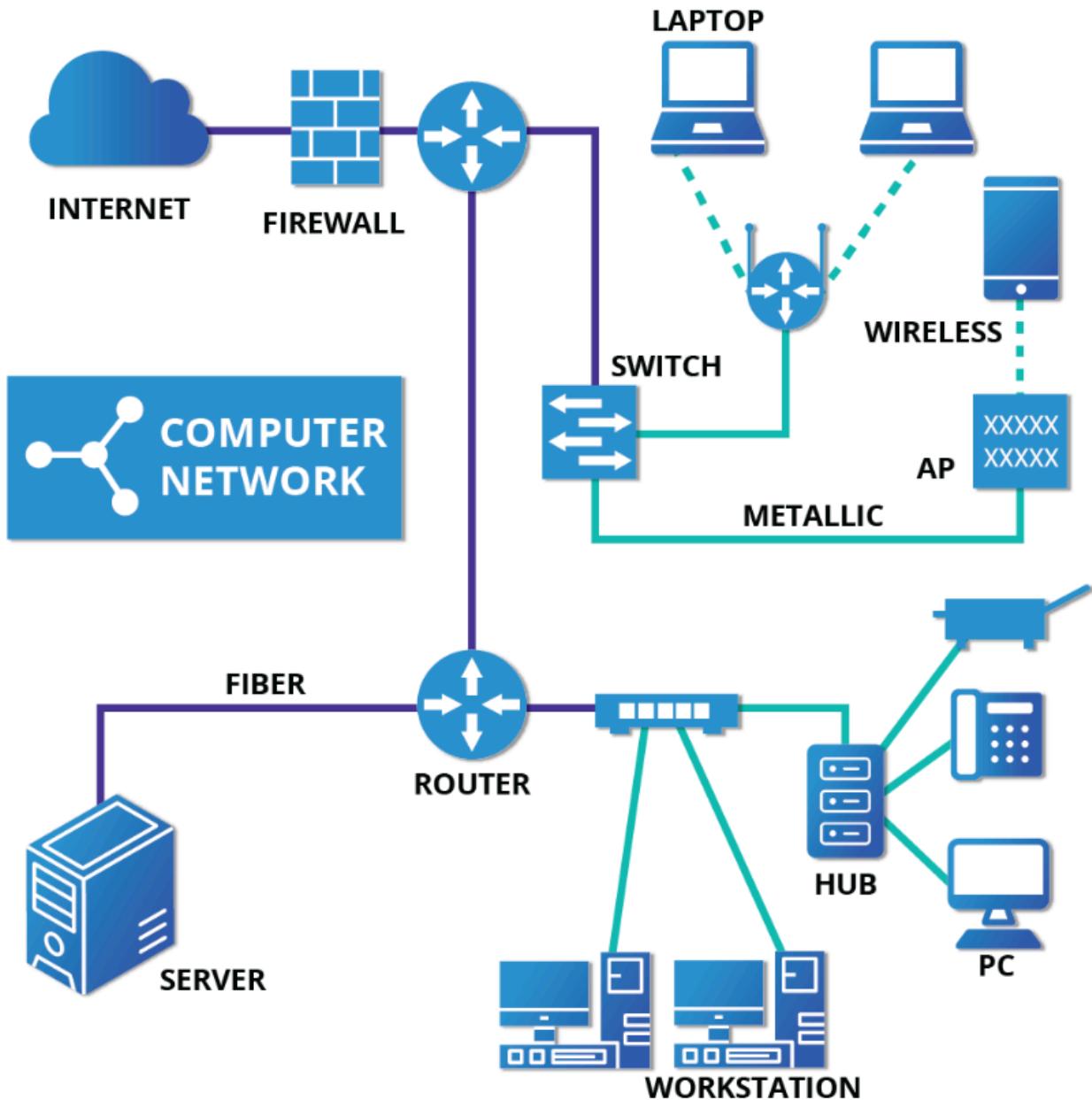
- Ask for something
- Data security / user control / Access restriction / System integrity (Need)
- r / w / x (Type)
- **Read:** Read the content from folder **Write:** Write content **Execute:** Run

ls -l

(folder /file) (user) (group) (other) code owner group size modification date (file / folder name)
folder: d socket file: s link file: l
file: -
code: number of hard links to the directory.
user: u
group: g
other: o



Linux and Networking



Network Manager

- Network Manager establish your connections and keep track of your settings
- Used to list all available network (wired and wireless) and allow the choice of a wired, wireless and mobile broadband network, handle passwords and setup VPNs

Wired connections (Ethernet):

When you connect a network cable to your computer, it usually works automatically. The computer detects the cable and the network by itself. Network Manager then gets the required network settings automatically using **DHCP**, so you don't have to do anything.

If the network does **not** use DHCP, you can still set the network details manually using Network Manager. You can also change the **MAC address** if your network card allows it. The MAC address is a unique number that identifies your network card.

Wireless connections (Wi-Fi):

Wi-Fi does not connect automatically in most cases. You need to choose a wireless network from the list of available networks. Network Manager shows all nearby Wi-Fi networks and also shows which one you are currently connected to.

You can save Wi-Fi networks, edit their settings, delete them, and choose which network should connect automatically when it is available.

Protocol

Network Time Protocol (NTP)

Dynamic Host Configuration Protocol (DHCP)

Internet Message Access Protocol (IMAP)

Post Office Protocol (POP)

File Transfer Protocol (FTP)

User Management

- Core function of Linux system administration
- Control system access, Ensure security and privileges for task
- Used to create multi-user environment
- Linux system supports up to 60000 users
- UID 0 (Super user), UID 1-999 (System accounts), UID 1000+ (Regular Users)

Key Benefits

- Secures the system from unauthorized access
- Ensures users can perform their roles without interfering with others
- Helps in auditing and tracking user activity

User Type	Description
Root (Superuser)	Full system control. Can install software, change config files, and delete anything. Powerful but risky.
Regular User	Limited access. Can create files, run applications, but not modify system-level settings.
Sudo User	Regular user with temporary admin rights via the <code>sudo</code> command. Common in modern systems.
System/Service Account	Non-human accounts used by services (e.g., <code>mysql</code> , <code>nginx</code>). Limited privileges.
Guest User	Temporary users with minimal privileges. Changes are not saved after logout.

User Groups

- It is a collection of users
- Used to manage file and system permission for many users at once

Types of User Groups

Primary Group (Default)

- Every Linux user is assigned one primary group.
- By default, this group usually has the same name as the user.
- It helps manage file ownership cleanly without much extra configuration

Secondary Group

- A user can be a part of multiple secondary groups.
- These groups provide extra access to files, folders, or services.
- They are commonly used for team-based access or system-level permissions (e.g., accessing Docker, video devices, or running sudo).

Text Editor:

Text editors are software applications used to create, view, and edit plain text files. They are an essential tool for programming, writing, and editing configuration files. Here's a simple explanation of text editors, including their types and features

What is a Text Editor?

A text editor is a program that allows you to write and modify text. Unlike word processors (such as Microsoft Word), text editors do not typically include advanced formatting options like bold or italic text. Instead, they focus on plain text, which is useful for coding and scripting.

Types of Text Editors

1. Basic Text Editors:

Examples: Notepad (Windows),TextEdit (macOS in plain text mode), Nano (Linux).

Features: Simple interface, basic text editing functionalities (cut, copy, paste, find, replace).

2. Advanced Text Editors:

Examples: Sublime Text, Visual Studio Code, Atom, Notepad++, Vim, Emacs.

Features: Syntax highlighting, code completion, plugins/extensions, version control integration, multiple cursors, and more.

Key Features of Text Editors

1. Syntax Highlighting:

What it is: Color-coding of text according to the syntax of the programming language being used.

Benefit: Makes code easier to read and understand.

2. Code Completion:

What it is: Suggests possible completions for partially typed words or phrases.

Benefit: Speeds up coding and reduces typing errors.

3. Plugins and Extensions:

What it is: Additional tools or functionalities that can be added to the editor.

Benefit: Enhances the capabilities of the editor, such as adding support for more languages, themes, or tools like linters and debuggers.

4. Version Control Integration:

What it is: Integration with version control systems like Git.

Benefit: Allows you to manage code changes and collaborate with others directly within the editor.

5. Search and Replace:

What it is: Tools to find specific text within files and replace it if needed.

Benefit: Saves time when making widespread changes to a document or codebase.

6. Multi-Cursor and Multi-Selection:

What it is: Allows editing of multiple lines or sections of text simultaneously.

Benefit: Increases efficiency when making repetitive changes.

7. Customization:

What it is: Ability to customize the look and behavior of the editor (themes, keybindings).

Benefit: Makes the editor more comfortable and efficient to use according to personal preferences.

1. Atom

Platform: Windows, macOS, Linux

Features:

- Hackable: Known as a "hackable text editor for the 21st century," Atom is highly customizable and extendable.
- Built-in Package Manager: Easily search for and install new packages or create your own.
- Smart Autocompletion: Provides suggestions as you type, helping speed up coding.
- File System Browser: Browse and open files and projects in one window.
- Multiple Panes: Split your interface into multiple panes to compare and edit code across files.
- Find and Replace: Powerful search and replace feature within files, across projects.
- Integration with Git and GitHub: Seamless integration with Git version control and GitHub.

2. Sublime Text

Platform: Windows, macOS, Linux

Features:

- Performance: Fast and responsive, even with large files.
- Multiple Selections: Edit multiple lines or instances of text at once.
- Command Palette: Access functions via a searchable palette.
- Split Editing: View and edit multiple files or locations within the same file side-by-side.
- Goto Anything: Quickly navigate to files, symbols, or lines.
- Package Control: Easily manage plugins and extensions to enhance functionality

Installation:

- 1) subl --version
- 2) sudo apt update
- 3) sudo apt install sublime-text
- 4) sudo apt remove sublime-text

3. Vim

Platform: Windows, macOS, Linux (widely used on Linux)

Features:

- Keyboard-Centric: Efficient editing with minimal mouse use, relying heavily on keyboard shortcuts.
- Modes: Different modes for inserting text, editing, and command execution (e.g., normal mode, insert mode).
- Customization: Highly configurable through .vimrc and plugins.
- Remote Editing: Excellent for remote work via terminal over SSH.
- Lightweight: Low resource usage, making it suitable for various environments, including servers.
- Extensibility: Large ecosystem of plugins for additional functionalities.

Command	Description
vim --version	
sudo apt install vim	
sudo apt remove vim	
Vim	About vim
vimtutor	About tutorial
Open Editor and Modes	
vim file_name	Start the editor and edit file_name
i	Enter to Insert Mode from Command Mode
:	Enter to Line Mode from Command Mode
esc	Exit from Insert Mode and Enter to Command Mode Exit from Line MOde and ENter to Command Mode
Basic Navigation	
vim -r file_name	Start and edit myfile in recovery mode from a system crash
:r file_name	Read in file and insert at current position
:w	Write to the file
:w file_name	Write out to file
:w! file_name	Overwrite file
:wq / :x	Save and exit
cat / open file_name	To show content of file
:q	Quit
:q!	Exit without save
Changing Cursor Position	
Shift g	To move bottom
gg	To move top
Searching for Text in vi	
/ word	Search from start
? word	Search from last
:%s/word1/word2/g	Replace word
U	undo
Ctrl + R	redo

4. Visual Studio Code (VS Code)

Platform: Windows, macOS, Linux

Features:

- Extensible: Rich extension marketplace for additional functionalities.
- Integrated Terminal: Access a command-line interface directly within the editor.
- Git Integration: Built-in support for version control with Git.
- IntelliSense: Advanced code completion and error checking.
- Debugging: Powerful debugging capabilities.
- Customization: Highly customizable with themes, keybindings, and settings.

5. Nano

Platform : Linux , macOS,Windows

Features:

User-Friendly Interface:

Simple and Intuitive:

- Nano offers a straightforward and easy-to-use interface, making it accessible to new users.

On-Screen Shortcuts:

- All essential commands are displayed at the bottom of the screen, so users don't need to memorize complex key bindings.

Basic Text Editing:

- **Basic Operations:** Nano supports basic text editing operations like cut, copy, paste, and delete.
- **Line Numbering:** It can display line numbers for easier navigation and editing.

Search and Replace:

- Nano includes a simple search and replace functionality, which can be used to find and modify text in the document.

Syntax Highlighting:

- It offers syntax highlighting for various programming languages and file types, making it easier to read and write code.

Undo/Redo:

- Nano supports multiple levels of undo and redo, allowing users to revert or reapply changes easily.

Configurable Settings:

- **Customizable Key Bindings:** Users can customize key bindings to suit their preferences.
- **Configuration Files:** Nano can be configured using .nanorc files, where users can set preferences like tab size, syntax highlighting, etc.

File Management:

- **File Browser:** Nano includes a simple file browser for opening and saving files.
- **Backup Files:** It can automatically create backup files, which is helpful in case of accidental data loss.

Internationalization and Localization:

- Nano supports multiple languages, making it accessible to a broader audience.

Lightweight and Fast:

- It has a small footprint and is fast, making it suitable for use on systems with limited resources.

Command	Description
Nano –version	
sudo apt install nano	
sudo apt remove nano	
Nano file_name	Create file and write into it
Ctrl G	Display the help screen
Ctrl X	Save permission > give file name > Exit
Ctrl O	Write to a file
Ctrl R	Insert contents from another file to the current buffer
Ctrl C	Show cursor position
Ctrl K	Cut
Ctrl U	Paste
Ctrl W	Search the word
Ctrl T	Spell Check

6. gedit

Platform : Linux, macOS, Windows

7. emacs

Platform : Linux, macOS, Windows

About:

Command	Description
emacs file_name	Create file and write into it
Ctrl-x i	Insert prompted for file at current position
Ctrl-x s	Save all files
Ctrl-x Ctrl-w	Write to a file giving a new name when prompted
Ctrl-x Ctrl-s	Saves the current file
Ctrl-x Ctrl-c	Exit after being prompted to save ant modified field

Course

- [INTRODUCTION TO LINUX \(LFS101\)](#)
 - [GFG](#)
-

Project

- Create users and set permissions
 - Schedule backups using cron
 - Archive logs automatically
 - Host a basic web page using Nginx
 - Secure the system with firewall rules
 - Review key commands from all days
 - Final system cleanup and optimization techniques
-

Documentation:

Linux
Ubuntu
Commands (man,help)
GNU Info
Other ([Gentoo Handbook](#), [Ubuntu Documentation](#), [Fedora Documentation](#))
[Document](#)

Extension

Dock to Dash
Arc Menu
Extension List
Add to Desktop
Forge
GNOME Fuzzy app search

Desktop Icon NG
Removable Driver Menu
Custom window

Important Links

01 Course Introduction

[Course](#)
[Class Forum](#)
[Support](#)

02 Linux Foundation

[Linux Foundation](#)
[Events](#)
[Education](#)
[Linux Distro](#)

03 Linux Philosophy and Concepts

[Installation](#)
[Kernel](#)

04 Linux Basic and System Startup

[FHS](#)

05 Graphical Desktop

[Wayland](#)
[GNOME](#)
[TRY IT 1](#)
[TRY IT 2](#)

07 Common Applications

[libreoffice](#)

08 Command Line Operation

[TRY IT 1](#)
[TRY IT 2](#)
[TRY IT 3](#)
[TRY IT 4](#)

09 Finding Linux Documentation

[Document](#)
[Man Pages](#)
[More](#)

11 File Operations

[Document](#)

[NFS](#)
[TRY_IT_1](#)
[TRY_IT_2](#)

12 Text editor

[VScode](#)

13 User Environment

[TRY_IT_1](#)

14 Manipulating Text

[TRY_IT_1](#)
[TRY_IT_2](#)
[TRY_IT_3](#)
[TRY_IT_4](#)
[TRY_IT_5](#)
[TRY_IT_6](#)

15 Network Operations

[Wikimedia](#)
[Octets](#)
[TRY_IT_1](#)
[TRY_IT_2](#)
[TRY_IT_3](#)
[TRY_IT_4](#)

17 More On Bash Shell Scripting

[FIPS 140-2](#)

18 Printing

[CUPS](#)
[GitLab](#)
[TRY_IT_1](#)
[TRY_IT_2](#)

19 Local Security Principles

[John The Ripper](#)
[GRUB2 Password Protection](#)

20 Course Completion

[FeedBack](#)

LAB

03 Linux Philosophy and Concepts
[Lab_3.1](#)

04 Linux Basic and System Startup

[Lab_4.1](#)

05 Graphical Desktop

[Lab_5.1](#)

[Lab_5.2](#)

[Lab_5.3](#)

06 System Configuration from the Graphical Interface

[Lab_6.1](#)

[Lab_6.2](#)

[Lab_6.3](#)

[Lab_6.4](#)

07 Common Applications

[Lab_7.1](#)

08 Command Line Operation

[Lab_8.1](#)

[Lab_8.2](#)

[Lab_8.3](#)

[Lab_8.4](#)

[Lab_8.5](#)

09 Finding Linux Documentation

[Lab_9.1](#)

[Lab_9.2](#)

[Lab_9.3](#)

[Lab_9.4](#)

10 Processes

[Lab_10.1](#)

[Lab_10.2](#)

[Lab_10.3](#)

[Lab_10.4](#)

11 File Operations

[Lab_11.1](#)

[Lab_11.2](#)

[Lab_11.3](#)

12 Text editor

[Lab_12.1](#)

[Lab_12.2](#)

[Lab_12.3](#)

[Lab_12.4](#)

13 User Environment

[Lab_13.1](#)

[Lab_13.2](#)

[Lab_13.3](#)

[Lab_13.4](#)

14 Manipulating Text

[Lab_14.1](#)

[Lab_14.2](#)

[Lab_14.3](#)

[Lab_14.4](#)

[Lab_14.5](#)

15 Network Operations

[Lab_15.1](#)

[Lab_15.2](#)

16 The Bash Shell And Basic Scripting

[Lab_16.1](#)

[Lab_16.2](#)

[Lab_16.3](#)

[Lab_16.4](#)

[Lab_16.5](#)

[Lab_16.6](#)

17 More On Bash Shell Scripting

[Lab_17.1](#)

[Lab_17.2](#)

[Lab_17.3](#)

18 Printing

[Lab_18.1](#)

[Lab_18.2](#)

19 Local Security Principles

[Lab_19.1](#)

[Lab_19.2](#)

QUIZ

02 Linux Foundation

[Chap_2](#)

03 Linux Philosophy and Concepts

[Chap_3](#)

04 Linux Basic and System Startup

[Chap_4](#)

05 Graphical Desktop

[Chap_5](#)

06 System Configuration from the Graphical Interface[Chap_6](#)**07 Common Applications**[Chap_7](#)**08 Command Line Operation**[Chap_8](#)**09 Finding Linux Documentation**[Chap_9](#)**10 Processes**[Chap_10](#)**11 File Operations**[Chap_11](#)**12 Text Editor**[Chap_12](#)**13 User Environment**[Chap_13](#)**14 Manipulating Text**[Chap_14](#)**15 Network Operations**[Chap_15](#)**16 The Bash Shell And Basic Scripting**[Chap_16](#)**17 More On Bash Shell Scripting**[Chap_17](#)**18 Printing**[Chap_18](#)**19 Local Security Principles**[Chap_19](#)**20 Course Completion**[Chap_20](#)