LINUX FOR DEVOPS & Cloud

Complete Study Guide

Date: 19 February 2026

LinkedIn Profile: *https://www.linkedin.com/in/sumit-deshmukh-3b781527a/*

1. What is Linux?

Linux is an Operating System (OS) — like Windows or macOS. But unlike them, Linux is open-source, meaning it's free and its code is publicly available.
Linux is used in servers, cloud computing, mobile (Android), embedded systems, and DevOps tools.

Basic Components of Linux OS

Component	Explanation
Kernel	The core part of Linux that talks to hardware (like CPU, RAM, disks).
Shell	A program that lets you interact with the OS via commands.
File System	Organizes and stores data. Everything in Linux is treated as a file (even devices).
GUI	Optional — Linux can have graphical interfaces like GNOME or KDE, but often used via terminal.

2. Linux Directory Structure

Linux starts with root (/) directory. Below it, there are folders with fixed purposes:

Directory	Purpose
/	Root of the file system
/home	User folders (like C:\Users)
/bin	Essential command binaries (like ls, cp, mv)
/etc	System config files
/var	Variable files (like logs)

/tmp	Temporary files
/usr	User programs and libraries
/root	Home directory of the root (admin) user
/dev	Device files (like USB, hard disks)
/proc	System and process info (virtual folder)

3. Basic Linux Commands

Command	Use
pwd	Show current directory
ls	List files and folders
cd	Change directory
mkdir	Make a directory
touch	Create an empty file
cp	Copy files/folders
mv	Move or rename files
rm	Delete files
cat	Show file contents
clear	Clear the terminal
exit	Close the terminal or shell

4. User and Permissions

Linux is multi-user. Every file and folder has permissions and ownership.

Users

- **root: Superuser (admin)**
- Normal users: Limited access

Permission Types

Each file has 3 types of permissions for:

- Owner
- Group
- Others

Permission types:

- r → read
- w → write
- x → execute

Example Output

```
-rwxr-xr-- 1 user group 1234 Jul 1 file.sh
```

Breakdown: -rwxr-xr--

Part	Meaning
rwx (owner)	Full permission: read, write, execute
r-x (group)	Read & execute only
r-- (others)	Read only

5. Package Management (Installing Software)

- Ubuntu/Debian-based: apt-get install <package-name>
- Red Hat/CentOS: yum install <package-name> or dnf install
- Others: May use pacman, zypper, snap, flatpak

6. Processes and System Monitoring

Command	Description
ps	Show running processes
top	Live view of system usage (CPU, memory)
kill	Stop a process
htop	Better version of top (if installed)

7. File Compression and Archiving

Command	Description
tar -cvf	Create archive
tar -xvf	Extract archive
gzip, gunzip	Compress and decompress files

8. Important Concepts

Concept	Meaning
Shell Script	A file with commands you can execute like a program (.sh)
Cron Job	Schedule a task to run automatically
Log files	Located in /var/log, useful for troubleshooting
SSH	Remote login to a Linux machine (ssh user@ip)
Sudo	Temporarily run commands as root user (sudo <command>)

9. How Linux Helps in DevOps

Linux is the foundation of:

- Cloud servers (AWS EC2, GCP, Azure)
- CI/CD pipelines
- Docker & Kubernetes
- Automation tools like Ansible, Terraform

10. CRUD Operations in Linux

CRUD stands for: Create, Read, Update, Delete
In Linux, CRUD operations can be performed on: Files, Directories, File content, System records (like user data or processes)

C – CREATE Operations

Create a File

```
- touch filename.txt
```

Create a File with Content

```
- echo "Hello, Linux!" > hello.txt
- cat > data.txt
```

Type your content, then Press Ctrl+D to save **Create a Directory (Folder)**

```
- mkdir foldername
- mkdir projects
```

Create Nested Directories

```
- mkdir -p devops/scripts/yaml
```

R – READ Operations

View File Content

```
- cat filename.txt
- less filename.txt   # Scrollable
- more filename.txt   # Page-wise output
```

View First or Last Lines

```
- head filename.txt   # First 10 lines
- tail filename.txt   # Last 10 lines
```

Read a Directory

```
- ls         # List files
- ls -l      # Detailed view
- ls -a      # Show hidden files
```

Read a Specific Line

```
- sed -n '3p' filename.txt   # Show 3rd line
- awk 'NR==3' filename.txt   # Another method
```

U – UPDATE Operations

Append to File

```
- echo "New line" >> file.txt
```

Edit File (Using Editors)

- nano filename.txt – beginner-friendly terminal editor
- vim filename.txt – powerful but advanced editor **Replace Text in File**

```
- sed -i 's/oldtext/newtext/g' filename.txt
- sed -i 's/Devops/Linux DevOps/g' notes.txt
```

Rename File or Folder

```
- mv oldname.txt newname.txt
```

Move File to Different Location

```
- mv file.txt /home/user/docs/
```

D – DELETE Operations

Delete a File

```
- rm filename.txt
```

Delete a Directory

```
- rm -r foldername
```

Force Delete

```
- rm -rf foldername
```

⚠ **Be careful with rm -rf / — it can wipe out the entire OS.**

Real-Time DevOps Use Cases of CRUD in Linux

Task	Linux Command
Create a config file	touch nginx.conf
Read server logs	less /var/log/syslog
Update YAML for deployment	nano deployment.yaml
Delete temporary files	rm -rf /tmp/*
Move backup	mv backup.tar.gz /mnt/backup/

Sample Exercise to Practice

```
# Step 1: Create a directory and file
- mkdir devops_practice
- cd devops_practice
- touch inventory.txt
# Step 2: Add data
- echo "Server1: 192.168.1.1" >> inventory.txt
- echo "Server2: 192.168.1.2" >> inventory.txt
# Step 3: Read and edit
- cat inventory.txt
- sed -i 's/192.168.1.2/10.0.0.2/' inventory.txt
# Step 4: Delete the file
- rm inventory.txt
```

11. VIM Editor in Linux

What is Vim?

Vim (Vi IMproved) is a modal text editor in Linux. Used for editing config files, shell scripts, YAML, Docker, Terraform, and more — especially in headless servers and DevOps workflows. Runs in the terminal and is lightweight, powerful, and scriptable.

Vim Has 3 Primary Modes

Mode Name	Purpose	Enter Mode	Exit Mode
1. Normal Mode	Default mode — for navigation, deleting, copying	Open Vim or press Esc	Press i, v, or :
2. Insert Mode	Typing text (like in Notepad)	Press i, I, a, A, o, O	Press Esc
3. Command-Line Mode	Save, exit, search, run commands	Press : from Normal Mode	Press Enter or Esc

Normal Mode – The Control Hub

Action	Key
Move cursor	h (left), l (right), j (down), k (up)
Word jump	w (next word), b (previous word)
Start of line	0
End of line	$
Top of file	gg
Bottom of file	G
Delete line	dd
Copy (yank) line	yy
Paste	p
Undo	u
Redo	Ctrl + r

Insert Mode – For Writing or Editing Text

Key	Function
i	Insert before cursor
I	Insert at start of line

a	Insert after cursor
A	Insert at end of line
o	Open a new line below
O	Open a new line above

Once in insert mode, you type normally like any text editor. Press Esc to go back to Normal Mode.

Command-Line Mode – For Save, Quit, Search

From Normal Mode, press : (colon) to enter Command Mode.

Command	Meaning
:w	Save file
:q	Quit
:wq or ZZ	Save and quit
:q!	Force quit without saving
:x	Save and exit (if changes made)
:set nu	Show line numbers
:set nonu	Hide line numbers

Editing Operations in Vim (DevOps Oriented)

File & Config Editing

- vim /etc/nginx/nginx.conf

- Navigate to the line using arrow keys or j/k.
- Press i to enter insert mode.
- Make changes.
- Press Esc, then type :wq to save and exit.

Searching and Replacing

- /nginx

Then press n (next match) or N (previous match)

```
:%s/oldtext/newtext/g  # Replace all occurrences
```

Copy, Cut, Paste (Yank, Delete, Put)

Action	Key
Copy (line)	yy
Copy (3 lines)	3yy
Delete (cut) line	dd
Delete 5 lines	5dd
Paste below	p
Paste above	P

Line and Block Editing

- Visual Mode (v): Select characters/words
- Visual Line Mode (V): Select entire lines
- Visual Block Mode (Ctrl + v): Select columns Then you can press y, d, or p to copy, cut, or paste.

Practical DevOps Examples

Task	Vim Command
Edit crontab	crontab -e (uses Vim by default)
Change pod name in YAML	vim pod.yaml, /name, i, edit
Format Terraform code	vim main.tf, use =G to auto-indent
Edit Dockerfile	vim Dockerfile
Restart NGINX config	Edit nginx.conf then sudo systemctl restart nginx

Vim Cheat Sheet Summary

Mode	Enter	Purpose	Exit
Normal	Open Vim / Esc	Navigation, editing	i, :, v
Insert	i, a, o, etc.	Typing text	Esc
Command	:	Save, quit, search	Enter or Esc

12. Linux File & Directory Permissions

Why Permissions Matter?

Linux is a multi-user OS, and permissions protect files, scripts, and system settings from unauthorized access or modification. Common in DevOps, cloud environments, and CI/CD pipelines where secure access is critical.

User Categories

Symbol	User Type
u	User (Owner)
g	Group
o	Others (Everyone else)

Permission Types

Symbol	Permission	Meaning
r	Read	View content
w	Write	Modify content
x	Execute	Run file (scripts, binaries) or enter directory

Example Output from ls -l

```
-rwxr-xr-- 1 devops team 2048 Jul 1 deploy.sh
```

Part	Meaning
-	File type (- = file, d = directory)
rwx	Owner: read, write, execute
r-x	Group: read, execute
r--	Others: read only
devops	Owner username
team	Group name

Numeric (Octal) Permissions

Permission	Binary	Octal
r	100	4
w	010	2
x	001	1

rwx combo	Value
rwx	7
rw-	6
r--	4
r-x	5

chmod – Change Permissions

Symbolic Method

```
- chmod u+x script.sh    # Add execute to user
- chmod g-w file.txt     # Remove write from group
- chmod o=r file.txt     # Others: read-only
```

Numeric (Octal) Method

```
- chmod 644 file.txt     # rw-r--r--
- chmod 755 script.sh    # rwxr-xr-x
- chmod 700 secrets.txt  # rwx------ (only owner can access)
```

chown – Change Ownership

```
- chown user:group file
- chown devops:team deploy.sh
```

Permissions for Directories

Permission	Effect
r	Can list files (ls)
w	Can add/remove files
x	Can enter directory (cd)

Special Permissions

Symbol	Name	Use
s	Setuid / Setgid	Run program with owner/group privileges
t	Sticky Bit	Used in /tmp so only file owner can delete their files

```
chmod +t /shared/folder
```

DevOps Use Cases – Permissions

Task	Related Permission
Make shell script executable	chmod +x script.sh
Restrict .env to user only	chmod 600 .env
Secure private keys	chmod 400 key.pem
Allow group to deploy	chown :deployers deploy.sh
Lock config file	chmod 444 config.yml (read-only)

Permissions Summary Table

Octal	Symbolic	Meaning
777	rwxrwxrwx	Everyone full access
755	rwxr-xr-x	Owner full, others can read/execute
700	rwx------	Only owner access
644	rw-r--r--	Read/write owner, read others
600	rw-------	Only owner can read/write

13. Linux User Management

Why Is User Management Important?

In a multi-user system (like Linux servers or cloud environments), managing users, groups, and permissions ensures: Security & isolation, Controlled access to files/scripts/services, Auditing and accountability.

Types of Users

User Type	Description
Root	Superuser with all permissions
System Users	Created by services (e.g., nginx, mysql)
Regular Users	Created by admins for login and tasks

Create, Delete, and Manage Users

Create User

```
- sudo adduser riyas
```

- Adds user with home directory: /home/riyas
- Prompts to set password

```
- sudo useradd -m riyas
- sudo passwd riyas
```

Delete User

```
- sudo deluser awais          # Keeps home dir
- sudo deluser --remove-home awais
- sudo userdel -r awais
```

Modify User

```
- sudo usermod -aG devops awais      # Add to 'devops' group
- sudo usermod -l newname oldname    # Rename user
- sudo usermod -d /new/home/path awais  # Change home dir
```

Key User Info Files

File	Purpose
/etc/passwd	User account info (username, UID, GID, shell)
/etc/shadow	Encrypted passwords
/etc/group	Group definitions
/etc/sudoers	Who can use sudo

View User Info

```
- cat /etc/passwd | grep awais
- id awais       # Show UID, GID, groups
- groups awais   # Show group memberships
```

Group Management

- sudo groupadd devops
- sudo usermod -aG devops riyas
- sudo gpasswd -d awais devops
- sudo groupdel devops

Sudo (Superuser Do) Access

Grant Sudo Access

- sudo usermod -aG sudo riyas
- sudo visudo
- awais ALL=(ALL:ALL) ALL

Limited Sudo (for security)

- awais ALL=(ALL) NOPASSWD: /bin/systemctl restart nginx

Lock and Unlock Users

- sudo usermod -L awais # Lock Account
- sudo usermod -U awais # Unlock Account
- sudo usermod -s /usr/sbin/nologin awais # Disable Login Shell

Real DevOps Use Cases – User Management

Task	Command
Create a user for Jenkins agent	adduser jenkins_agent
Restrict a user to deploy only	usermod -aG docker deployer
Setup SSH-only user	Create user, disable password, set up SSH key
Group-level permission for /var/www	Create webdev group, set directory group ownership
Give NGINX team restart permission only	sudo visudo + command restriction
SFTP-only access	Create chrooted SFTP users with nologin shell

User Management Cheat Sheet

Task	Command
Add user	adduser <name>
Delete user	deluser <name> or userdel -r

Add to group	usermod -aG <group> <user>
List groups	groups <user>
Create group	groupadd <group>
Lock account	usermod -L <user>
Give sudo access	usermod -aG sudo <user>

14. Linux Process Management

What Is a Process?

A process is any program or command that is running on your Linux system. It has a unique PID (Process ID) and can be running, sleeping, stopped, or zombie.

Process States

State	Meaning
R	Running
S	Sleeping (idle but waiting)
T	Stopped
Z	Zombie (terminated but not cleaned up)
D	Uninterruptible sleep (I/O wait)

View Running Processes

- ps -ef

Shows user, PID, parent PID (PPID), time, and command

- top

Interactive, live view of CPU, memory, and processes

- htop
- sudo apt install htop # Debian/Ubuntu
- sudo yum install htop # RHEL/CentOS

Search for a Specific Process

- ps aux | grep nginx
- pgrep nginx # Show PID(s) of nginx

Understand Process Hierarchy

```
- pstree  # Visualizes parent-child process structure
```

Start a Process

```
./script.sh
./script.sh &       # Run in Background
- nohup ./script.sh &  # Keep Running After Logout
```

Kill/Stop/Manage a Process

Action	Command
Kill by PID	kill <PID>
Force kill	kill -9 <PID>
Kill by name	pkill nginx
Kill all matching	killall nginx

```
kill -9 1243
- pkill -f python
```

Foreground vs Background Jobs

```
- ./longtask.sh  # Run in Foreground
```

Press Ctrl + Z → pauses task, then run bg → resumes in background

```
jobs          # View Jobs
fg %1         # Bring Back to Foreground
```

Set Priority (nice, renice)

```
- nice -n 10 ./task.sh  # Launch with Low Priority
- renice -n 5 -p 1234  # Change Priority of Running Process
```

Lower value = higher priority: -20 (highest), 19 (lowest)

Process Details & Memory/CPU Stats

```
- ps aux --sort=-%mem | head  # Check Memory Usage
- ps aux --sort=-%cpu | head  # Check CPU Usage
```

Process Management Summary

Command	Purpose
ps -ef	List all processes
top / htop	Live process monitor
kill, pkill, killall	Stop processes
jobs, fg, bg	Manage background/foreground jobs
nice, renice	Adjust process priority
pgrep, pstree	Search or view hierarchy
lsof, strace	Debug processes/files

15. Linux Package Management

What Is Package Management?

Packages are compressed files that contain programs, libraries, or tools. Package managers are tools to install, update, remove, or search for software on Linux. DevOps often relies on package management to automate infrastructure setup (e.g., installing Docker, Ansible, or monitoring tools).

Types of Linux Package Managers

Distro Family	Tool	File Extension
Debian/Ubuntu	apt, dpkg	.deb
RHEL/CentOS/Fedora	yum, dnf, rpm	.rpm
Arch Linux	pacman	.pkg.tar.zst
Universal	snap, flatpak, AppImage	—

APT (Advanced Packaging Tool) – Ubuntu/Debian

Task	Command
Update list of packages	sudo apt update
Upgrade all packages	sudo apt upgrade

Install package	sudo apt install nginx
Remove package	sudo apt remove nginx
Search package	apt search <name>
Get info	apt show <package>
Clean old packages	sudo apt autoremove

Install from .deb File

```
- sudo dpkg -i file.deb
- sudo apt -f install   # Fix missing dependencies
```

YUM/DNF – RHEL, CentOS, Fedora

DNF is the newer replacement for YUM.

Task	YUM	DNF
Install	sudo yum install nginx	sudo dnf install nginx
Remove	sudo yum remove nginx	sudo dnf remove nginx
Update system	sudo yum update	sudo dnf upgrade
Search	yum search nginx	dnf search nginx

```
- sudo rpm -ivh package.rpm   # Install from .rpm
```

pacman – Arch Linux

```
- sudo pacman -Syu        # Full system update
- sudo pacman -S package    # Install
- sudo pacman -R package    # Remove
- sudo pacman -Ss search    # Search
```

snap – Universal Package Manager

- Works across distros.
- Packages include all dependencies (like containers).
- Slower but very easy to use.

```
- sudo snap install code --classic
- sudo snap remove code
- snap list
```

flatpak – Alternative Universal Manager

```
- flatpak install flathub com.spotify.Client
```

```
- flatpak run com.spotify.Client
```

Package Management Summary

Action	APT	YUM/DNF	RPM	Snap
Install	apt install	yum install	rpm -ivh	snap install
Remove	apt remove	yum remove	rpm -e	snap remove
Update system	apt upgrade	yum update	—	—
List installed	dpkg -l	yum list installed	rpm -qa	snap list

16. Linux Service Management

What is a Service?

A service (also called a daemon) is a background process, like: nginx, docker, sshd, mysql, etc. Services start manually (on demand) or automatically (at boot). In Linux, services are managed using init systems: Older: SysVinit (commands:
service, chkconfig) | Modern: systemd (commands: systemctl, journalctl)

systemctl – Modern Service Manager

Most modern Linux distributions (Ubuntu >=15, CentOS 7+, Debian >=8, RHEL 7+) use systemd.

Task	Command
Start a service	sudo systemctl start nginx
Stop a service	sudo systemctl stop nginx
Restart a service	sudo systemctl restart nginx
Reload without restart	sudo systemctl reload nginx
Check status	systemctl status nginx

Enable/Disable at Boot

Action	Command
Enable auto-start	sudo systemctl enable nginx

Disable auto-start	sudo systemctl disable nginx
Check if enabled	systemctl is-enabled nginx

View Service Logs

- journalctl -u nginx

Option	Description
-u	Show logs for specific service
-f	Follow logs (like tail -f)
--since '10 min ago'	Filter by time

- journalctl -u docker -f

Anatomy of a .service File

```
# /etc/systemd/system/myapp.service
[Unit]
Description=My Flask App
After=network.target

[Service]
ExecStart=/usr/bin/python3 /opt/myapp/app.py
WorkingDirectory=/opt/myapp
Restart=always
User=ubuntu

[Install]
WantedBy=multi-user.target
```

- sudo systemctl daemon-reload
- sudo systemctl enable myapp
- sudo systemctl start myapp

Legacy Commands (SysVinit-based distros)

Task	Command
Start	sudo service nginx start
Stop	sudo service nginx stop
Restart	sudo service nginx restart

Enable at boot	chkconfig nginx on
Disable	chkconfig nginx off

Service Management – systemctl Cheat Sheet

Command	Use
systemctl start <service>	Start service now
systemctl stop <service>	Stop it
systemctl restart <service>	Restart it
systemctl status <service>	Show status
systemctl enable <service>	Start on boot
systemctl disable <service>	Don't start on boot
journalctl -u <service>	View logs
systemctl daemon-reexec	Reload systemd itself

17. Linux Network Management

Why Network Management Matters

In Linux-based systems (servers, containers, VMs), network management helps to: Configure IP addresses, DNS, routing | Check connectivity and troubleshoot issues | Manage firewalls, ports, and network services | Set up virtual networks
(important in Docker/Kubernetes)

Essential Networking Commands

```
-   ip addr show   # Modern — Check IP Address & Interfaces
-   ifconfig       # Legacy (requires net-tools)
-   hostname -I    # Show IP only
-   ip route show  # View Routing Table
-   route -n
-   cat /etc/resolv.conf  # Check DNS Info
-   hostname           # Show hostname
-   hostnamectl set-hostname newname  # Change Hostname
```

Connectivity & Troubleshooting Tools

Tool	Use
ping <host>	Check if host is reachable
traceroute <host>	Show path to host
nslookup <host>	DNS resolution
dig <host>	Advanced DNS info
telnet <host> <port>	Test port connectivity
nmap <host>	Scan open ports
netstat -tulnp	Show listening ports (TCP/UDP)
ss -tuln	Modern version of netstat
curl, wget	Test HTTP/HTTPS connections
tcpdump	Packet capture tool
ip a, ip r, ip link	Interface and routing config

Configure Network Interfaces

Using ip (Modern Tool)

```
- sudo ip addr add 192.168.1.10/24 dev eth0  # Assign IP manually
- sudo ip link set eth0 up
- sudo ip addr del 192.168.1.10/24 dev eth0  # Remove IP
```

Using nmcli (NetworkManager CLI)

```
- nmcli device status                  # Show devices
- nmcli connection up "Wired connection 1"   # Enable connection
```

Network Configuration Files

File	Purpose
/etc/hosts	Map hostnames to IPs manually
/etc/hostname	System's hostname
/etc/resolv.conf	DNS nameservers
/etc/network/interfaces	Debian-based static IP config

/etc/sysconfig/network-scripts/ifcfg-*	RHEL-based network config

Firewall Management

UFW (Ubuntu/Debian)

```
- sudo ufw status
- sudo ufw allow 22    # Allow SSH
- sudo ufw deny 80
- sudo ufw enable
```

Firewalld (RHEL/CentOS/Fedora)

```
- sudo firewall-cmd --list-all
- sudo firewall-cmd --add-port=8080/tcp --permanent
- sudo firewall-cmd --reload
```

iptables (Legacy/Manual)

```
- sudo iptables -L                          # List rules
- sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
```

Host to Host Communication

Task	Command
Test SSH	ssh user@host
Share file	scp file.txt user@host:/path
Sync folders	rsync -av /source/ user@host:/target/
Setup port forwarding	ssh -L 8080:localhost:80 user@remote

18. Linux Troubleshooting

What Is Troubleshooting?

Troubleshooting in Linux means identifying and fixing: System errors, Service failures, Performance bottlenecks, Network issues, Disk problems, Permissions and access issues.

System Health Check

```
- top
- htop
- uptime          # Check load average
- free -h         # Check memory usage
```

```
- vmstat 1          # System performance in real-time
- df -h             # Show available disk space
- du -sh *          # Check size of subdirectories
- lsblk             # List block devices
- dmesg | grep -i error
- sudo smartctl -a /dev/sda
```

Log Files (Goldmine of Info)

Location	Purpose
/var/log/syslog or /var/log/messages	General system logs
/var/log/auth.log	Login attempts, sudo access
/var/log/dmesg	Kernel and hardware logs
/var/log/nginx/ or /var/log/httpd/	Web server logs
/var/log/mysql/	DB logs
/var/log/journal/	systemd logs (if journald enabled)

```
- tail -n 50 /var/log/syslog
- journalctl -xe
```

Service Failures (Systemd-based systems)

```
- systemctl status nginx
- journalctl -u nginx -b
- sudo systemctl restart nginx
- sudo systemctl daemon-reload
```

Permission Issues

Common symptoms: Permission denied errors | Files not executable | Services unable to read config files

```
- ls -l filename
- chmod +x script.sh
- chown user:group file
```

Application Debugging

Tool	Use

bash -x script.sh	Debug bash scripts line by line
python3 -m pdb script.py	Python debugger
npm run dev or node app.js	Watch console logs for Node.js
curl -v	Verbose API call debugging

Process & Resource Issues

```
- ps -ef, top, htop              # List processes
- kill <PID> or pkill <name>     # Kill process
- ps aux --sort=-%mem | head     # Memory hogs
- ps aux --sort=-%cpu | head     # CPU hogs
```

Reboot, Shutdown, Recovery

Task	Command
Reboot	sudo reboot
Shutdown	sudo shutdown now
Rescue Mode (grub)	Hold Shift at boot > Select Advanced > Recovery mode
Remount root fs	mount -o remount,rw /
Filesystem check	fsck /dev/sda1 (carefully, needs unmounted disk)

Troubleshooting Checklist

- Is the service running?
- Is the port open and listening?
- Can you ping / curl the target?
- Do you see anything in the logs?
- Is it a permission or ownership problem?
- Is the disk or memory full?
- Has anything changed (deployment, config, updates)?

19. User Access – Full Admin vs Limited Access

Scenario Overview

Assume you're a DevOps/Linux admin and need to: Add a new user, Give them either full sudo (admin) access or limited command access (e.g., restart Apache only)

Step-by-Step: Create and Manage Linux User Access

Step 1: Create a New User

```
- sudo adduser john
```

This creates a new user john with a home directory. **Step 2: Set a Password for the User**

```
- sudo passwd john
```

Step 3: Give Full Admin (sudo) Access

```
- sudo usermod -aG sudo john     # Option 1: Ubuntu/Debian
- sudo usermod -aG wheel john    # Option 2: CentOS/RHEL
```

This gives full root-equivalent access via sudo.

Step 4: Allow Specific Command via sudoers

```
- sudo visudo
- john ALL=(ALL) NOPASSWD: /bin/systemctl restart apache2
```

Allow Multiple Commands

```
- john ALL=(ALL) NOPASSWD: /sbin/ifconfig, /usr/bin/apt-get update
```

Additional User Access Controls

```
- which <command>           # Find Command Full Paths
- sudo usermod -s /usr/sbin/nologin john   # Restrict Shell Access
- groups john                 # Check Group of a User
- sudo deluser john sudo     # Remove Sudo Access (Ubuntu/Debian)
- sudo gpasswd -d john wheel  # Remove Sudo Access (RHEL/CentOS)
```

User Access Summary Table

Task	Command
Add user	sudo adduser john
Set password	sudo passwd john
Full admin	sudo usermod -aG sudo john
Partial access	sudo visudo → add custom rule

Check access	sudo -l -U john
Remove sudo	sudo deluser john sudo

20. Linux File System Structure

The Linux file system is a hierarchical directory structure that starts from the root (/) directory and branches out to other subdirectories.

Root Directory /

- The top-level directory of the Linux file system.
- All files and directories start from here.
- It is the parent of all other directories.

Standard Directories Under /

Directory	Description
/bin	Essential user binaries (commands): ls, cp, mv, rm, etc.
/sbin	System binaries: Admin-level tools like iptables, reboot, ifconfig.
/boot	Contains bootloader files, Linux kernel (vmlinuz), initrd, GRUB config.
/dev	Device files: e.g., /dev/sda, /dev/tty0. Represents hardware devices as files.
/etc	System-wide configuration files: e.g., /etc/passwd, /etc/hosts, /etc/fstab.
/home	User home directories: /home/riyas, /home/mohan.
/lib	Essential shared libraries for binaries in /bin and /sbin.
/media	Mount point for removable media: USB, CD/DVD auto-mounted here.
/mnt	Used for temporarily mounting file systems manually by the user.
/opt	Optional application packages (e.g., third-party apps like Chrome, VMware).
/proc	Virtual filesystem providing process and kernel info (e.g., /proc/cpuinfo).

/root	Home directory of root user. Different from /home/root.
/run	Runtime data since the last boot, like PID files, socket files.
/srv	Service-related data for servers like FTP, WWW.
/sys	Virtual filesystem showing kernel info about devices.
/tmp	Temporary files. Cleared on reboot.
/usr	Secondary hierarchy for read-only user data; has subdirs like /usr/bin, /usr/lib.
/var	Variable data like logs, spool files, cache, and emails.

Important /etc Files

File	Purpose
/etc/passwd	Stores user account information
/etc/shadow	Contains user password hashes
/etc/group	Stores group info
/etc/hostname	Sets system hostname
/etc/fstab	Contains auto-mount information
/etc/hosts	Maps IP addresses to hostnames
/etc/resolv.conf	DNS configuration
/etc/systemd/	Contains service files and system boot configuration

Important /var Subdirectories

Directory	Purpose
/var/log	System log files (important for troubleshooting)
/var/spool	Print and mail spools
/var/tmp	Temp files preserved between reboots
/var/cache	Cached data from applications

/var/www	Web files (for Apache/Nginx servers)

Virtual Filesystems: /proc, /sys, /dev

/proc

- Dynamic, virtual filesystem.
- Used to access process info, system uptime, memory stats.
- Example: cat /proc/cpuinfo, cat /proc/meminfo.

/sys

- Interfaces with kernel devices.
- Helps in examining and interacting with hardware.

/dev

- Represents devices as files.
- Example: /dev/sda = hard disk, /dev/ttyUSB0 = USB device.

Package & Binary Locations

Path	Use
/bin	Essential system binaries
/usr/bin	Non-essential user commands
/sbin, /usr/sbin	Admin commands
/lib, /usr/lib	Shared libraries
/opt	Third-party packages

Linux File System Hierarchy Diagram

```
/
├── bin
├── boot
├── dev
├── etc
├── home
│   ├── user1
│   └── user2
├── lib
├── media
├── mnt
├── opt
├── proc
├── root
├── run
├── sbin
├── srv
├── sys
├── tmp
├── usr
│   ├── bin
│   ├── lib
│   └── share
└── var
    ├── log
    ├── cache
    └── tmp
```

Key Concepts to Remember

- Everything in Linux is a file (including hardware and processes).
- The file system follows FHS (Filesystem Hierarchy Standard).
- Root (/) is the origin — all other directories are children of it.
- Separation of concerns: logs, binaries, configs, libraries — all have dedicated locations.

Document Prepared: 19 February 2025

LinkedIn: https://www.linkedin.com/in/sumit-deshmukh-3b781527a/