

# LINUX FOR NETWORK ENGINEER

# Click IT

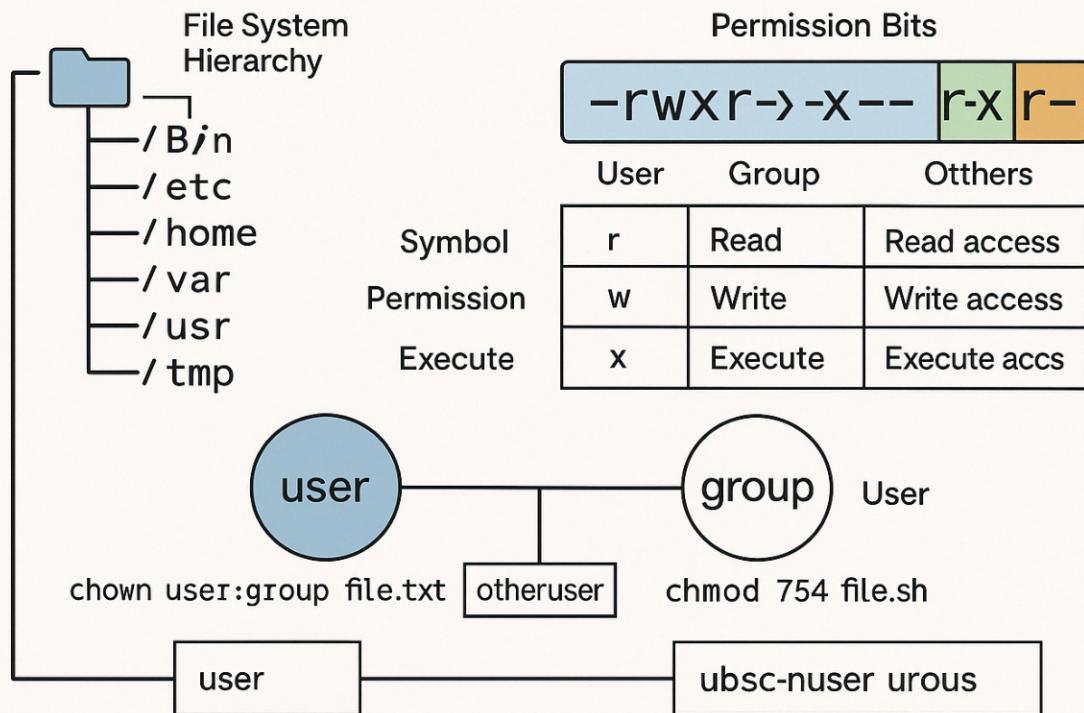
## LINUX BASICS

21-APRIL-2025

## STRUCTURE

LINUX

### Linux File and Directory Permissions



# INTRODUCTION - WHAT IS LINUX?

Linux is the **operating system of the internet**. From network devices to cloud servers and automation controllers, Linux powers them all. For network engineers, understanding Linux isn't optional — it's foundational.

## WHY LINUX?

- Used in routers (Cisco IOS XR, JunOS), firewalls, load balancers
- Core OS for servers, automation, and DevOps
- Flexible, open-source, scriptable
- Supports powerful networking tools: tcpdump, nmap, wireshark, netstat, etc.

In this guide, we'll explore everything from distro selection to permissions, from scripting to automation.

# VISUAL GUIDE TO LINUX PERMISSIONS, FILE SYSTEM & OWNERSHIP

Understanding file permissions and the Linux filesystem hierarchy is critical for both system security and automation scripts. This visual diagram provides a comprehensive view of how Linux handles:

## FILE SYSTEM HIERARCHY

- Root (/) is the starting point of everything.
- Key directories like /bin, /etc, /home, /var, /usr, and /tmp organize the OS and user data.

## PERMISSION BITS BREAKDOWN

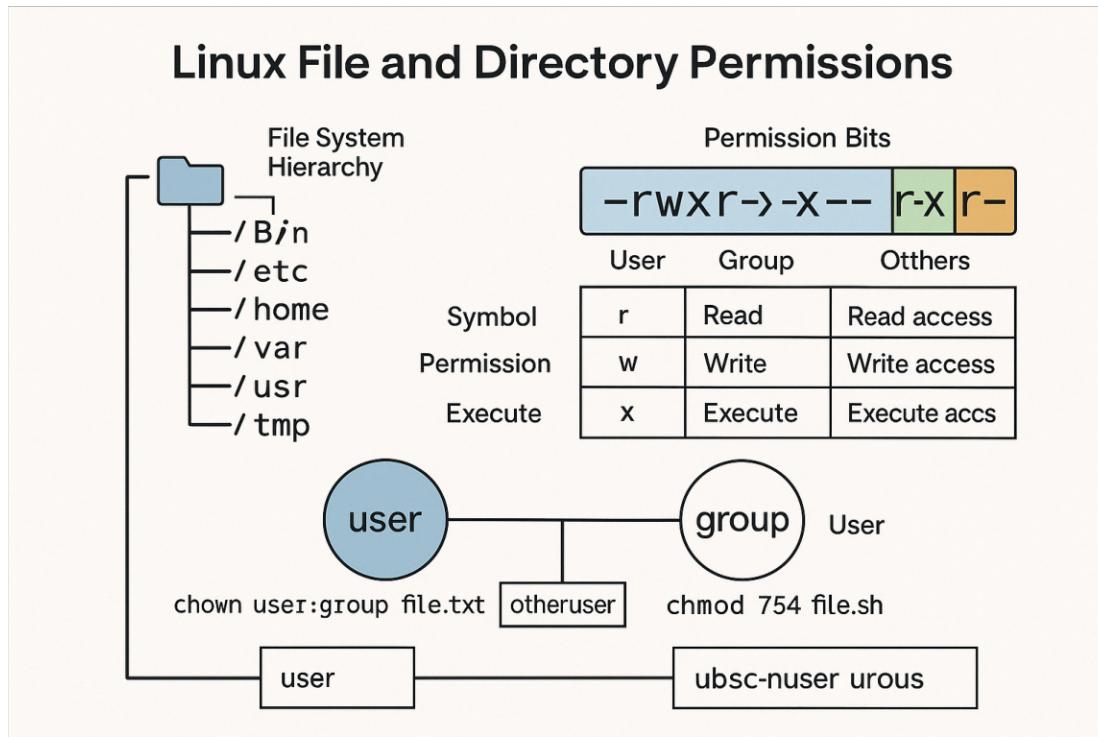
Linux permissions are expressed using a symbolic and numeric format:

- -rwxr-xr-- = file with user (rwx), group (r-x), others (r--)
- This translates to **read**, **write**, and **execute** rights for different users.

Symbol	Meaning	Applies To
r	Read access	User, Group, Others
w	Write access	User, Group, Others
x	Execute access	User, Group, Others

Use chmod to change permissions:

```
chmod 754 file.sh
```



# CHOOSING THE RIGHT LINUX DISTRO

There are hundreds of Linux distributions. Here's how to choose based on **purpose**:

Distro	Based On	Use Case	Best For
<b>Ubuntu</b>	Debian	Labs, cloud, DevOps	Beginner-friendly
<b>Debian</b>	-	Stable routers/firewalls	Minimalist systems
<b>CentOS/AlmaLinux</b>	RHEL	Data center, enterprise use	Red Hat environments
<b>Kali Linux</b>	Debian	Security, pen testing	Ethical hacking
<b>OpenWRT</b>	Custom	Embedded routers	IoT, firmware
<b>Arch Linux</b>	-	Learning Linux deeply	Advanced users

Example: Want to use Ansible, Git, Docker? Go for Ubuntu 22.04 LTS.

Installation tools:

- Rufus (Windows) to create bootable USB
- dd command (Linux/Mac)
- Try inside VirtualBox/VMware for a lab

# TEXT EDITORS

A network engineer often needs to modify configuration files, create scripts, or troubleshoot logs. That's where text editors come in.

## Popular Text Editors:

### 1. **Nano (Simple & Friendly)**

```
nano /etc/network/interfaces
```

- Ctrl+O: Save
- Ctrl+X: Exit

### 2. **Vim (Powerful but has learning curve)**

```
vim /etc/hosts
```

- i: Insert mode
- Esc: Exit insert
- :wq: Save and quit

### 3. **VS Code (Graphical, Modern IDE)**

```
sudo snap install code --classic
```

Use Vim for remote SSH sessions; use VS Code for local DevOps scripting.

# LINUX FUNDAMENTALS

Understanding Linux begins with mastering its **core commands** and internal logic. Linux is a **command-line-first OS**, meaning its real power is unlocked through the terminal.

## 1. NAVIGATING THE FILESYSTEM

```
pwd          # Show current directory path
cd /etc      # Move to /etc directory
cd ~         # Go to user's home directory
cd ..        # Move one directory up
cd -         # Jump to previous directory
ls           # List files
ls -l        # List with permissions, owner, size, date
ls -a        # Include hidden files
ls -lh       # Human-readable sizes
```

**Tip:** Use tree to view folder structure visually.

```
sudo apt install tree
tree /etc | head -20
```

## 2. VIEWING AND EDITING FILES

```
cat /etc/os-release    # Show contents of file
less /var/log/syslog   # View long file, scrollable
head -n 10 file.txt    # Show first 10 lines
tail -n 20 file.txt    # Show last 20 lines
tail -f /var/log/auth.log # Live monitor logs
```

less allows scroll with spacebar (down) and b (back).

## 3. CREATING FILES & DIRECTORIES

```
touch myfile.txt       # Create new empty file
mkdir backups          # Create directory
mkdir -p configs/interfaces # Nested directory creation
```

-p ensures parent directories are created if missing.

## 4. FILE TYPE AND CONTENT INSPECTION

```
file /bin/bash         # See file type (binary, text, etc.)
wc -l filename.txt    # Count lines
wc -w filename.txt    # Count words
stat filename.txt     # Detailed file info (modification time, etc.)
```

## 5. SEARCHING FOR FILES AND CONTENT

### FIND FILES:

```
find /etc -name "hosts"          # Exact match  
find . -type f -name "*.conf"   # All .conf files in current dir
```

### SEARCH INSIDE FILES:

```
grep "interface" /etc/network/interfaces  
grep -rn "hostname" /etc          # Recursive + show line number
```

✗ Use egrep or grep -E for extended regex:

```
grep -E "eth[0-9]+" interfaces.txt
```

## 6. UNDERSTANDING COMMAND STRUCTURE

Most commands follow this format:

command [options] [arguments]

### EXAMPLE:

```
ls -al /etc  
# command = ls  
# option = -a (all files), -l (long format)  
# argument = /etc (directory)
```

## 7. WILDCARDS & EXPANSION

Wildcards help target multiple files:

```
ls *.conf      # All .conf files  
ls a*         # Files starting with 'a'
```

Brace expansion:

```
mkdir folder_{1..5}  # Creates folder_1 to folder_5
```

## 8. KEYBOARD SHORTCUTS (COMMAND LINE EFFICIENCY)

Shortcut	Action
<b>Tab</b>	Auto-complete
<b>Ctrl + C</b>	Kill current process
<b>Ctrl + D</b>	Log out / end input
<b>Ctrl + U</b>	Delete whole line

<b>Ctrl + A</b>	Move to beginning of line
<b>Ctrl + E</b>	Move to end of line
<b>!!</b>	Run last command again
<b>!ssh</b>	Run last command starting with ssh

## 9. ALIASES (CUSTOMIZE YOUR SHELL)

Aliases save time:

```
alias ll='ls -alF'
alias gs='git status'
alias pingg='ping google.com'
```

To make them permanent, add to `~/.bashrc`:

```
echo "alias cls='clear'" >> ~/.bashrc
source ~/.bashrc
```

## 10. SHELL ENVIRONMENT VARIABLES

```
echo $HOME
echo $USER
echo $PATH
export MYVAR="NetworkEngineer"
```

To persist across sessions, add to `~/.bashrc` or `~/.profile`.

## 11. GETTING HELP

```
man ls          # Manual page
ls --help       # Brief help
whatis grep    # One-line description
which tcpdump   # Path to binary
```

Use `man -k <keyword>` to search man pages:

```
man -k network
```

## PRO TIPS FOR NETWORK ENGINEERS:

- Always use less or grep when analyzing log files.
- Use alias for long commands you run daily (e.g., BGP traceroutes).
- Combine tools in pipelines:

```
netstat -tunap | grep :22 | less
```

# UNDERSTANDING THE LINUX FILE SYSTEM

```
/  
└── bin/      → Essential binaries (e.g., ls, cp)  
└── etc/      → System configs (e.g., networking)  
└── home/     → User home directories  
└── root/     → Root user's home  
└── usr/      → User applications  
└── var/      → Logs, spools  
└── tmp/      → Temporary files  
└── dev/      → Devices (e.g., /dev/sda)
```

## Commands:

```
df -h      # Disk usage  
du -sh *   # Size of current dir  
mount      # Mounted filesystems
```

/etc/ is critical for sysadmins and neteng — store configs like DNS, network, sshd.

# FILE TOOLS, OWNERSHIP, PERMISSIONS

Use ls -l to view permissions:

```
-rwxr-xr-- 1 root root 1234 Apr 20 file.sh
```

Meaning:

- - : Regular file
- rwx : Owner can read/write/execute
- r-x : Group can read/execute
- r-- : Others can only read

## Ownership Commands:

```
chown netadmin:neteng file.txt  # Change owner  
chmod 755 file.txt           # rwx for owner, rx for others
```

Use umask to define default permissions.

# COPY, MOVE, DELETE FILES

## Copy files

```
cp file.txt /backup/  
cp -r configs/ /etc/           # Recursive copy
```

## Move files

```
mv file.txt /var/tmp/  
mv *.log /var/logs/
```

## Delete files

```
rm file.txt  
rm -rf /tmp/old_logs/
```

**Deletion is permanent — no Recycle Bin. Use trash-cli if needed:**

```
sudo apt install trash-cli  
trash-put file.txt
```

# USERS, GROUPS & PASSWORDS

Linux is a **multi-user OS**.

## Create users:

```
sudo adduser netadmin  
sudo passwd netadmin
```

## Groups:

```
sudo groupadd neteng  
sudo usermod -aG neteng netadmin
```

## File to know:

- /etc/passwd – user accounts
- /etc/shadow – encrypted passwords
- /etc/group – group memberships

View info:

```
id netadmin  
groups netadmin
```

# PERMISSIONS DEEP DIVE

## OCTAL VALUES

Symbol	Meaning	Value
r	Read	4
w	Write	2
x	Execute	1

```
chmod 755 file.sh    # Owner rwx, Group rx, Others rx  
chmod 600 config.cfg # Only owner can read/write
```

## SYMBOLIC MODE:

```
chmod u+x script.sh  # Add execute to user  
chmod go-rwx file.txt # Remove all access from group and others
```

Use stat filename to see detailed permission + ACLs.

# PROCESSES

Linux manages processes with IDs (PIDs).

## View Processes:

```
ps aux | grep ssh  
top          # Live usage  
htop          # Better UI
```

## Kill Processes:

```
kill 1234      # Send SIGTERM  
kill -9 1234   # Force kill (SIGKILL)
```

## Background Jobs:

```
./long_job.sh &  
jobs  
fg %1        # Resume job
```

`nohup` and `screen` let you run persistent processes over SSH.

# INSTALLING PACKAGES

## APT (Debian/Ubuntu)

```
sudo apt update  
sudo apt install net-tools curl  
sudo apt remove apache2
```

## YUM/DNF (Red Hat)

```
sudo yum install tcpdump  
sudo dnf remove nginx
```

## Search for packages:

```
apt-cache search nmap
```

Use snap and flatpak for containerized applications.

## BUILD A NETWORK TOOLBOX

```
sudo apt install rsyslog  
sudo nano /etc/rsyslog.conf
```

Enable UDP logging:

```
module(load="imudp")  
input(type="imudp" port="514")
```

Restart:

```
sudo systemctl restart rsyslog
```

Check logs:

```
tail -f /var/log/syslog
```

Configure routers to send syslog to this server.

## BASH SCRIPTING FOR NETWORKING

### Script: IP Interface Reporter

```
#!/bin/bash  
echo "Checking all interfaces..."  
ip -brief addr show | grep -v lo | while read line; do  
    echo $line  
done
```

Save it as interfaces.sh:

```
chmod +x interfaces.sh  
./interfaces.sh
```

Automate:

```
crontab -e  
# Add:  
0 7 * * * /home/user/scripts/interfaces.sh >> /home/user/logs/report.log
```

# CONCLUSION: YOUR LINUX JOURNEY BEGINS HERE

You've just explored a deep dive into the Linux universe — not as a generic user, but as a **network engineer with a mission**: automation, DevOps, visibility, and control over infrastructure.

## WHAT YOU'VE ACCOMPLISHED

- Mastered **Linux distributions** and picked the right one for network and automation labs
- Gained fluency in **navigation, file systems, and file permissions** — the backbone of Linux
- Learned how to create, move, edit, and search through files like a sysadmin
- Understood how **users, groups, and permissions** work — critical for securing multi-user environments
- Used real-world networking tools: tcpdump, nmap, iperf, rsyslog, and even wrote your first **bash script**
- Practiced **package management, process handling, and text editing** with confidence

## WHY THIS MATTERS

In today's world, **networking is not just cables and routing protocols** — it's automation, configuration management, cloud provisioning, container orchestration, and observability. And behind all of that? **Linux**.

Whether you're:

- Troubleshooting packet drops with tcpdump
- Writing Ansible playbooks to configure routers
- Running Docker containers for NetBox or Grafana
- Monitoring logs from routers via a Linux syslog server

**Linux is the core toolset** that connects everything.