Linux Commands for SRE

Linux File System Navigation & Commands

🏠 Basic Navigation Commands

🔹 pwd – Print current working directory
🔹 ls – List files and directories
🔹 cd <directory> – Change directory

```
cd /etc      # Go to the /etc directory
ls -l        # List files in long format
```

📂 File & Directory Management

📁 **Create & Navigate**
🔹 mkdir folder1 – Create a new directory
🔹 rmdir folder1 – Remove an empty directory
📄 **File Operations**
🔹 touch file.txt – Create a new file
🔹 cat file.txt – Show file contents
🔹 head -5 file.txt – Display the first 5 lines
🔹 tail -5 file.txt – Display the last 5 lines
🔹 tail -f /var/log/syslog – Continuously monitor a log file
📑 **Copy, Move & Remove**
🔹 cp source destination – Copy files
🔹 mv old_name new_name – Rename or move files
🔹 rm file.txt – Remove a file
🔹 rm -r folder1 – Remove a directory

🛠️ System Information Commands

👤 **User Information**
🔹 whoami – Show the current logged-in user
🔹 hostname – Display the system hostname
💻 **System Details**
🔹 uname -a – Show OS & kernel details
🔹 uptime – Show system uptime
🔹 lscpu – Display CPU information
🔹 cat /proc/cpuinfo – View detailed CPU info
🔹 cat /proc/meminfo – View memory details

📝 **Text Editors**
✏️ **vi (Advanced, Preferred for Servers)**
🔹 Open: vi filename
🔹 Edit mode: Press i

- 🔹 Save & exit: Press ESC, type :wq, press Enter
- 🔹 Exit without saving: Press ESC, type :q!, press Enter

📝 **nano (Easier for Beginners)**

- 🔹 Open: nano filename
- 🔹 Edit content
- 🔹 Save: CTRL + O, then Enter
- 🔹 Exit: CTRL + X

🔍 Searching & Filtering Files

- 🔹 find – Locate files based on name, type, size, modification time
- 🔹 grep – Search inside files

SRE Use Cases

✅ **Find a specific configuration file**

```
find / -type f -name "nginx.conf"
```

✅ Find files modified in the last 24 hours

```
find /etc -type f -mtime -1
```

✅ Find large log files (greater than 10MB)

```
find /var/log -type f -size +10M
```

📌 Grep – Search Inside Files

- 🔹 grep "error" /var/log/nginx/*.log – Search for "error" in Nginx log files
- 🔹 grep -c "failed" /var/log/auth.log – Count occurrences of "failed" in auth logs
- 🔹 grep -r "server_name" /etc/nginx/ – Recursively search for "server_name" in all files under /etc/nginx/
- 🔹 grep -n "gcloud" test/APMTrianing/README.md – Show line numbers with matches
- 🔹 grep -E "timeout|connection" /var/log/nginx/*.log – Search for multiple patterns (timeout or connection)
- 🔹 grep -n "max_connections" /etc/mysql/my.cnf – Show line numbers for matches in MySQL config

📍 **Locate – Find Files Quickly**

locate file.txt – Find files instantly using a pre-built database

- 🔹 **Difference Between find & locate**
  - 🕒 find – Real-time search, always up-to-date
  - ⚡ locate – Faster but relies on an indexed database (updated periodically)

🔐 Linux User & Permissions Management

📜 Understanding File Permissions

👀 Read (r) | ✍️ Write (w) | ⚙️ Execute (x)

🔹 Check File Permissions

📜 Understanding File Permissions

👀 Read (r) | ✍️ Write (w) | ⚙️ Execute (x)

🔹 Check File Permissions

```
ls -l /home/user/file.txt
```

📌 Example Output:

```
-rw-r--r--  1 user user 1024 Mar 15 10:00 file.txt
```

✔️ **rw-** → Owner has **read & write** permission

✔️ **r--** → Group has **read-only** access

✔️ **r--** → Others have **read-only** access

⚙️ Changing Permissions

🔹 Give execute permission to all users

```
chmod 755 myfile.sh
```

🔹 Change file ownership

```
chown user:group myfile.sh
```

⚠️ File Permission Issues & Best Practices

🔍 Check File Owner

```
ls -l /opt/app/config.json
```

🔹 **Secure File Access** (Only owner can read & write)

```
chmod 600 file.txt
```

🔹 Grant Read Permission

```
sudo chmod +r /home/testuser/secretfile.txt
```

🔹 Change Ownership

```
sudo chown appuser:appuser /opt/app/config.json
```

🔹 Grant Read & Execute to Others, but Only Owner Can Write

```
sudo chmod 755 /opt/app
```

🔹 Recommended Permissions for Logs & Configs

```
chmod 770 /var/log/myapp/
```

🔹 Recommended for Web Apps & Shared Directories

```
chmod 775 /var/www/html/
```

📦 APT Package Management (Ubuntu & Debian)

APT (Advanced Package Tool) is used to manage .deb packages on Ubuntu and Debian.

🔹 Real-Time Use Cases of APT in DevOps & Cloud

🌐 Scenario 1: Setting Up a Web Server on Ubuntu

💡 **Use Case**: You need to install Nginx on a cloud VM to serve a website.

```
sudo su
sudo apt update                # Update package list
sudo apt install nginx -y      # Install Nginx
nginx -v                       # Check Nginx version
sudo systemctl enable nginx    # Enable auto-start on reboot
sudo systemctl start nginx     # Start Nginx service
sudo systemctl status nginx    # Check Nginx service status
```

🔧 Package Maintenance & Fixes

```
sudo apt update --fix-missing    # Fix missing dependencies
sudo apt remove nginx -y         # Remove Nginx but keep config files
sudo apt purge nginx -y          # Remove Nginx completely (incl. config)
sudo apt autoremove -y           # Remove unused dependencies
```

🔍 Searching & Viewing Packages

```
apt search mysql-server          # List available MySQL packages
apt show git                     # Show details of the Git package
apt list --installed | grep nginx    # Verify installed package version
```

⚠️ Handling Broken Packages

```
sudo apt --fix-broken install      # Fix dependency issues
sudo apt reinstall mysql-server    # Reinstall MySQL Server
```

⚙️ Managing Services (systemctl & journalctl)

🔄 Service Management

```
systemctl status nginx   # Check if Nginx is running
systemctl start nginx    # Start Nginx service
systemctl stop nginx     # Stop Nginx service
```

📜 Monitoring Logs in Real-Time (journalctl)

🔹 journalctl helps monitor logs from **systemd services** like web servers and databases.

```
journalctl -u nginx       # View logs for Nginx service
journalctl -u nginx -f    # Follow logs in real-time
```

💡 **Real-time example**: As a DevOps engineer, use journalctl to monitor logs for nginx, apache2, or mysql to ensure services run correctly and troubleshoot issues.

🌍 Network Tools: cURL & Wget

📡 cURL (Client URL) - HTTP Requests & API Testing

```
curl -I https://example.com        # Check if a website is reachable
curl -Is https://example.com | head -n 1  # Quick site status check
curl -v https://example.com        # Get full HTTP response details
curl --insecure -v https://example.com 2>&1 | grep -i "expire date"  # Check SSL expiry
```

📡 API Requests with cURL

```
curl -X GET https://jsonplaceholder.typicode.com/posts/1
curl -X POST https://api.example.com/data \
     -H "Content-Type: application/json" \
     -d '{"name": "John", "age": 30}'
```

📁 File Downloading with cURL

```
curl -O https://example.com/file.tar.gz
```

📥 Wget - Downloading Files & Websites

```
wget https://example.com/file.zip
```

📦 Compression & Extraction

Command	Description
tar -czvf backup.tar.gz /home/user	📌 Compress a directory
tar -xzvf backup.tar.gz	📌 Extract a compressed file
zip -r archive.zip folder/	📌 Create a ZIP file
unzip archive.zip	📌 Extract a ZIP file

⚙️ Linux Process & Service Management

Command	Description
ps aux	📌 View all running processes
`ps -aux --sort=-%cpu	head -10`
`ps -aux --sort=-%mem	head -10`
top	📌 Monitor CPU & memory usage in real-time
htop	📌 Interactive process viewer (requires installation)

Command	Description
kill 1234	📌 Kill a process with process ID 1234

💻 Check CPU Usage

Command	**Description**
pidstat -d 1 5	📌 Monitor CPU usage of processes every second for 5 intervals

---

💻 Check Memory Usage

Command	Description
free -m	📌 Show total, used, and free memory in MB
free -h	📌 Show memory usage in a human-readable format
vmstat 1 5	📌 Display CPU & memory usage every second for 5 intervals
top	📌 Monitor system resource usage in real-time
htop	📌 Advanced process viewer for monitoring memory usage

🛠️ Real-Time Scenarios & Troubleshooting

🔹 **Scenario 1: Disk Space Running Low on Production Server**

📌 **Issue**: The application is slow, and logs show "No space left on device."

Command	Description
df -h	📌 Check disk usage in human-readable format
du -sh /var/log/*	📌 Find large log files
rm -rf /var/log/old.log	📌 Delete unwanted logs

📁 Swap Space in Linux

🔹 **Swap is disk-based memory that Linux uses when RAM is full.**

Command	Description
swapon -s	📌 Check active swap usage

🔐 SCP & SSH

🔹 SSH Key-Based Authentication

📌 Step 1: Generate SSH Key on VM1

```
ssh-keygen -t rsa -b 4096
ls -l ~/.ssh/id_rsa.pub
cat ~/.ssh/id_rsa.pub  # Copy this public key
```

📌 Step 2: Configure Remote Server (10.128.0.63)

1.**Log into the remote VM using the console:**

```
ssh <your-username>@10.128.0.63
```

2.**Setup SSH Keys:**

```
mkdir -p ~/.ssh
vi ~/.ssh/authorized_keys  # Paste the public key from VM1 & save
chmod 700 ~/.ssh
chmod 600 ~/.ssh/authorized_keys
sudo systemctl restart ssh  # Restart SSH service
```

4.**Test SSH Login from VM1:**

```
ssh <your-username>@10.128.0.63
```

📁 SCP (Secure Copy Protocol) - File Transfers

Command Description

scp <file> username@remote-ip:<filepath>	📌 Copy a file to a remote server
scp /home/user/file.txt username@10.128.0.63:/home/user/	📌 Copy file.txt to remote VM (10.128.0.63)

🛡 Firewall Management (UFW - Uncomplicated Firewall)

📌 In a production environment, managing the firewall is crucial for securing network communication.

```
sudo ufw status  📌 Check firewall status
sudo ufw allow 80/tcp  📌 Allow HTTP (port 80) traffic
```