**lsof:** The lsof command in Unix/Linux stands for "List Open Files". It shows information about files that are currently open by active processes.

List all open files by a specific user:

**lsof -u username**

Find which process is using a specific port:

**lsof -i: portnumber**

**Kill a process using a specific port  
First, find the PID using: lsof -i :<port>**

**Then, kill the process:kill -9 <PID>**

**-i → Lists network files (TCP, UDP, etc.)**

**-u <user> → Lists files opened by a specific user**

**-p <PID> → Lists files opened by a specific process ID**

**+D <directory> → Lists all files opened under a specific directory**

**-t → Returns only process IDs (useful for scripting)**

**Inode:**

An inode is a data structure used by Unix/Linux file systems to store information about a file or directory. Every file has a unique inode that contains metadata about the file but not its name or actual data.

#### **What does an inode store?**

* File type (regular file, directory, symbolic link, etc.)
* Permissions (read, write, execute)
* Owner and group ID
* File size
* Timestamps (creation, modification, access times)
* Link count (how many hard links point to this file)
* Pointers to data blocks (where the actual data is stored on disk)

#### **How to check inode of a file: ls -i filename**

**Process ID PID:**

A Process ID (PID) is a unique number assigned by the operating system to each process running on a system. It's used to track and manage processes.

**List all running processes with their PIDs:** ps aux

**Find the PID of a specific process:** pidof process\_name

**Get the PID of a running process by name:** pgrep process\_name

**Kill a process by PID:** kill -9 <PID>

**Top command:**

The top command in Unix/Linux a dynamic, constantly-updating view of system processes. It displays information about CPU usage, memory usage, running processes, load average.

| PID | USER | PR | NI | VIRT | RES | SHR | S | %CPU | %mem | TIME | CMD |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

**PID:** Process ID of the task.

**USER:** Owner of the process.

**PR:** Priority of the process.

**NI:** Nice value (affects process priority).

**VIRT:** Virtual memory used by the process.

**RES:** Physical RAM used by the process.

**SHR:** Shared memory used by the process.

**S:** Process status (R = running, S = sleeping, Z = zombie, etc.).

**%CPU:** CPU usage percentage.

**%MEM:** RAM usage percentage.

**TIME+:** Total CPU time used by the task.

**COMMAND:** Name of the running command/process.

**chmod:**

The chmod command in Unix/Linux is used to **change the permissions** (read, write, execute) of files and directories.

| Each file has three permission groups:  * **User (u):** The owner of the file. * **Group (g):** A group of users. * **Others (o):** Everyone else. | Each group can have:  * **r** → Read permission (4) * **w** → Write permission (2) * **x** → Execute permission (1) |
| --- | --- |

**chown:**

The chown command changes the owner and/or group of a file or directory.

**Change owner only:** sudo chown username filename

**Change owner and group:** sudo chown username:groupname filename

**Change ownership recursively (for directories and contents):**sudo chown -R username:groupname /path/to/directory

**Verify ownership:** ls -l filename

**tail:**

The tail command in Unix/Linux is used to display the last few lines of a file, typically for monitoring logs or outputs in real-time. By default, it shows the last 10 lines of a file.

**find:**

The find command in Unix/Linux is used to **search for files and directories** in a directory hierarchy based on various criteria like name, size, type, or modification time.

#### **Basic Syntax:** find [path] [expression] [options]

* **Find a file by name:** find /path/to/search -name "filename.txt"
* **Find files by type:**
* **Regular files:** find /path/to/search -type f
* **Directories:**find /path/to/search -type d
* **Find files by size:** find /path/to/search -size +100M
* **Find files modified in the last 7 days:** find /path/to/search -mtime -7

**grep:**

The grep command searches for patterns in text using regular expressions. It's often used to filter text output or find specific strings inside files.

#### **Basic Syntax: grep [options] "pattern" filename**

* **Search for a specific word in a file: grep "error" logfile.txt**
* **Search recursively in all files inside a directory:**

**grep -r "TODO" /path/to/code/**

* **Search for a case-insensitive match: grep -i "warning" logfile.txt**
* **Display line numbers for matches: grep -n "failed" logfile.txt**
* **Count the number of matches: grep -c "success" logfile.txt**
* **Invert match (show lines that don’t contain the pattern):**

**grep -v "debug" logfile.txt**

* **Use grep with other commands (pipe):dmesg | grep "usb"**

(Filters the system messages for USB-related logs)

**ps:**The ps command in Unix/Linux is used to display information about active processes running on the system. It's like a snapshot of the processes at a given moment

**Search for a process by name:**ps aux | grep "nginx"

* a: Shows all processes from all users.
* u: Displays processes with user-oriented format.
* x: Includes processes not attached to a terminal.

**Kill a process by PID:** kill <PID>

**Get just the PIDs of a specific process:**ps -C process\_name -o pid=

**SU and SUDO:**

| **Authentication** | Requires root user's password | Requires user's own password |
| --- | --- | --- |
| **Scope** | Switches to root or another user entirely | Runs a **single command** with privileges |
| **Security** | Less secure (shares root password) | More secure (user-specific permissions) |
| **Audit Log** | No command auditing | Commands logged in /var/log/auth.log |
|  |  |  |

**Creating files in Linux:**

### **1.Using touch Command: touch filename.txt**

* **Creates an empty file if it doesn't exist.**
* **Updates the timestamp if the file already exists.**

### **2.Using echo Command:**

### **Creates filename.txt with the specified content.:**

### **echo "Hello, World!" > filename.txt**

**Use >> to append instead of overwrite:**

**echo "New Line" >> filename.txt**

### **3. Using cat Command: cat > filename.txt 4. Using nano, vim, or vi Editors**

**nano filename.txt**

**vim filename.txt**

**vi filename.txt**

* **Creates the file if it doesn’t exist.**
* **Save the file after editing (Ctrl + O in nano, :wq in vim).**

### **df -h Command:**

### *Disk Space Usage in Human-Readable Format*

The df command in Linux is used to display information about disk space usage on file systems. The -h flag stands for "human-readable", which presents the sizes in easily understandable units like KB, MB, GB, or TB instead of bytes.

**Logs in LINUX:**

In Linux, logs are typically stored in the /var/log/ directory and are managed by logging services like rsyslog, journalctl, or logrotate.

### **1. Viewing Logs**

**cat /var/log/syslog**

**tail -f /var/log/syslog**

**less /var/log/syslog (**Use less for large logs with scrolling) **grep "error" /var/log/syslog** (filter logs of error word in logs)