
Chapter 1: Introduction to Linux

---

1. What Is Linux?

Definition (Interview-Oriented)

Linux is an open-source operating system kernel that manages hardware resources and allows software applications to run.

A commonly accepted interview-friendly definition:

Linux is an open-source, Unix-like operating system used widely in servers, cloud platforms, and DevOps environments.

Why Linux Exists and Why It Matters

Computers require an operating system to:

- Manage CPU, memory, disk, and network
- Run applications securely and efficiently

Linux was created to provide:

- Stability
- Security
- Flexibility
- Cost-free usage (open source)

Why interviewers care:

- Most production servers run Linux
- Cloud platforms (AWS, Azure, GCP) are Linux-based
- Containers and CI/CD pipelines rely heavily on Linux

How Linux Works (High-Level View)

Linux acts as an interface between hardware and users.

Basic flow:

```
User → Shell → Kernel → Hardware
```

Users never interact with hardware directly.
The kernel handles all hardware communication.

---

2. Linux vs Unix

Definitions

- Unix: Proprietary operating system originally developed in the 1970s
- Linux: Free and open-source operating system inspired by Unix principles

---

Key Differences

Feature	Unix	Linux
Source code	Closed	Open-source
Cost	Paid	Free
Customization	Limited	Extensive
Usage today	Legacy systems	Servers, cloud, DevOps

Interview note:

Linux is Unix-like but not Unix.

---

3. Linux Architecture

Linux consists of four **major layers**:

1. Hardware
2. Kernel
3. Shell
4. Applications / Users

Architecture Explained with Real-Life Analogy

- **Hardware**: Physical resources (CPU, RAM, disk)
- **Kernel**: Controls and allocates resources
- **Shell**: Interface that accepts user commands
- **User**: Person or application issuing commands

Users interact with the shell, not directly with the kernel or hardware.

---

4. Linux Kernel

Definition

The kernel is the core component of Linux responsible for managing system resources and communicating with hardware.

Responsibilities of the Kernel

- Process management
- Memory management
- Disk and file system access
- Networking
- Device drivers
- Security and access control

Without the kernel, Linux cannot function.

Interview-ready line:

The kernel manages resources and provides controlled access to hardware.

---

5. Kernel Space vs User Space

Definitions

- User Space: Area where user applications run
- Kernel Space: Area where kernel code executes with full privileges

---

Why This Separation Exists

- Improves system security
- Prevents system-wide crashes
- Isolates application failures

If a user application crashes, the operating system remains stable because the kernel is protected.

---

6. What Is a Shell?

Definition

A shell is a command-line interface that allows users to interact with the Linux kernel.

How a Shell Works

```
User types command → Shell interprets → Kernel executes → Output returned
```

Common Shells

- bash
- sh
- zsh
- fish

Interview note:

A shell is an interface, not the kernel itself.

---

7. What Happens When You Run a Command?

Example Command : `ls -l`

Execution Flow

1. Shell reads the command
2. Shell searches for the executable using PATH
3. Shell creates a process
4. Kernel schedules CPU time
5. Kernel accesses the file system
6. Output is sent back to the shell
7. Shell displays the result

This explanation is frequently asked in interviews.

---

8. Process vs Thread vs Task

Definitions

- Process: A running instance of a program
- Thread: A lightweight execution unit inside a process
- Task: Kernel’s internal representation of a process or thread

---

Real-Life Example

- Web browser is a process
- Each browser tab is a thread

Interview line:

A process can contain multiple threads, but each thread belongs to one process.

---

9. What Is a Daemon?

Definition

A daemon is a background process that runs continuously without user interaction.

Examples

- sshd (SSH service)
- crond (cron scheduler)
- nginx or httpd (web server)

Daemon process names often end with the letter “d”.

---

10. Load Average

Definition

Load average represents the average number of processes waiting for CPU or I/O resources.

Interview Explanation

- Load of 1.00 on a single-core system means full utilization
- High load does not always mean high CPU usage
- Disk or I/O waits can also increase load

---

11. Swap Memory

Definition

Swap is disk space used as an extension of physical memory when RAM is exhausted.

Key Points

- Prevents system crashes due to memory exhaustion
- Excessive swap usage significantly slows down performance

---

12. PATH Environment Variable

Definition

PATH is an environment variable that tells the shell where to look for executable commands.

Example

```
echo $PATH
```

If a command is not located in a PATH directory, the shell returns “command not found”.

---

13. umask

Definition

umask defines the default permissions assigned to newly created files and directories.

Interview Explanation

umask subtracts permissions from the system default values.

Chapter 1: Interview Takeaways

After this chapter, you should confidently explain:

- What Linux is and why it is used
- Linux architecture
- Kernel and shell roles
- Command execution flow
- Process, thread, and daemon concepts
- Load, swap, PATH, and umask basics
