Operating System (OS) - Basics Notes

An **Operating System (OS)** is a **software that acts as an interface between the user and computer hardware**. It manages hardware resources and provides services to computer programs.

- Examples: **Windows, Linux, macOS, Android, iOS**

Key Functions:

- Manages **hardware** (CPU, memory, I/O devices)
- Provides a **user interface** (GUI or CLI)
- Runs and manages **applications**
- Ensures **security and access control**

Objectives of Operating System

- **Resource Management:** Efficient use of CPU, memory, and I/O devices
- **User Convenience:** Makes computer easy to use
- **System Efficiency:** Fast, reliable, and stable performance
- **Security:** Protects data from unauthorized access
- **Error Detection & Handling:** Detects and recovers from errors

Types of Operating Systems

A. Based on User Interaction

i. **Batch OS (**1950s – 1960s)
   a. Jobs are collected and processed in **batches**
   b. User does not interact during execution
   c. Example: Early IBM mainframes

ii. **Time-Sharing OS (Multitasking) (**Late 1960s – 1970s)
   a. CPU time is shared among multiple users
   b. Allows multiple tasks to run **simultaneously**
   c. Example: UNIX

iii. **Distributed OS (**1980s – 2000s)
   a. Manages a **group of independent computers** and makes them appear as a single system
   b. Example: Amoeba, LOCUS

iv. **Network OS(**1980s – 1990s)
   a. Provides **network connectivity and file sharing**
   b. Example: Novell NetWare

v. **Real-Time OS (RTOS) (**1990s – Present)
   a. Processes data **immediately**
   b. Example: Medical systems, embedded systems

B. Based on Processing

i. **Single-User, Single-Tasking OS**

a. Only one user can do **one task** at a time
b. Example: DOS

ii. **Single-User, Multi-Tasking OS**
a. One user can do **multiple tasks** simultaneously
b. Example: Windows, macOS

iii. **Multi-User OS (**1980s – Present)
a. Multiple users can use the system **simultaneously**
b. Example: UNIX, Linux

4. Components of Operating System

i. **Kernel**
a. Core part of OS
b. Manages CPU, memory, and devices

ii. **Shell**
a. Interface between **user and kernel**
b. Can be CLI (Command Line Interface) or GUI (Graphical User Interface)

iii. **File System**
a. Organizes data in files and directories

iv. **Device Drivers**
a. Software to control hardware devices

v. **System Utilities**
a. Tools for **system maintenance and monitoring**

5. Functions of Operating System

i. **Process Management**
a. Creates, schedules, and terminates processes

ii. **Memory Management**
a. Allocates memory to processes efficiently

iii. **File Management**
a. Manages storage, access, and organization of files

iv. **Device Management**
a. Controls input/output devices

v. **Security and Protection**
a. Protects system resources and data

6. Advantages of Operating System

- Efficient hardware utilization
- Provides **easy user interface**
- Enables **multitasking**
- Ensures **data security**
- Manages **errors and resources** effectively

On-Premises (On-Prem)

- On-Premises means all **hardware, software, and servers are physically present at your location (office/home)**.

- You manage everything: servers, networking, storage, security.

Example:

- Company installs its own database server and file server inside office.

Advantages:

- Full control over **data and hardware**
- Customization according to company needs
- Security under your own management

Disadvantages:

- High **initial cost** (servers, storage, software licenses)
- Need **IT staff** to maintain and troubleshoot
- Scaling is difficult — need to buy more hardware

Why use it:

- For **sensitive data** or legacy applications that cannot move to cloud.
- Example: Banks, government offices with strict data policies

Hosted Server

- A **hosted server** is a server located at a **data center managed by a third-party company**, but you rent it or lease it.
- You access it remotely but the physical hardware is **not at your premises**.

Example:

- Renting a dedicated server from GoDaddy or HostGator for your website.

Advantages:

- No need to **buy hardware**
- IT maintenance is partly handled by the hosting provider
- More reliable than on-premises if provider has good infrastructure

Disadvantages:

- Less control over hardware compared to on-premises
- Can be expensive if traffic or storage grows
- Dependent on provider's uptime and security

Why use it:

- For **web hosting, applications, or businesses** that want to avoid physical server management but need full server resources.

Cloud Platform

- Cloud platform provides **on-demand computing resources (servers, storage, databases, applications) over the internet**.
- You don't manage physical servers; the cloud provider handles infrastructure.

Example:

- **Amazon Web Services (AWS)**, **Microsoft Azure**, **Google Cloud Platform (GCP)**

Advantages:

- **Scalable** – pay only for what you use
- **Accessible anywhere** via internet
- Minimal **hardware maintenance**
- Supports **backup, disaster recovery, and multi-region deployment**
- Integrates with modern technologies like **AI, ML, IoT**

Disadvantages:

- Dependent on **internet connectivity**
- Some concerns about **data privacy/security**

Why use it:

- For **startups, modern applications, enterprise apps**, and **global reach**
- Example: Netflix, Spotify, and Google use cloud platforms to deliver services worldwide

Real-Life Examples of Cloud Services

i. **Google Drive / OneDrive / Dropbox**
   a. **Type:** Cloud Storage
   b. **Use:** Store files online, access anywhere, share with others

ii. **Netflix / Spotify**
   a. **Type:** Cloud-based Streaming
   b. **Use:** Stream movies, music without downloading; content served via cloud servers

iii. **Gmail / Outlook**
   a. **Type:** Cloud Email (SaaS)
   b. **Use:** Access email from any device via internet

iv. **AWS / Microsoft Azure / Google Cloud Platform**
   a. **Type:** Cloud Computing Platforms (IaaS / PaaS)
   b. **Use:** Host websites, run applications, data storage, AI/ML workloads

AWS Global Infrastructure (Structure)

AWS provides cloud services all over the world.
To do this efficiently, it divides its global system into **three main levels**:

- Region
- Availability Zone (AZ)
- Data Center (DC)

1. Region

- A **Region** is a **geographical area** that contains multiple Availability Zones.
- Each region is completely **independent** from others to ensure reliability and data control.
- Users can choose which region they want their data or services to be hosted in — usually the one **closest to their customers** for **low latency** and **better performance**.

Examples of Regions:

- Asia Pacific (Mumbai) → code: **ap-south-1**
- US East (Ohio) → code: **us-east-2**
- Europe (London) → code: **eu-west-2**

Each Region contains:

- 2 or more **Availability Zones (AZs)**
- Multiple **Data Centers**
- Independent **networking, power, and operations**

Purpose: To provide local access and comply with data privacy or government regulations.

2. Availability Zone (AZ)

- An **Availability Zone** is a **logical group of one or more data centers** inside a region.
- Each AZ has its own **power, cooling, and networking** system.
- AZs in the same region are connected to each other with **high-speed private fiber networks**.

If one AZ fails (for example, due to a natural disaster or power issue), the others continue to work — this provides **high availability**.

Example (Mumbai Region):

- ap-south-1a
- ap-south-1b
- ap-south-1c

Each of these represents a separate AZ within the same **ap-south-1 (Mumbai)** region.

Purpose: To provide redundancy and fault tolerance within a region.

3. Data Center (DC)

- A **Data Center** is a **physical building** that contains AWS servers, networking devices, and storage systems.

- Each Availability Zone has **multiple data centers**.
- Data centers are **redundant** — if one fails, others in the same AZ take over.

Purpose: To actually store and process the data physically.

Hierarchy of AWS Infrastructure

```
World
   ↓
AWS Region (Example: Mumbai)
   ↓
Availability Zones (Example: ap-south-1a, ap-south-1b)
   ↓
Data Centers (Physical server buildings)
```

Real-Life Analogy

Think of AWS as a **school system**:

AWS Concept	Real-Life Example	Meaning
Region	A City	Each city has multiple schools (regions are large areas)
Availability Zone	A School	Each school (AZ) has its own building and systems
Data Center	Classrooms	Each school (AZ) has multiple classrooms (servers)

If one school closes (AZ failure), the others keep running.
That's why AWS services **never stop** — they are built for **high availability** and **fault tolerance**.

Key Points

- Each **Region** works independently.
- **AZs** in the same region are connected with private high-speed networks.
- Each **AZ** has **multiple Data Centers** for backup and stability.
- This structure makes AWS **secure, reliable, and scalable**.

Summary Table

Level	Full Name	Example	Purpose
Region	Geographical area	Asia Pacific (Mumbai) → ap-south-1	Choose data location
Availability Zone	One or more data centers	ap-south-1a, ap-south-1b	Redundancy & high availability
Data Center	Physical building	Servers and storage rooms	Store and process data

Cloud Service Models: IaaS, PaaS, SaaS

Cloud services are delivered in **different models**, depending on **how much control you want** and **how much the provider manages**.

The **three main models** are:

IaaS – Infrastructure as a Service

IaaS provides **virtualized computing resources** over the internet. You get **servers, storage, and networking**, but you manage your **OS, applications, and data**.

Key Features:

- Provides virtual machines, storage, and networks.
- Highly scalable.
- Pay only for what you use (on-demand).
- You control OS, apps, and middleware.

Examples:

- **AWS EC2** – virtual servers
- **Google Compute Engine (GCE)**
- **Microsoft Azure Virtual Machines**

Analogy:
Renting a **furnished apartment**:

- Landlord provides the building, electricity, water (infrastructure).
- You decide furniture, decoration, and how you use it (OS & apps).

PaaS – Platform as a Service

Definition:
PaaS provides a **platform to develop, run, and manage applications** without worrying about the underlying infrastructure.
You focus only on **code and data**, the provider manages OS, servers, storage, and networking.

Key Features:

- Ready-to-use environment for development.
- Automatic scaling and load balancing.
- Ideal for developers to build applications quickly.
- Reduces management overhead.

Examples:

- **Google App Engine**
- **AWS Elastic Beanstalk**

- **Microsoft Azure App Services**

Analogy:
Renting a **fully furnished office with computers and internet**:

- Office provider gives everything needed to work.
- You just bring your team and start working (write code and deploy apps).

SaaS – Software as a Service

Definition:
SaaS delivers **fully functional applications** over the internet.
You don't worry about servers, storage, or code. You just **use the software**.

Key Features:

- Access via browser or app.
- No installation or maintenance required.
- Updates and security handled by provider.

Examples:

- **Gmail** – Email service
- **Google Docs / Microsoft Office 365**
- **Salesforce** – CRM software

Analogy:
Using a **food delivery app**:

- The restaurant (provider) cooks and delivers the food.
- You just eat (use the software), nothing else to manage.

Comparison Table

Feature	IaaS	PaaS	SaaS
Managed by Provider	Infrastructure	Infrastructure + Platform	Everything
Managed by User	OS, Apps, Data	Apps, Data	Only use software
Examples	AWS EC2, Google Compute Engine	AWS Elastic Beanstalk, Google App Engine	Gmail, Salesforce
Control	High	Medium	Low
Use Case	IT admins, system architects	Developers	End users

Computer Networks

A **computer network** is a system in which **two or more computers** are connected to **share resources and communicate** with each other.

Purpose of Networking:

- Share data and files
- Share hardware resources (printers, scanners)
- Internet access
- Communication (email, chat, video calls)

2. Components of a Network

- **Nodes (Devices):** Computers, servers, printers, smartphones
- **Transmission Media:**
  - Wired: Ethernet cables, fiber optics
  - Wireless: Wi-Fi, Bluetooth
- **Switches & Hubs:** Connect multiple devices in a network
- **Routers:** Connect networks to the internet or other networks
- **Network Interface Card (NIC):** Hardware in each device to connect to the network

3. Types of Computer Networks

A. LAN – Local Area Network

- **Definition:** Network covering a **small geographical area** like a home, office, or school.
- **Speed:** High (100 Mbps – 10 Gbps)
- **Ownership:** Usually **private**, managed by an organization
- **Purpose:** Share files, printers, and applications
- **Example:** Office network, school lab network

B. MAN – Metropolitan Area Network

- **Definition:** Network covering a **city or a town**
- **Speed:** Medium to high
- **Ownership:** Usually owned by **service providers**
- **Purpose:** Connect multiple LANs in a city
- **Example:** City-wide Wi-Fi networks, city government network

C. WAN – Wide Area Network

- **Definition:** Network covering a **large geographical area**, often a country or continent
- **Speed:** Lower than LAN, depends on technology
- **Ownership:** Multiple organizations or service providers
- **Purpose:** Connect LANs and MANs over long distances
- **Example:** The Internet, bank networks

D. PAN – Personal Area Network

- **Definition:** Small network for **personal devices**

- **Range:** Usually a few meters
- **Purpose:** Connect personal devices like smartphone, laptop, printer
- **Example:** Bluetooth connection between phone and headphones

E. CAN – Campus Area Network

- **Definition:** Network connecting multiple LANs in a **campus or university**
- **Purpose:** Share resources within a campus
- **Example:** University network connecting labs, library, and offices

IP Address

- Unique identifier for a device on a network.
- Used to route data to the correct device.
- Example: 192.168.1.1

Port

- Communication endpoint on a device.
- Specifies which service or application should receive the data.
- Example: Port 80 → HTTP, Port 443 → HTTPS

Protocol

- Set of rules for communication between devices on a network.
- Ensures data is sent, received, and understood correctly.
- Examples:
    - HTTP → Web pages
    - FTP → File transfer
    - SMTP → Email
    - TCP/IP → Reliable data transmission

Firewall

A **firewall** is a **network security device or software** that monitors and controls the incoming and outgoing network traffic based on predefined security rules. It acts as a barrier between a **trusted internal network** (like a company LAN) and an **untrusted external network** (like the Internet).

Purpose

- **Protect the network** from unauthorized access.
- **Prevent attacks** such as hacking, viruses, and malware.
- **Control traffic**: Allows only legitimate data and blocks harmful traffic.

How a Firewall Works

- **Traffic Monitoring:** Checks all data packets entering or leaving the network.

- **Rule Enforcement:** Compares each packet against a set of security rules.
- **Decision Making:**
  - **Allow:** If the traffic meets security policies.
  - **Block:** If the traffic is suspicious or violates rules.
- **Logging & Alerts:** Records suspicious activities and can notify administrators.

Types of Firewalls

1. **Based on Implementation**
   - **Hardware Firewall:**
     - Physical device installed between the network and the Internet.
     - Usually used in offices and enterprises.
     - Example: Cisco ASA, Fortinet.
   - **Software Firewall:**
     - Installed on individual computers or servers.
     - Controls traffic to and from that device.
     - Example: Windows Defender Firewall.
2. **Based on Filtering Method**
   - **Packet Filtering Firewall:**
     - Examines packets at the network layer (IP, port, protocol).
     - Fast but less secure.
   - **Stateful Inspection Firewall:**
     - Tracks the state of active connections.
     - Allows packets only if they are part of a legitimate session.
   - **Proxy Firewall (Application Firewall):**
     - Acts as an intermediary between user and Internet.
     - Examines application-level data (HTTP, FTP).
     - Example: Web filtering.
   - **Next-Generation Firewall (NGFW):**
     - Combines traditional firewall features with **deep packet inspection**, **intrusion prevention**, and **application awareness**.

Router

A **router** is a networking device that connects multiple networks together and forwards data packets between them.
It determines the best path for data to travel from the source to the destination.

Functions of Router:

- Connects **different networks** (for example, LAN to WAN or Internet).
- Routes data packets based on **IP addresses**.
- Assigns **IP addresses** to devices using **DHCP (Dynamic Host Configuration Protocol)**.
- Acts as a **gateway** between local and external networks.
- Provides **network security** and can include **firewall functions**.

Works On:

- **Network Layer (Layer 3)** of the **OSI Model**.

Example:

- A Wi-Fi router that connects multiple devices in a home network to the Internet.

Types of Routers:

- **Wired Router:** Uses cables for connection.
- **Wireless Router:** Uses Wi-Fi signals.
- **Core Router:** Used in large organizations or ISPs.

Switch

A **switch** is a networking device that connects multiple devices (like computers and printers) within the same network (LAN) and allows them to communicate efficiently.

Functions of Switch:

- Connects **devices** within a **Local Area Network (LAN)**.
- Uses **MAC addresses** to identify devices.
- Transfers data only to the **specific destination device**, not to all.
- Reduces network congestion and increases efficiency.
- Manages data traffic within a LAN.

Works On:

- **Data Link Layer (Layer 2)** of the **OSI Model**.

Example:

- A switch connects multiple computers and printers in an office so that they can share files or resources.