

Introduction to Operating Systems

Introduction to Operating Systems

Course notes – concise, organized, and ready for review

1. What Is an Operating System (OS)?

Aspect	Description
Definition	System software that manages hardware resources and provides services for application programs.
Primary Goals	• Resource management • User convenience • System security & stability
Typical Users	End users, developers, system administrators, embedded devices.
Examples	Windows, macOS, Linux, Android, iOS, FreeRTOS, QNX.

2. Core Functions of an OS

1. Process Management

- Creation, scheduling, termination of processes/threads.
- Context switching, inter-process communication (IPC).

2. Memory Management

- Allocation & deallocation of RAM.
- Virtual memory, paging, segmentation, swapping.

3. File System Management

- Organization of data on storage media.
- Directory structures, permissions, mounting.

4. Device (I/O) Management

Introduction to Operating Systems

- Drivers translate generic OS calls to hardware-specific actions.
- Buffering, caching, spooling.

5. Security & Protection

- Authentication, access control lists (ACLs), encryption.
- Isolation of processes (user vs. kernel mode).

6. User Interface

- Command-line interface (CLI) – e.g., Bash, PowerShell.
- Graphical user interface (GUI) – windows, icons, menus.

3. OS Architecture Models

Model	Key Characteristics
Monolithic Kernel	All services run in kernel space; fast but large.
Microkernel	Minimal kernel (scheduling, IPC, basic memory); other services run in user space.
Hybrid Kernel	Combines monolithic performance with microkernel modularity (e.g., Windows NT).
Exokernel	Exposes hardware resources directly; OS provides library OSes.
Layered (Modular) Architecture	System built as a stack of layers; each layer uses services of the one below.

4. Process vs. Thread

Concept	Process	Thread
Definition	Independent execution unit with its own address space.	Lightweight unit of execution sharing a process's address space.
Overhead	Higher (context switch, memory).	Lower (share code, data, files).
Communication	Requires IPC mechanisms.	Implicit via shared memory.
Use Cases	Isolated tasks, security boundaries.	Parallelism within a single application.

Introduction to Operating Systems

5. Scheduling Basics

1. Scheduling Goals

- Maximize CPU utilization.
- Minimize response time & waiting time.
- Ensure fairness and meet deadlines (real-time).

2. Types of Schedulers

- **Long-term (admission)** – decides which jobs enter the ready queue.
- **Short-term (CPU)** – selects the next process to run.
- **Medium-term (swapper)** – moves processes between main memory & swap.

3. Common Algorithms

- **FCFS (First-Come, First-Served)** – simple, non-preemptive.
- **SJF (Shortest Job First)** – optimal average waiting time, needs job length prediction.