

Introduction to Operating Systems

Introduction to Operating Systems

Course level: Undergraduate – Foundations of Computer Science

1. What Is an Operating System (OS)?

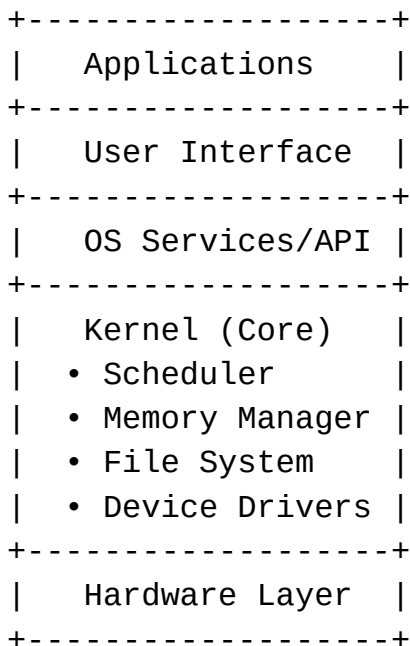
Aspect	Description
Definition	System software that manages hardware resources and provides services for application programs.
Primary Goals	• Efficient resource utilization • System stability and security • User convenience and abstraction
Typical Users	• End users (via GUIs, shells) • Application developers (via APIs) • System administrators (via tools & scripts)

2. Core Functions of an OS

Function	Key Responsibilities	Example Mechanisms
Process Management	Create, schedule, synchronize, and terminate processes/threads.	Scheduler, context switch, inter-process communication (IPC) primitives.
Memory Management	Allocate/deallocate memory, provide isolation, handle paging/swapping.	Virtual memory, page tables, heap/stack allocation.
File System Management	Organize, store, retrieve, and protect data on storage devices.	Hierarchical directories, inode structures, journaling.
Device Management	Abstract hardware devices, mediate I/O operations.	Device drivers, interrupt handling, DMA.
Security & Protection	Enforce access control, authentication, isolation.	User IDs, permissions, security tokens, sandboxing.
User Interface	Provide a way for humans to interact with the system.	Command-line shells, graphical desktops, APIs.

Introduction to Operating Systems

3. System Architecture Overview



- **Kernel** – central part running in privileged mode (kernel mode).
 - **User Space** – applications and most services run in unprivileged mode (user mode).
-

4. Types of Operating Systems

Category	Characteristics	Typical Examples
Batch OS	Jobs processed without user interaction; queue-based.	Early IBM mainframes.
Time-Sharing (Multitasking) OS	Multiple users share CPU via time slicing.	Unix, early versions of Windows.
Real-Time OS (RTOS)	Guarantees response within strict timing constraints.	VxWorks, QNX, FreeRTOS.
Distributed OS	Manages a collection of independent computers as a single system.	Google's Borg, Apache Hadoop (framework).
Embedded OS	Tailored for specific hardware with limited resources.	Embedded Linux, Zephyr.
Mobile OS	Optimized for touch input, power management, app sandboxing.	Android, iOS.

Introduction to Operating Systems

5. Process vs. Thread

Concept	Process	Thread
Definition	An executing program with its own address space.	A lightweight unit of execution within a process; shares address space.
Overhead	Higher (context switch involves memory map).	Lower (shares resources, faster switch).
Isolation	Strong – crash of one process does not affect others.	Weak – bug in one thread can corrupt whole process.
Use Cases	Running separate applications, security boundaries.	Parallel tasks