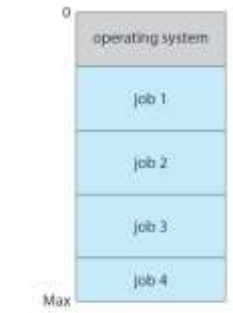
Unit I: Overview of Operating System

Operating-System Structure:

- An operating system provides the environment within which programs are executed. Internally, operating systems vary greatly in their makeup, One of the most important aspects of operating systems is the ability to multiprogram. A single program cannot, in general, keep either the CPU or the I/O devices busy at all times. Single users frequently have multiple programs running.
- **Multiprogramming** increases CPU utilization by organizing jobs (code and data) so that the CPU always has one to execute.
- Given Diagram illustrates the Jobs & Memory layout:
- The operating system keeps several jobs in memory simultaneously Since, in general, main memory is too small to accommodate all jobs, the jobs are kept initially on the disk in the **job pool**.
- This pool consists of all processes residing on disk awaiting allocation of main memory. The set of jobs in memory can be a subset of the jobs kept in the job pool. The operating system picks and begins to execute one of the jobs in memory.
- Eventually, the job may have to wait for some task, such as an I/O operation, to complete. In a non-multiprogrammed system, the CPU would sit idle. In a multiprogrammed system, the operating system simply switches to, and executes, another job. When ***that*** job needs to wait, the CPU switches to ***another*** job, and so on. Eventually, the first job finishes waiting and gets the CPU back. As long as at least one job needs to execute, the CPU is never idle.



Memory layout for a multiprogramming system

Operating-System Operations

- As mentioned earlier, modern operating systems are **interrupt driven**. If there are no processes to execute, no I/O devices to service, and no users to whom to respond, an operating system will sit quietly, waiting for something to happen.
- Events are almost always signaled by the occurrence of an interrupt or a trap.
- A **trap** (or an **exception**) is a software-generated interrupt caused either by an error (for example, division by zero or invalid memory access) or by a specific request from a user program that an operating-system service be performed
- Since the operating system and the users share the hardware and software resources of the computer system, we need to make sure that an error in a user program could cause problems only for the one program running.
- With sharing, many processes could be adversely affected by a bug in one program.
- For example, if a process gets stuck in an infinite loop, this loop could prevent the correct operation of many other processes. More subtle errors can occur in a multiprogramming system, where one erroneous program might modify another program, the data of another program, or even the operating system itself.
- A properly designed operating system must ensure that an incorrect (or malicious) program cannot cause other programs to execute incorrectly.

Dual-Mode and Multimode Operation

- In order to ensure the proper execution of the operating system, we must be able to distinguish between the execution of operating-system code and user defined code.
- The approach taken by most computer systems is to provide hardware support that allows us to differentiate among various modes of execution.
- At the very least, we need two separate ***modes*** of operation: **user mode** and **kernel mode**
- A bit, called the **mode bit**, is added to the hardware of the computer to indicate the current mode: kernel (0) or user (1).

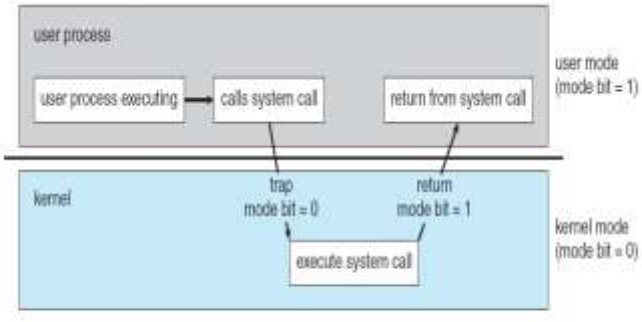


Figure 1.10 Transition from user to kernel mode.

- When the computer system is executing on behalf of a user application, the system is in user mode. However, when a user application requests a service from the operating system (via a system call), the system must transition from user to kernel mode to fulfill the request.
- When the computer system is executing on behalf of a user application, the system is in user mode. However, when a user application requests a service from the operating system (via a system call), the system must transition from user to kernel mode to fulfill the request.

Process Management

- A program does nothing unless its instructions are executed by a CPU. A program in execution, as mentioned, is a process. A time-shared user program such as a compiler is a process. A word-processing program being run by an individual user on a PC is a process.
- A process needs certain resources—including CPU time, memory, files, and I/O devices—to accomplish its task. These resources are either given to the process when it is created or allocated to it while it is running.
- In addition to the various physical and logical resources that a process obtains when it is created, various initialization data (input) may be passed along.
- For example, consider a process whose function is to display the status of a file on the screen of a terminal. The process will be given the name of the file as an input and will execute the appropriate instructions and system calls to obtain and display the desired information on the terminal.
- When the process terminates, the operating system will reclaim any reusable resources.
- A program is a ***passive*** entity, like the contents of a file stored on disk, whereas a process is an ***active*** entity. A single-threaded process has one **program counter** specifying the next instruction to execute.
- A process is the unit of work in a system. A system consists of a collection of processes, some of which are operating-system processes (those that execute system code) and the rest of which are user processes (those that execute user code). All these processes can potentially execute concurrently—by multiplexing on a single CPU,
- **The operating system is responsible for the following activities in connection with process management:**

✓ Scheduling processes and threads on the CPUs

✓ Creating and deleting both user and system processes

✓ Suspending and resuming processes

✓ Providing mechanisms for process synchronization

✓ Providing mechanisms for process communication

Memory Management

- The main memory is central to the operation of a modern computer system. Main memory is a large array of bytes, ranging in size from hundreds of thousands to billions. Each byte has its own address.
- Main memory is a repository of quickly accessible data shared by the CPU and I/O devices.
- The central processor reads instructions from main memory during the instruction-fetch cycle and both reads and writes data from main memory during the data-fetch cycle
- The main memory is generally the only large storage device that the CPU is able to address and access directly. For example, for the CPU to process data from disk, those data must first be transferred to main memory by CPU-generated I/O calls. In the same way, instructions must be in memory for the CPU toexecute them.

Memory Management

- The operating system is responsible for the following activities in connection with memory management:

➢ Keeping track of which parts of memory are currently being used and who is using them

➢ Deciding which processes (or parts of processes) and data to move into and out of memory

➢ Allocating and deallocating memory space as needed

Storage Management

- The operating system abstracts from the physical properties of its storage devices to define a logical storage unit, the **file**. The operating system maps files onto physical media and accesses these files via the storage devices.
- It is Categorized in two types

1. File –System Management

2. Mass Storage Management

- **File-System Management**

- File management is one of the most visible components of an operating system. Computers can store information on several different types of physical media. Magnetic disk, optical disk, and magnetic tape are the most common.
- Each of these media has its own characteristics and physical organization
- Each medium is controlled by a device, such as a disk drive or tape drive, that also has its own unique characteristics. These properties include access speed, capacity, data-transfer rate, and access method
- A file is a collection of related information defined by its creator. Commonly, files represent programs (both source and object forms) and data. Data files may be numeric, alphabetic, alphanumeric, or binary. Files may be free-form (for example, text files), or they may be formatted rigidly (for example, fixed fields).
- Clearly, the concept of a file is an extremely general one.
- The operating system is responsible for the following activities in connection with file management:
  - Creating and deleting files
  - Creating and deleting directories to organize files
  - Supporting primitives for manipulating files and directories
  - Mapping files onto secondary storage
  - Backing up files on stable (nonvolatile) storage media

Mass-Storage Management

- main memory is too small to accommodate all data and programs, and because the data that it holds are lost when power is lost, the computer system must provide secondary storage to back up main memory. Most modern computer systems use disks as the principal on-line storage medium for both programs and data
- Most programs—including compilers, assemblers, word processors, editors, and formatters—are stored on a disk until loaded into memory. They then use the disk as both the source and destination of their processing.
- Hence, the proper management of disk storage is of central importance to a computer system. The operating system is responsible for the following activities in connection with disk management:

- **Free-space management**
- **Storage allocation**
- **Disk scheduling**

Protection and Security:

- If a computer system has multiple users and allows the concurrent execution of multiple processes, then access to data must be regulated.
- For that purpose, mechanisms ensure that files, memory segments, CPU, and other resources can be operated on by only those processes that have gained proper authorization from the operating system.
- For example, memory-addressing hardware ensures that a process can execute only within its own address space.
- The timer ensures that no process can gain control of the CPU without eventually resigning control.
- Device-control registers are not accessible to users, so the integrity of the various peripheral devices is protected.
- **Protection**, then, is any mechanism for controlling the access of processes or users to the resources defined by a computer system. This mechanism must provide means to specify the controls to be imposed and to enforce the controls.
- A system can have adequate protection but still be prone to failure and allow inappropriate access. Consider a user whose authentication information (her means of identifying herself to the system) is stolen. Her data could be copied or deleted, even though file and memory protection are working. It is the job of **security** to defend a system from external and internal attacks. Such attacks spread across a huge range and include viruses and worms, denial-of serviceAttacks
- identity theft, and theft of service (unauthorized use of a system). Prevention of some of these attacks is considered an operating-system function on some systems

Distributed Systems

- A distributed system is a collection of physically separate, possibly heterogeneous, computer systems that are networked to provide users with access to the various resources that the system maintains.
- Access to a shared resource increases computation speed, functionality, data availability, and reliability. Some operating systems generalize network access as a form of file access, with the details of networking contained in the network interface's device driver.

- Others make users specifically invoke network functions. Generally, systems contain a mix of the two modes—for example FTP and NFS. The protocols that create a distributed system can greatly affect that system's utility and popularity.

Distributed Systems

- A **network**, in the simplest terms, is a communication path between two or more systems. Distributed systems depend on networking for their functionality.
- **TCP/IP** is the most common network protocol, and it provides the fundamental architecture of the Internet. Most operating systems support TCP/IP, including all general-purpose ones.
- Networks are characterized based on the distances between their nodes.
- A **local-area network (LAN)** connects computers within a room, a building, or a campus.
- A **wide-area network (WAN)** usually links buildings, cities, or countries. A global company may have a WAN to connect its offices worldwide.
- The continuing advent of new technologies brings about new forms of networks. For example, a **metropolitan-area network (MAN)** could link buildings within a city.
- Blue Tooth and 802.11 devices use wireless technology to communicate over a distance of several feet, in essence creating a **personal-area network (PAN)** between a phone and a headset or a smartphone and a desktop computer.

Computing Environments

- How operating systems are used in a variety of computing environments.
  - Traditional Computing
  - Mobile Computing
  - Distributed Systems
  - Client–Server Computing
  - Peer-to-Peer Computing
  - Virtualization
  - Cloud Computing
  - Real-Time Embedded Systems

Traditional Computing

- As computing has matured, the lines separating many of the traditional computing environments have blurred. Consider the "typical office environment." Just a few years ago, this environment consisted of PCs connected to a network, with servers providing file and print services.
- Remote access was awkward, and portability was achieved by use of laptop computers. Terminals attached to mainframes were prevalent at many companies as well, with even fewer remote access and portability options.

Mobile Computing

- **Mobile computing** refers to computing on handheld smartphones and tablet
- computers. These devices share the distinguishing physical features of being portable and lightweight.
- Historically, compared with desktop and laptop computers, mobile systems gave up screen size, memory capacity, and overall functionality in return for handheld mobile access to services such as e-mail and web browsing.
- Over the past few years, however, features on mobile devices have become so rich that the distinction in functionality between, say, a consumer laptop and a tablet computer may be difficult to discern.

Client–Server Computing

- As PCs have become faster, more powerful, and cheaper, designers have shifted away from centralized system architecture. Terminals connected to centralized systems are now being supplanted by PCs and mobile devices.
- Correspondingly, user-interface functionality once handled directly by centralized systems is increasingly being handled by PCs, quite often through a web interface.
- As a result, many of today's systems act as **server systems** to satisfy requests generated by **client systems**. This form of specialized distributed system, called a **client–server** system

Peer-to-Peer Computing

- Another structure for a distributed system is the peer-to-peer (P2P) system model. In this model, clients and servers are not distinguished from one another.
- Instead, all nodes within the system are considered peers, and each may act as either a client or a server, depending on whether it is requesting or providing a service.
- Peer-to-peer systems offer an advantage over traditional client-server systems. In a client-server system, the server is a bottleneck; but in a peer-to-peer system, services can be provided by several nodes distributed throughout the network.

Virtualization

- Virtualization is a technology that allows operating systems to run as applications within other operating systems.
- At first blush, there seems to be little reason for such functionality. But the virtualization industry is vast and growing, which is a testament to its utility and importance.

Cloud Computing

- **Cloud computing** is a type of computing that delivers computing, storage, and even applications as a service across a network.
- In some ways, it's a logical extension of virtualization, because it uses virtualization as a base for its functionality.
- For example, the Amazon Elastic Compute Cloud **(EC2)** facility has thousands of servers, millions of virtual machines, and petabytes of storage available for use by anyone on the Internet.
- Users pay per month based on how much of those resources they use.

Real-Time Embedded Systems

- Embedded computers are the most prevalent form of computers in existence.
- These devices are found everywhere, from car engines and manufacturing robots to DVDs and microwave ovens.
- They tend to have very specific tasks. The systems they run on are usually primitive, and so the operating systems provide limited features.
- Usually, they have little or no user interface, preferring to spend their time monitoring and managing hardware devices, such as automobile engines and robotic arms.

Assignments:

- Define Operating system & explain what operating system Do?
- Explain Computer System Organization in Brief.
- Explain Computer System Architecture. (OR)
  - Explain Single Processor System
  - Explain Multiprocessor system
- Explain Operating System Structure
- Explain in brief Operating system Operations
- Write Short Notes On:
  - Process Management
  - Memory Management
  - Storage Management
  - Distributed System