Operating Systems – BCS303

Course Coordinator:

Dr. Neethi M V

Assistant Professor

Dept. of CSE-DS

ATMECE, Mysuru

Module 1 – Introduction to operating systems System structures

Session 3

- Operating System operations
- Process Management
- Memory Management
- Storage Management
- I/O System
- Protection and Security

Operating-System Operations

- 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 signaled by the occurrence of an **interrupt** or a **trap**.
- A **trap** (or an exception) is a software-generated interrupt. For each type of interrupt, separate segments of code in the operating system determine what action should be taken.
- An interrupt service routine is provided that is responsible for dealing with the interrupt.

Dual-Mode Operation

- Since the operating system and the user programs share the hardware and software resources of the computer system, it has to be made sure that an error in a user program cannot cause problems to other programs and the Operating System running in the system.

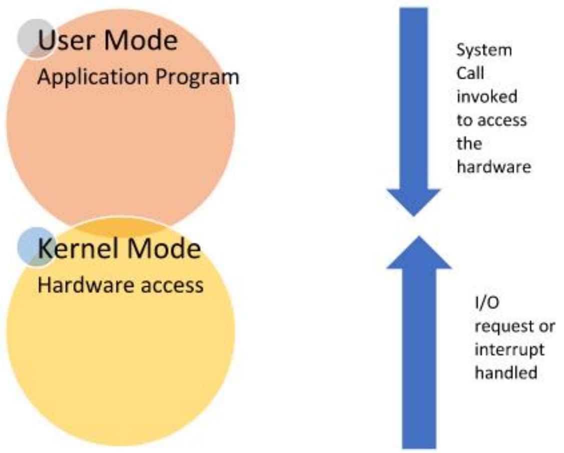
- The approach taken is to use a hardware support that allows us to differentiate among various modes of execution.

- The system can be assumed to work in two separate modes of operation:
  1. User mode
  2. Kernel mode (supervisor mode, system mode, or privileged mode).

Dual-Mode Operation

- A hardware bit of the computer, called the mode bit, is used to indicate the current mode: kernel (0) or user (1). With the mode bit, we are able to distinguish between a task that is executed by the operating system and one that is executed by the user.

- When the computer system is executing a user application, the system is in user mode. When a user application requests a service from the operating system (via a system call), the transition from user to kernel mode takes place.



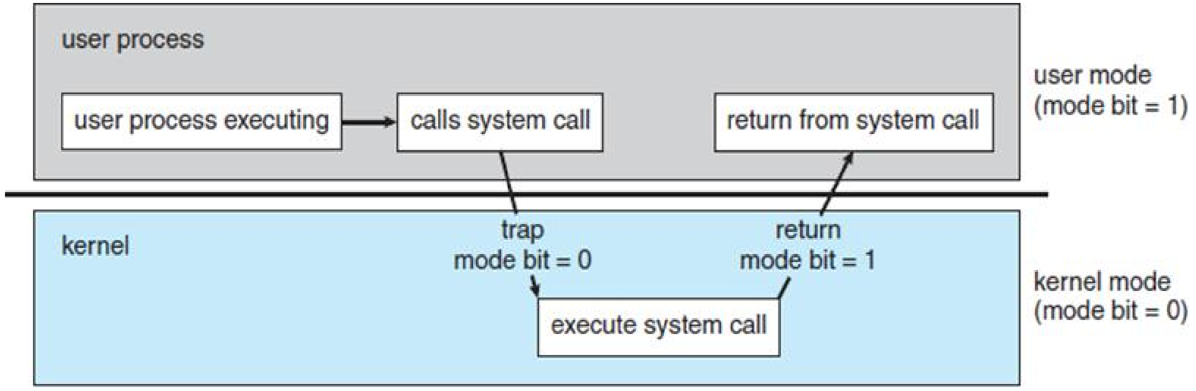


Figure Transition from user to kernel mode.

Dual-Mode Operation

- At system boot time, the hardware starts in kernel mode.
- The operating system is then loaded and starts user applications in user mode.
- Whenever a trap or interrupt occurs, the hardware switches from user mode to kernel mode (that is, changes the mode bit from 1 to 0). Thus, whenever the operating system gains control of the computer, it is in kernel mode.
- The dual mode of operation provides us with the means for protecting the operating system from errant users—and errant users from one another.

Dual-Mode Operation

- The hardware allows privileged instructions to be executed only in kernel mode. If an attempt is made to execute a privileged instruction in user mode, the hardware does not execute the instruction but rather treats it as illegal and traps it to the operating system. The instruction to switch to user mode is an example of a privileged instruction.

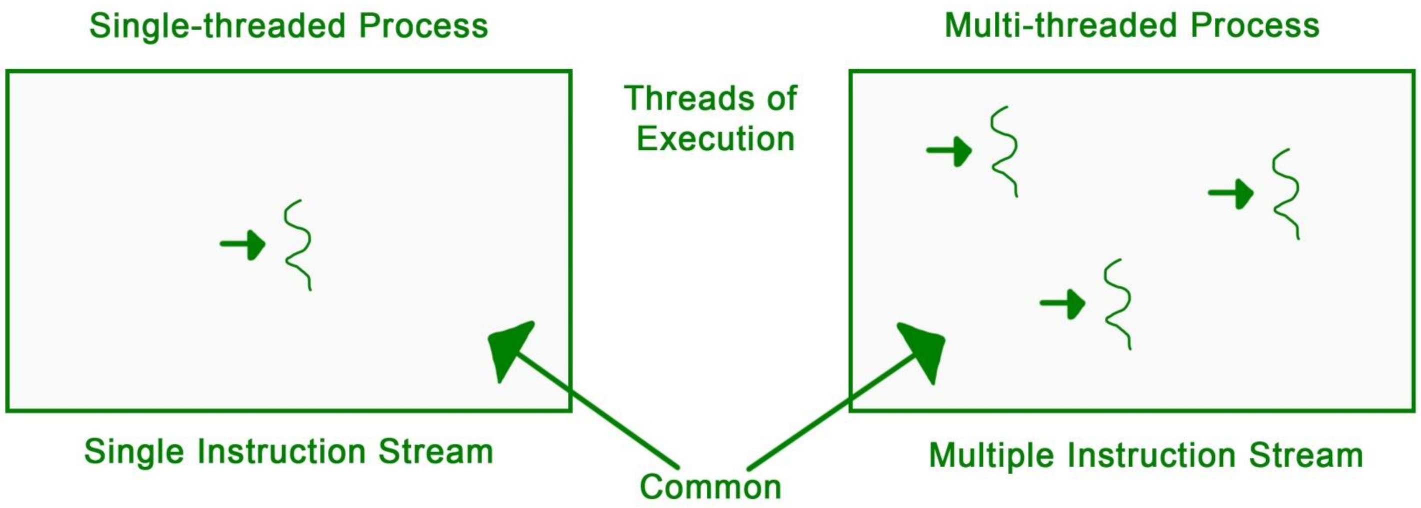
- Initial control is within the operating system, where instructions are executed in kernel mode. When control is given to a user application, the mode is set to user mode. Eventually, control is switched back to the operating system via an interrupt, a trap, or a system call.

Process Management

- A program under execution is a process.
- A process needs resources like CPU time, memory, files, and I/O devices for its execution. These resources are given to the process when it is created or at run time.
- When the process terminates, the operating system reclaims the resources. The program stored on a disk is a **passive entity** and the program under execution is an **active entity**.
- A **single-threaded process** has one program counter specifying the next instruction to execute. The CPU executes one instruction of the process after another, until the process completes.
- A **multithreaded process** has multiple program counters, each pointing to the next instruction to execute for a given thread.

Process Management

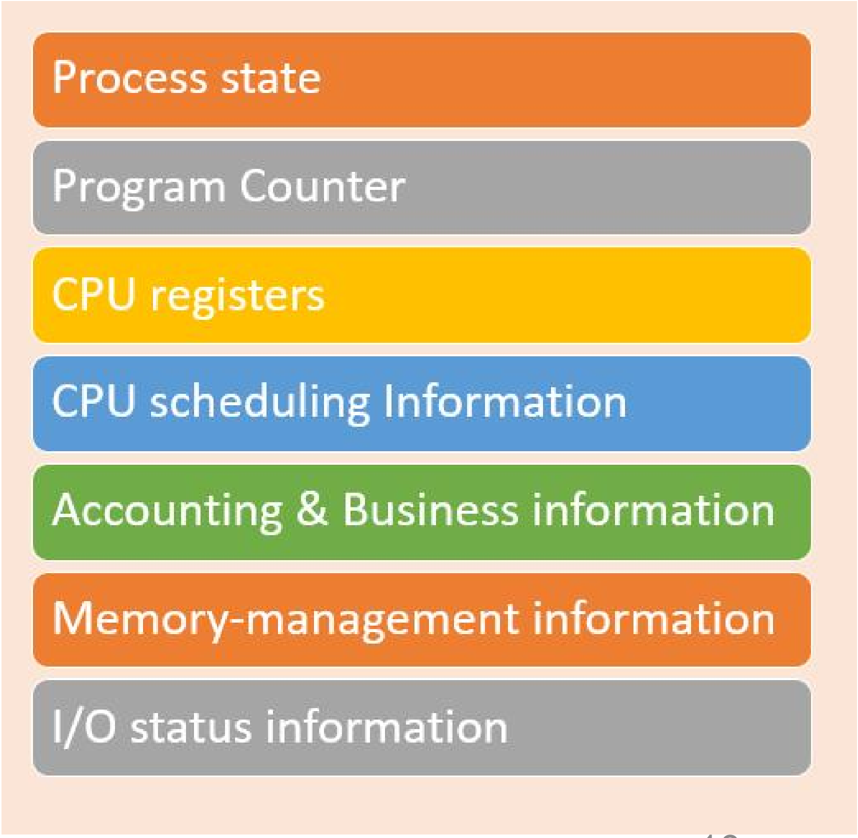
- The concept of multi-threading needs proper understanding of these two terms – a **process** and a **thread**. A process is a program being executed. A process can be further divided into independent units known as threads. A thread is like a small light-weight process within a process. Or we can say a collection of threads is what is known as a process.



Single Thread and Multi Thread Process

Process Management

- The operating system is responsible for the following activities in connection with process management:
  - Scheduling process and threads on the CPU
  - Creating and deleting both user and system processes
  - Suspending and resuming processes
  - Providing mechanisms for process synchronization
  - Providing mechanisms for process communication

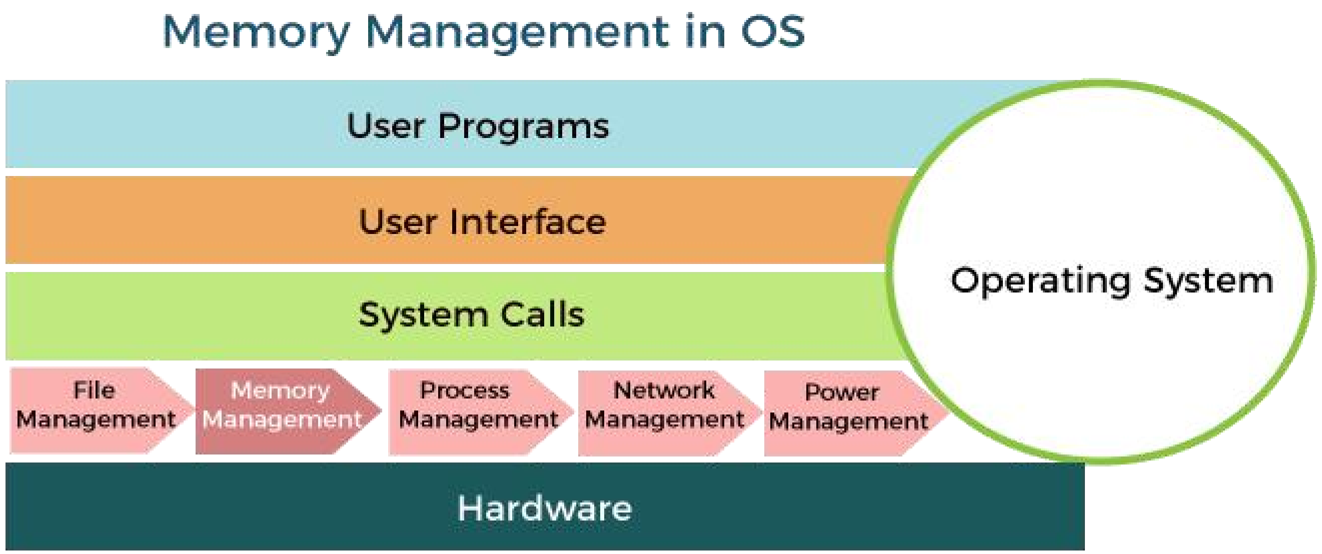


Memory Management

- Main memory is a large array of words or bytes. Each word or byte has its own address.
- Main memory is the storage device which can be easily and directly accessed by the CPU.
- As the program executes, the central processor reads instructions and also reads and writes data from main memory.
- To improve both the utilization of the CPU and the speed of the computer's response to its users, general-purpose computers must keep several programs in memory, creating a need for memory management.

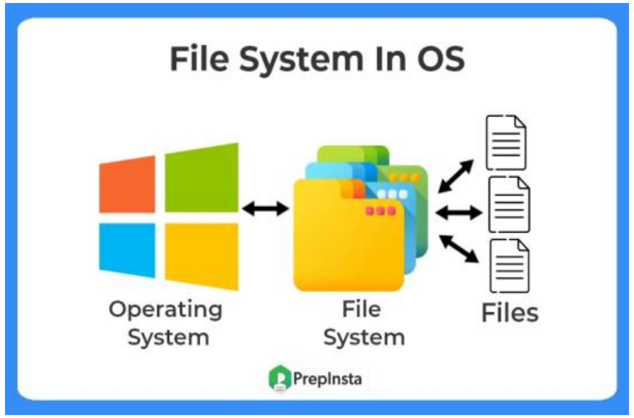
Memory Management

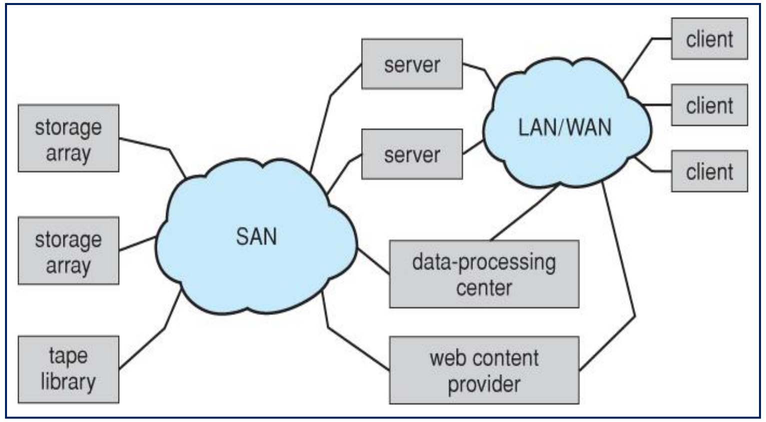
- The operating system is responsible for the following activities in connection with memory management:
  - Keeping track of which parts of memory are currently being used by user.
  - Deciding which processes and data to move into and out of memory.
  - Allocating and deallocating memory space as needed.

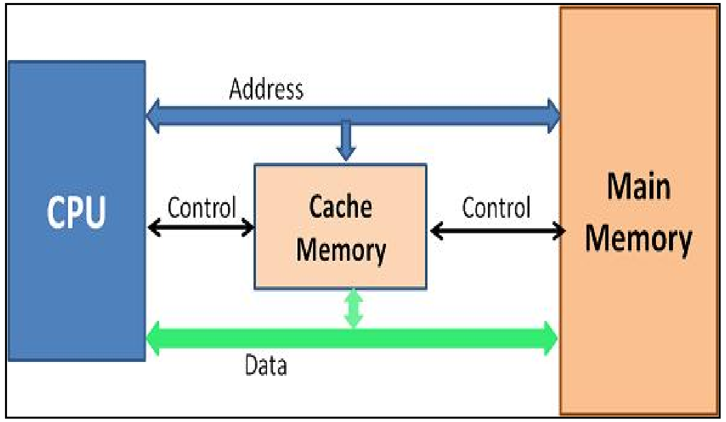


Storage Management

- There are three types of storage management
  - File system management
  - Mass-storage management
  - Cache management







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.
- A file is a collection of related information defined by its creator. Commonly, files represent programs 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).

Storage Management - File-System Management

- The operating system implements the abstract concept of a file by managing mass storage media. Files are normally organized into directories to make them easier to use.

- When multiple users have access to files, it may be desirable to control by whom and in what ways (read, write, execute) files may be accessed.

- 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

Storage Management - Mass-Storage Management

- As the main memory is too small to accommodate all data and programs, and as the data that it holds are erased when power is lost, the computer system must provide secondary storage to back up main memory.
- Most modern computer systems use disks as the 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 and 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.

Storage Management - Mass-Storage Management

- The operating system is responsible for the following activities in connection with disk management:
    - Free-space management
    - Storage allocation
    - Disk scheduling

- As the secondary storage is used frequently, it must be used efficiently. The entire speed of operation of a computer may depend on the speeds of the disk. Magnetic tape drives and their tapes, CD, DVD drives and platters are **tertiary storage** devices.

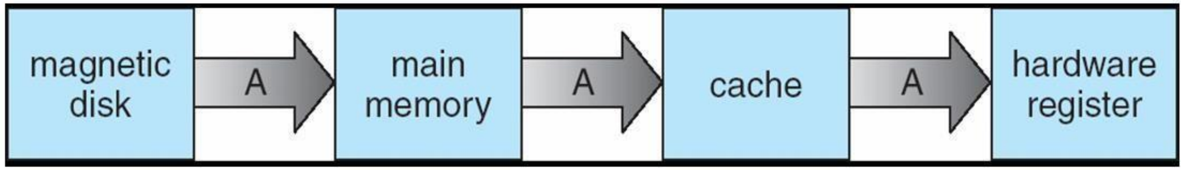
- The functions that operating systems provides include mounting and unmounting media in devices, allocating and freeing the devices for exclusive use by processes, and migrating data from secondary to tertiary storage.

Storage Management - Caching

- Caching is an important principle of computer systems. Information is normally kept in some storage system (such as main memory).
- As it is used, it is copied into a faster storage system— the cache—as temporary data.
- When a particular piece of information is required, first we check whether it is in the cache. If it is, we use the information directly from the cache; if it is not in cache, we use the information from the source, putting a copy in the cache under the assumption that we will need it again soon.
- Because caches have limited size, cache management is an important design problem. Careful selection of the cache size and page replacement policy can result in greatly increased performance.

Storage Management - Caching

- The movement of information between levels of a storage hierarchy may be either explicit or implicit, depending on the hardware design and the controlling operating-system software.
- For instance, data transfer from cache to CPU and registers is usually a hardware function, with no operating-system intervention.
- In contrast, transfer of data from disk to memory is usually controlled by the operating system.
- In a hierarchical storage structure, the same data may appear in different levels of the storage system. For example, suppose to retrieve an integer A from magnetic disk to the processing program.



Storage Management - Caching

- The operation proceeds by first issuing an I/O operation to copy the disk block on which A resides to main memory. This operation is followed by copying A to the cache and to an internal register. Thus, the copy of A appears in several places: on the magnetic disk, in main memory, in the cache, and in an internal register.

- In a multiprocessor environment, in addition to maintaining internal registers, each of the CPUs also contains a local cache. In such an environment, a copy of A may exist simultaneously in several caches.

- Since the various CPUs can all execute concurrently, any update done to the value of A in one cache is immediately reflected in all other caches where A resides. This situation is called **cache coherency**, and it is usually a hardware problem (handled below the operating-system level).

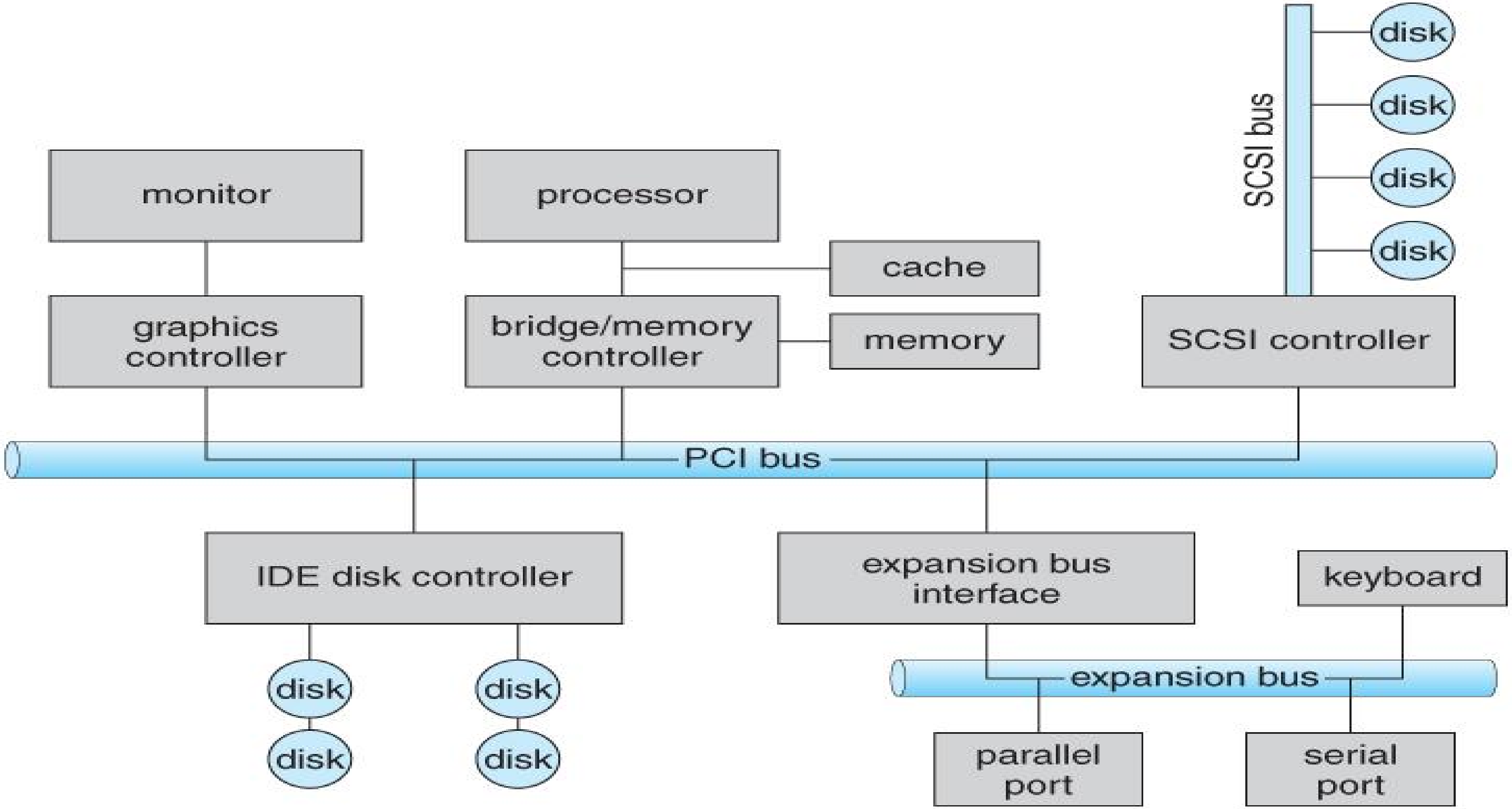
I/O Systems

- One of the purposes of an operating system is to hide the peculiarities of specific hardware devices from the user.

- The I/O subsystem consists of several components:
  - A memory-management component that includes buffering, caching, and spooling
  - A general device-driver interface
  - Drivers for specific hardware devices

- Only the device driver knows the peculiarities of the specific device to which it is assigned.

I/O Systems

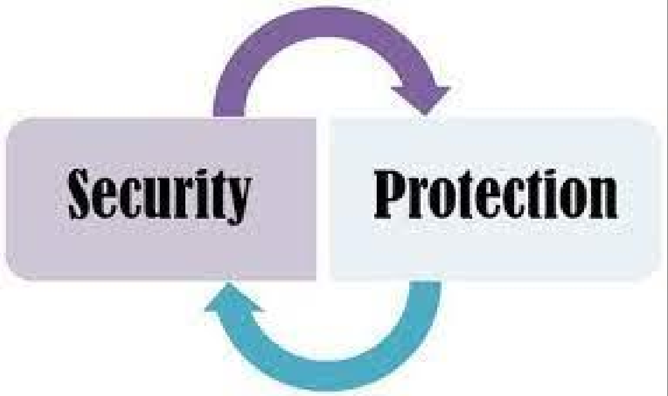

disk
disk
disk
disk
SCSI bus
monitor
processor
cache
graphics controller
bridge/memory controller
memory
SCSI controller
PCI bus
IDE disk controller
expansion bus interface
keyboard
disk
disk
disk
disk
expansion bus
parallel port
serial port

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 for a long time. Device-control registers are not accessible to users, so the integrity of the various peripheral devices is protected.

- **Protection** is a mechanism for controlling the access of processes or users to the resources defined by a computer system. This mechanism must provide means for specification of the controls to be imposed and means for enforcement.



Protection and Security

- Protection improves reliability. A protection-oriented system provides a means to distinguish between authorized and unauthorized usage. A system can have adequate protection but still be prone to failure and allow inappropriate access.

- Consider a user whose authentication information 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 service attacks etc.

- Protection and security require the system to be able to distinguish among all its users. Most operating systems maintain a list of user names and associated **user identifiers (user IDs).** When a user logs in to the system, the authentication stage determines the appropriate user ID for the user.

Protection and Security

Overview of Security and Protection (continued)

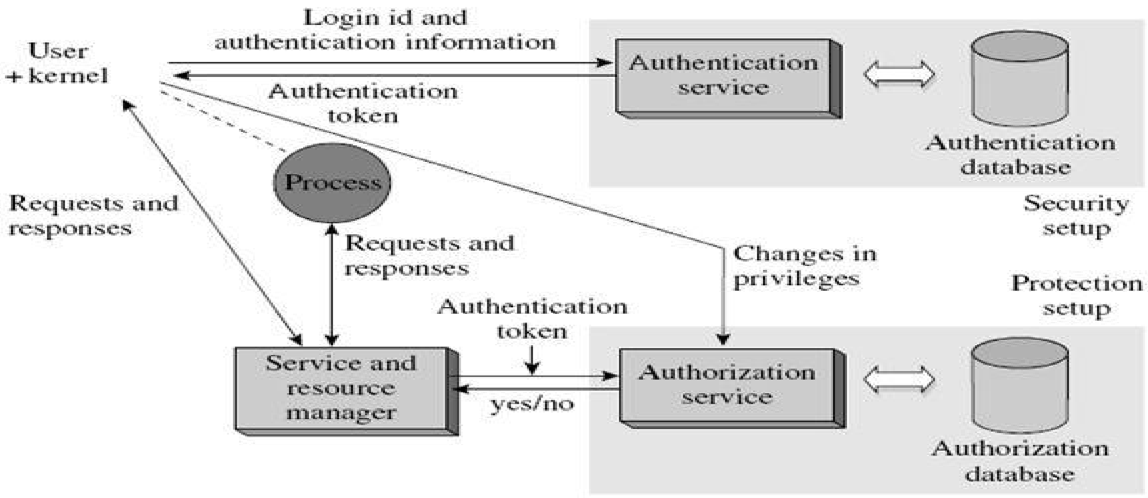


Figure 15.1 Generic security and protection setups in an operating system.

Summary

- In today's session, you all have gone through the following topics
  - Operating System operations
  - Process Management
  - Memory Management
  - Storage Management
  - I/O System
  - Protection and Security

Discussion and Interaction

Topics for Next Session

Module 1: Introduction to operating systems, System structures

Session 4

- Distributed Systems
- Special Purpose Systems
  - Multimedia Systems
  - Handheld Systems
- Computing Environments
  - Traditional Computing
  - Client-Server Computing
  - Peer-to-Peer Computing
  - Web-Based Computing


Thank you