Chapter 1 - Introduction

What Operating Systems Do :

- A computer system has four components: **the hardware**(CPU, memory and I/O)**, operating system, application programs** and **users.**
- From **user view,** we have single user computer focusing ease-of-use than resource utilization. Similarly we have **mainframe** or **minicomputer** designed to maximize the resource utilization. We also have **workstation** that are designed to compromise between individuals usability and resource utilization.
- From **System view,** we can see operating system as **resource allocator** working as manager of the conflicting requests. We can view of OS as **control program** the manages the execution of user programs to prevent errors and improper use of the computer.
- Operating System is the one program running at all times on the computer - usually called the **kernel,** along with **system programs** (associated with OS but are not part of kernel) and **application programs.** Mobile operating systems, in addition to these have **middleware** - a set of software frameworks that provides additional services to application developers.

Computer System Organisation

- Modern computer system consists of one or more CPUs and a number of **device controllers** which are incharge of a specific type of device. When system is rebooted or powered up - it run a initial program , or **bootstrap program**, stored within computer hardware (**ROM** or **EEPROM** known by the general term **firmware**). Once kernel is loaded, it start providing its services, system programs are loaded into memory becoming **system processes** or **system daemons**. In UNIX, the first program is “init”, once fully loaded system waits for some event to occur. The event is usually signaled by an interrupt from either hardware (sending signal to CPU) or software (by executing a special operation called system call). When CPU is interrupted, it stops what it is doing and immediately transfers execution to a fixed location containing **interrupt service routine.** To handle these we can have generic routine(which take us to particular ISR but are slow) or a table of pointers to interrupt routine can used to increase the speed. The **interrupt vector**, of address is indexed by unique device number, given with the interrupt request, to provide the address of the interrupt service routine for the interrupting device. Modern computer use the system stack to save the address of the interrupted instruction. It handle modification of processor state by interrupt service routine by recollecting the state from stack.
- General-purpose computers run most of their programs from rewritable memory, **RAM,** implemented in a semiconductor technology **Dynamic RAM**. ROM and

EEPROM cannot be erased hence contains program like bootstrap or smartphones have **factory-installed program**. In instruction-execution cycle with a **von-Neumann architecture,** first fetches an instruction from memory and stores it in **instruction register.** The instruction is decoded and associated operands are fetched from memory to in system registers. Since RAM is small and volatile most system have **secondary storage.** In addition to cost and speed, they differ being volatile and non-volatile nature.

- A general-purpose computer system consist of CPU and multiple device controllers that are connected through a common bus. The device controller is responsible for moving the data between the peripheral device that it controls and its local buffer storage. OSs have a **device driver** for each device controller. The device driver understands the device controller and provides the rest of the operating system with a uniform interface to the device. Device driver load the appropriate registers and device controller read them to understand the action to be performed. Once data transfer is complete the device controller inform device driver, which in turn transfer the control to OS, possibly sending the pointer to the data if the operation was to read. For large amount there is overhead in sending the data to the memory hence we use **Direct Memory Access (DMA).** Device controller transfer an entire block of data directly to memory without intervention of CPU.

Computer-System Architecture

- Most computer systems used a single processor, which may have some device-specific (special purpose) processor. All special purpose processors run a limited instruction set and do not run user processors. Once instructed they perform their instruction independently hence free up the CPU for processing the other instructions.
- **Multiprocessor systems** (also known as parallel systems or multicore systems) are becoming common because of various advantages like **increased throughput, economy of scale** (multiprocessor system having N processors cost less than N single processor systems) and **increased reliability.** Increased reliability of computer system is crucial in many application. **Graceful degradation** and **fault tolerance** allow systems to continue to provide their services even if some components suffer data failure. Multiple-processor systems in use today are of two types: **asymmetric multiprocessing** (following a boss-work relationship) and **symmetric multiprocessing** (SMP, in which each processor performs all tasks within the operating system i.e every one is peer). **Page 16 multicores systems and blade servers.**

- **Clustered system** composed of two or more system joined together. Generally used for providing high-availability service. **Asymmetric clustering** have one system in standby mode, just monitoring the server and once it fails it become the active server. In **symmetric clustering,** two or more hosts running applications and are monitoring each other. Clustering provides high-performance computing environment.

Operating-System Structure

- One of the most important aspect of operating system is multiprocessing capability. Multiprogramming increases CPU utilization by organizing jobs so that CPU always has one to execute. Main memory is too small to accomodate all jobs, hence they are kept initially on the disk in the **job pool.** If a job have to wait for some task to complete, the CPU switches to another job. **Time sharing** is logical extension of multiprogramming. Time sharing requires an **interactive** computer system and short **response time.** In time sharing, system switches between processes or users very frequently giving an impression that complete system is dedication for his use. System uses **job scheduling** to select a job from job pool and brings it in **ready queue.** It uses **CPU scheduling** to select a particular job from a ready queue for execution. A reasonable response time is achieved through **swapping,** whereby processes are swapped in and out of main memory to the disk. Concept of **virtualization** allows execution of a process that is not completely in memory.

Operating-System Operations

- Modern operating systems are **interrupt driven.** We have two different mode of operation: **user mode** and **kernel mode.** At boot time, hardware starts in kernel mode. The operating system is loaded and start user application in user mode. Whenever the operating system gains control of the computer, it is in kernel mode. The system always switches to user mode before passing control to a user program. System calls provide the means for a user program to ask the operating system to perform tasks reserved for the operating system on the user program's behalf. When a system call is executed, it is typically treated by the hardware as a software interrupt. Control passes through the interrupt vector to a service routing in the operating system, and the mode bit is set to kernel mode.

Process Management

- A program is a passive entity whereas a process is an active entity. A **single-threaded** process has one **program counter** specifying the next instruction to execute. A multithreaded process has multiple program counters,

each pointing to the next instruction to execute for a given thread. A process is a unit of work in a system. The operating system is responsible for the following activities in connection with process management:

1. Scheduling processes and threads on the CPUs.
2. Creating and deleting both user and system processes.
3. Suspending and resuming processes.
4. Providing mechanism for process synchronization.
5. Providing mechanism for process communication.

Memory Management

- The operating system is responsible for the following activities in connection with memory management:
    1. Keeping track of which parts of memory are currently being used and who is using them
    2. Deciding which processes (or parts of processes) and data to move into and out of memory
    3. Allocating and deallocating memory space are needed.

Storage Management

- File-System Management
    1. Creating and deleting files
    2. Creating and deleting directories to organize files
    3. Supporting primitives for manipulating files and directories
    4. Mapping files onto secondary storage
    5. Backing up files on stable (non-volatile) storage media.
- Mass-Storage Management
    1. Free storage management
    2. Storage allocation
    3. Disk scheduling
- Caching : When a information is used, it is copied into a faster storage system for temporary basis. When we need a particular piece of information, we first check whether it is in cache. Because caches have limited size, **cache management** is an important design problem. Main memory can be viewed as a fast cache for secondary storage. The movement of information between levels of storage hierarchy may cause data to appear in different levels of the storage system. In multitasking environment, where the CPU switches back and forth among various processes, extreme care must be taken to ensure that, if several processes wish to access the same resource, then each of these processes will obtain the most recently updated value of resource. In multiprocessor environment, each process

has its own cache, which may contain same data. Hence when one value is updated it should be reflected in the different cache immediately. This situation is called **cache coherency.**

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

Protection and Security

-

Kernel Data Structure

-

Computing Environment

-

Open Source Operating System

-