Introduction – Outline

- Goals of an operating system
- Computer system operation
- Types of operating systems
- Operating system tasks and functions

OS GOALS

What is an Operating System?

- A program that acts as an intermediary between users and the computer hardware.
- Who needs an OS?
  - all general purpose computers use an OS
  - OS makes a computer easier to use
- A better question: Who does not need OS?
  - Some specialized systems that usually do one thing (OS can be embedded in the application).
    - Microwave oven control, MP3 players, etc.
- Operating System's role:
  - User viewpoint
  - System viewpoint

User's View of the Operating System

- Make it easier to write programs
  - provide an abstraction over the hardware
- Provide more powerful instructions than the ISA
  - e.g., write(fileno, buf, len);
- Make it easy to run programs
  - loading program into memory, initializing program state, maintaining program counter, stopping the program
  - instead user types: *gcc hello.c* and then *./a.out*
- Utilities for multi-user mode operation
  - resource management, security, fairness, performance

System's View of the Operating System

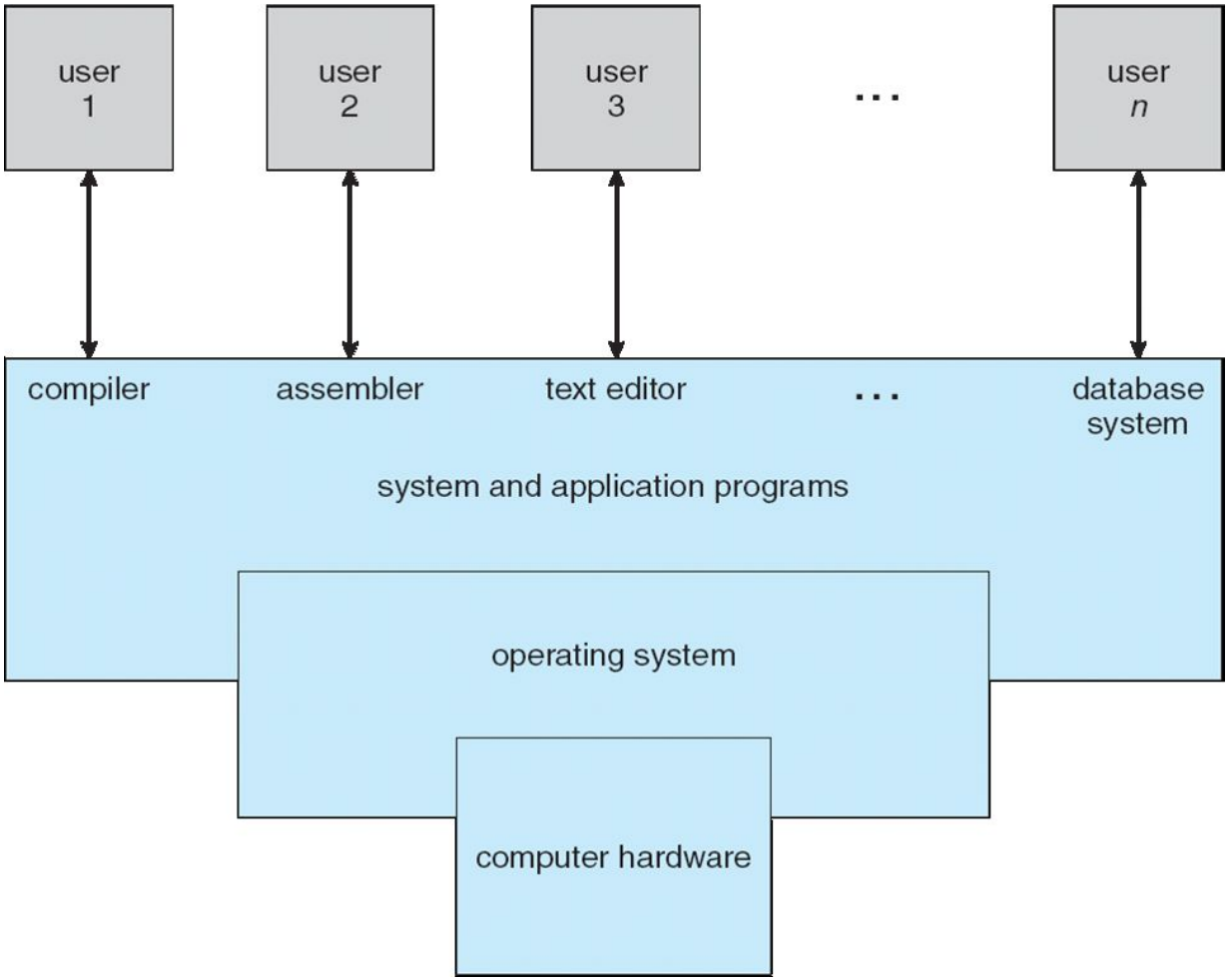
- OS as a resource allocator
  - manage CPU time, memory space, file storage space, I/O devices, network, etc.
  - fairly resolve conflicting requests for hardware resources from multiple user programs
- OS as a control program
  - control the various I/O devices and user programs

What Does an Operating System Do?

- OS is a program that acts as an intermediary between a user of a computer and the computer hardware.
  - provide an environment for easy, efficient program execution
- Operating system goals:
  - execute user programs and make solving user problems easier
  - make the computer system convenient to use
  - use the computer hardware in an efficient manner
- Definition
  - everything that a vendor ships when you order an *OS* !
  - program that is always running on the computer (*kernel)*

computer system operation

Components of a Computer System



Components of a Computer System

- Hardware
  - provides basic computing resources (CPU, memory, I/O devices)
- Operating system
  - controls and coordinates use of hardware among various applications and users
- Application programs
  - define the ways in which the system resources are used to solve the computing problems of the users (word processors, compilers, web browsers, database systems, video games)
- Users

Computer System Operation

- *Bootstrap program* for computer to start running
  - initialize CPU registers, device controllers, memory
  - locate and load the OS kernel
- OS starts executing the first program
  - waits for some event to occur (interrupt-driven)
- Occurrence of an interrupt (exceptions and traps)
  - processor checks for occurrence of interrupt
  - transfers control to the interrupt service routine, generally, through the *interrupt vector* table (IVT)
  - IVT is at a fixed address in memory (known to CPU)
  - execute the associated interrupt service routine
  - return control to the interrupted program

Operating System Operation

- OS hides the complexity and limitations of hardware (hardware interface) and creates a simpler, more powerful abstraction (OS interface).
- The bootstrap program locates the OS kernel and loads into memory.
- OS sits quietly, waits for events
  - hardware interrupts (sent to the CPU over the system bus)
  - software interrupts (software error or system call)

Operating System Types

Evolution of Operating Systems

- Batch systems
  - earliest operating systems
  - each user submits one or more *jobs*
  - collect a *batch* of jobs
  - process the batch one job at a time, one after the other
- Problems
  - Job 2 cannot start until job 1 finishes
  - I/O activity stalls the CPU; CPU under-utilized
  - No interactivity

Evolution of Operating Systems

- Multiprogramming
  - a single job cannot keep CPU and I/O devices busy at all times
  - OS is given multiple runnable jobs at a time
  - when current job has to wait (for I/O for example), OS switches to another job
  - multiple jobs kept in memory simultaneously
  - I/O and CPU computation can overlap
  - CPU-bound Vs. I/O bound jobs
  - OS goal: maximize system throughput

Evolution of Operating Systems

- Time sharing *(multitasking)*
  - logical extension of multiprogramming
  - CPU switches jobs so frequently that users can interact with each job while it is running
  - very fast response time
  - each user has at least one program executing in memory
  - OS scheduler decides which job will run if several jobs are ready to run at the same time
  - If processes don't fit in memory, *swapping* moves them in and out to run
  - *virtual memory* allows execution of processes not completely in memory
  - OS goal: optimize user response time

Evolution of Operating Systems

- Distributed Operating Systems
  - for physically separate, heterogeneous, networked computers
  - Hardware: computers with networks
  - OS goal: ease of resource sharing among machines
- Virtualization
  - OS itself runs under the control of an *hypervisor*
  - VM have own memory management, file system, etc.
  - VM is key feature of some operating systems
    - Windows server 2008, HP Integrity VM
  - hypervisor no longer optional
    - POWER5 and POWER6 from IBM

Operating System Tasks

Operating System Tasks

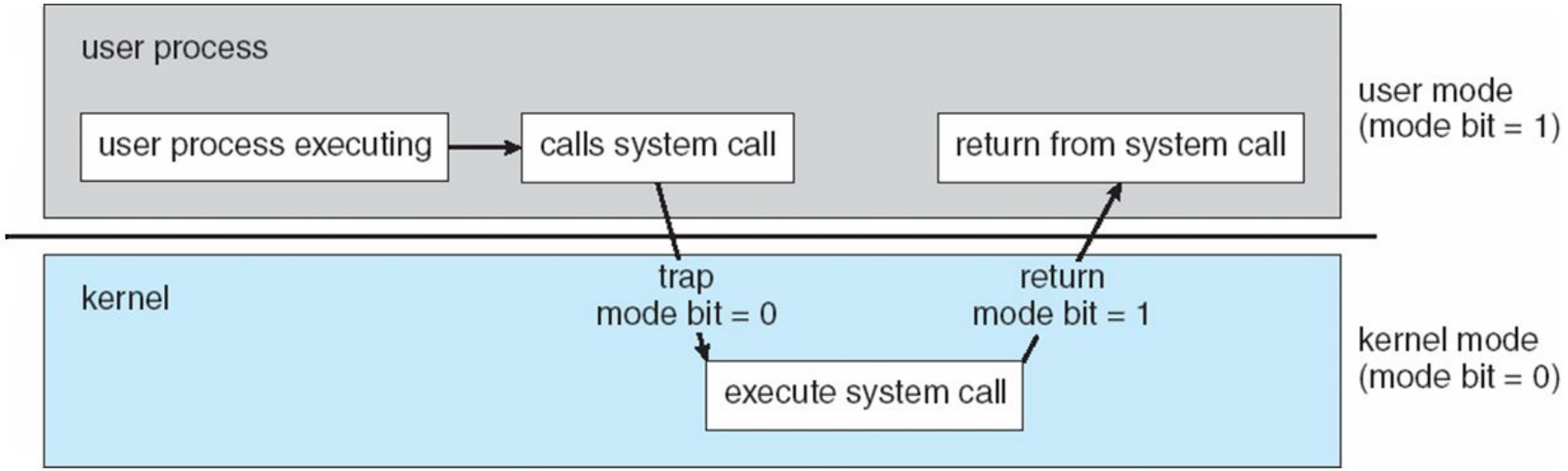
- Process co-ordination and security
- Process management
- Memory management
- Storage management
- Other issues in protection and security

Process Co-ordination & Security

- Applications should not crash into one-another
  - *address space:* all memory addresses that an application can touch
  - address space of one process is separated from address space of another process
- Applications should not crash the OS
  - *dual-mode* operation (*user* mode and *kernel* mode)
  - distinguish when system is running in user or system mode
  - *privileged* instructions only operate in kernel mode
  - system calls / returns change mode

Transition from User to Kernel Mode

- Timer to prevent infinite loop / process hogging resources
  - OS sets *timer* interrupt after specific period
  - hardware decrements counter
  - when counter zero generate an interrupt
  - set up before scheduling process to regain control or terminate program that exceeds allotted time



Process Management

- Process
  - is a program in execution
  - is a unit of work within the system
  - program is a *passive entity*, process an *active entity*
  - needs resources to accomplish its task
    - termination requires reclaim of any reusable resources
- Single-threaded process has one program counter specifying location of next instruction to execute
  - execute instructions sequentially, until completion
- Multi-threaded process has one program counter per thread.

Process Management Activities

- The operating system is responsible for the following activities in connection with process management
  - process scheduling
  - suspending and resuming processes
  - providing mechanisms for process synchronization
  - providing mechanisms for process communication
  - providing mechanisms for deadlock handling

Memory Management

- All data in memory before and after processing
- All instructions in memory in order to execute
- Memory management determines what is in memory when
  - optimizing CPU utilization and computer response to users
- Memory management activities
  - keeping track of which parts of memory are currently being used and by whom
  - deciding which processes (or parts thereof) and data to move into and out of memory
  - allocating and deallocating memory space as needed

Storage Management

- OS provides uniform, logical view of info. storage
  - abstracts physical properties to logical storage unit – file
  - medium controlled by device (i.e., disk drive, tape drive)
    - varying properties include access speed, capacity, data-transfer rate, access method (sequential or random)
- File-System management
  - Files usually organized into directories
  - Access control is provided to determine who can access what
  - OS activities include
    - Creating and deleting files and directories
    - Primitives to manipulate files and dirs
    - Mapping files onto secondary storage
    - Backup files onto stable (non-volatile) storage media

Storage Hierarchy

- Performance of various levels of storage depends on
  - distance from the CPU, size, and process technology used
- Movement between levels of storage hierarchy can be explicit or implicit

Level	1	2	3	4	5
Name	registers	cache	main memory	solid state disk	magnetic disk
Typical size	< 1 KB	< 16MB	< 64GB	< 1 TB	< 10 TB
Implementation technology	custom memory with multiple ports CMOS	on-chip or off-chip CMOS SRAM	CMOS SRAM	flash memory	magnetic disk
Access time (ns)	0.25 - 0.5	0.5 - 25	80 - 250	25,000 - 50,000	5,000,000
Bandwidth (MB/sec)	20,000 - 100,000	5,000 - 10,000	1,000 - 5,000	500	20 - 150
Managed by	compiler	hardware	operating system	operating system	operating system
Backed by	cache	main memory	disk	disk	disk or tape

I/O Subsystem

- One purpose of OS is to hide peculiarities of hardware devices from the user
- I/O subsystem responsible for
  - Memory management of I/O including buffering (storing data temporarily while it is being transferred), caching (storing parts of data in faster storage for performance), spooling (the overlapping of output of one job with input of other jobs)
  - General device-driver interface
  - Drivers for specific hardware devices

Protection and Security

- Protection – mechanism for controlling access of processes or users to resources defined by the OS
- Security – defense of the system against internal and external attacks
  - denial-of-service
  - worms
  - viruses
  - identity theft
  - theft of service

Protection and Security (2)

- Systems generally first distinguish among users, to determine who can do what
  - user identities (user IDs, security IDs) include name and associated number, one per user
  - user ID then associated with all files, processes of that user to determine access control
  - group identifier (group ID) allows set of users to be defined and controls managed, then also associated with each process, file
  - privilege escalation allows user to change to effective ID with more rights

summary

Summary – Operating System

- OS is the software layer between the hardware and user programs.
- OS is the ultimate API.
- OS is the first program that runs when the computer boots up.
- OS is the program that is always running.
- OS is the resource manager.
- OS is the creator of the virtual machine.

Summary – Operating System (2)

Reality	***Abstraction***
A single CPU	Multiple CPUs
Limited RAM capacity	Infinite capacity
Mechanical disk	File system
Insecure and unreliable networks	Reliable and secure
Many physical machines	A single machine