Introduction – Outline

- History of operating systems
- What does the operating system do?
- Where does the operating system fit in a computing system?
- What are the general operating system functions?

History of Operating Systems: Phase I

- Hardware Expensive, Humans Cheap
  - Hardware: mainframes
  - OS: human operators
    - Handle one job (a unit of processing) at a time
    - Computer time wasted while operators walk around the machine room

OS Design Goal

- Efficient use of the hardware
  - Batch system: collects a batch of jobs before processing them and printing out results
    - Job collection, job processing, and printing out results can occur concurrently
  - Multiprogramming: multiple programs can run concurrently
    - Example: I/O-bound jobs and CPU-bound jobs

History of Operating Systems: Phase II

- Hardware Cheap, Humans Expensive
  - Hardware: terminals
  - OS design goal: more efficient use of human resources
    - Timesharing systems: each user can afford to own terminals to interact with machines
History of Operating Systems: Phase III

- Hardware very cheap, humans very expensive
  - Hardware: personal computers
  - OS design goal: allowing a user to perform many tasks at the same time
    - Multitasking: the ability to run multiple programs on the same machine at the same time
    - Multiprocessing: the ability to use multiple processors on the same machine

Who needs OS?

- OS makes a computer easier to use
  - All general purpose computers need OS.
- A better question: Who does not need OS?
  - Some very specialized systems that usually do one thing (OS can be embedded in the application).
    - Microwave oven control
    - MP3 players, etc.

History of Operating Systems: Phase IV

- Distributed Systems
  - physically separate, heterogeneous, networked computers
  - Hardware: computers with networks
  - OS design 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

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)

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 the computer to start running
  - initialize CPU registers, device controllers, memory contents
  - 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)
  - transfer control to the interrupt service routine, generally, through the interrupt vector
  - execute the associated interrupt service routine
  - return control to the interrupted program

Operating System Structure

- Hide the complexity and limitations of hardware (hardware interface) and create a simpler, more powerful abstraction (OS interface).
  - **Multiprogramming**
    - needed for efficiency
    - single user cannot keep CPU and I/O devices busy at all times
    - multiprogramming organizes jobs (code and data) so CPU always has one to execute
    - subset of total jobs in system is kept in memory
    - one job selected and run via job scheduling
    - when it has to wait (for I/O for example), OS switches to another job

Operating System Structure (cont...)

- **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
    - **process**
  - If several jobs ready to run at the same time
    - **CPU scheduling**
  - 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

Operating System Operations

- 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 and from the OS
- 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
  - set interrupt after specific period
  - operating system 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

<table>
<thead>
<tr>
<th>Level</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>registers</td>
<td>cache</td>
<td>main memory</td>
<td>disk storage</td>
</tr>
<tr>
<td>Typical size</td>
<td>&lt; 1 KB</td>
<td>&gt; 16 MB</td>
<td>&gt; 16 GB</td>
<td>&gt; 100 GB</td>
</tr>
<tr>
<td>Implementation technology</td>
<td>custom memory with multiple ports, CMOS</td>
<td>on-chip or off-chip CMOS SRAM</td>
<td>CMOS DRAM</td>
<td>magnetic disk</td>
</tr>
<tr>
<td>Access time (ns)</td>
<td>0.25 – 0.5</td>
<td>0.5 – 25</td>
<td>80 – 250</td>
<td>5,000,000</td>
</tr>
<tr>
<td>Bandwidth (MB/sec)</td>
<td>20,000 – 100,000</td>
<td>5000 – 10,000</td>
<td>1000 – 5000</td>
<td>20 – 150</td>
</tr>
<tr>
<td>Managed by</td>
<td>compiler</td>
<td>hardware</td>
<td>operating system</td>
<td>operating system</td>
</tr>
<tr>
<td>Backed by</td>
<td>cache</td>
<td>main memory</td>
<td>disk</td>
<td>CD or tape</td>
</tr>
</tbody>
</table>

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 – 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)

<table>
<thead>
<tr>
<th>Reality</th>
<th>Abstraction</th>
</tr>
</thead>
<tbody>
<tr>
<td>A single CPU</td>
<td>Multiple CPUs</td>
</tr>
<tr>
<td>Limited RAM capacity</td>
<td>Infinite capacity</td>
</tr>
<tr>
<td>Mechanical disk</td>
<td>File system</td>
</tr>
<tr>
<td>Insecure and unreliable</td>
<td>Reliable and secure</td>
</tr>
<tr>
<td>networks</td>
<td></td>
</tr>
<tr>
<td>Many physical machines</td>
<td>A single machine</td>
</tr>
</tbody>
</table>