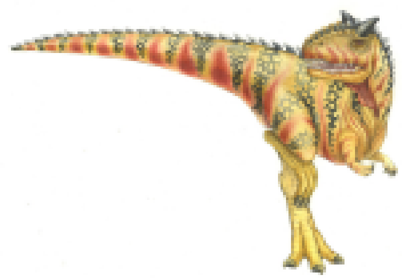
Chapter 1: Introduction



Outline

- What Operating Systems Do
- Computer-System Organization
- Computer-System Architecture

- Operating-System Operations
- Resource Management
- Security and Protection
- Virtualization
- Distributed Systems
- Kernel Data Structures
- Computing Environments

- Free/Libre and Open-Source Operating Systems

Objectives

- Describe the general organization of a computer systemandtherole of interrupts
- Describe the components in a modern, multiprocessor

computersystem

- Illustrate the transition from user mode to kernel mode
- Discuss how operating systems are used in various computingenvironments
- Provide examples of free and open-source operating systems

What Does the TermOperatingSystemMean?

- An operating system is “fill in the blanks”
- What about:
  - Program

- Hardware
- Compiler
- Etc.

What is an OperatingSystem?

- A program that acts as an intermediary between a user of acomputer and the computer hardware

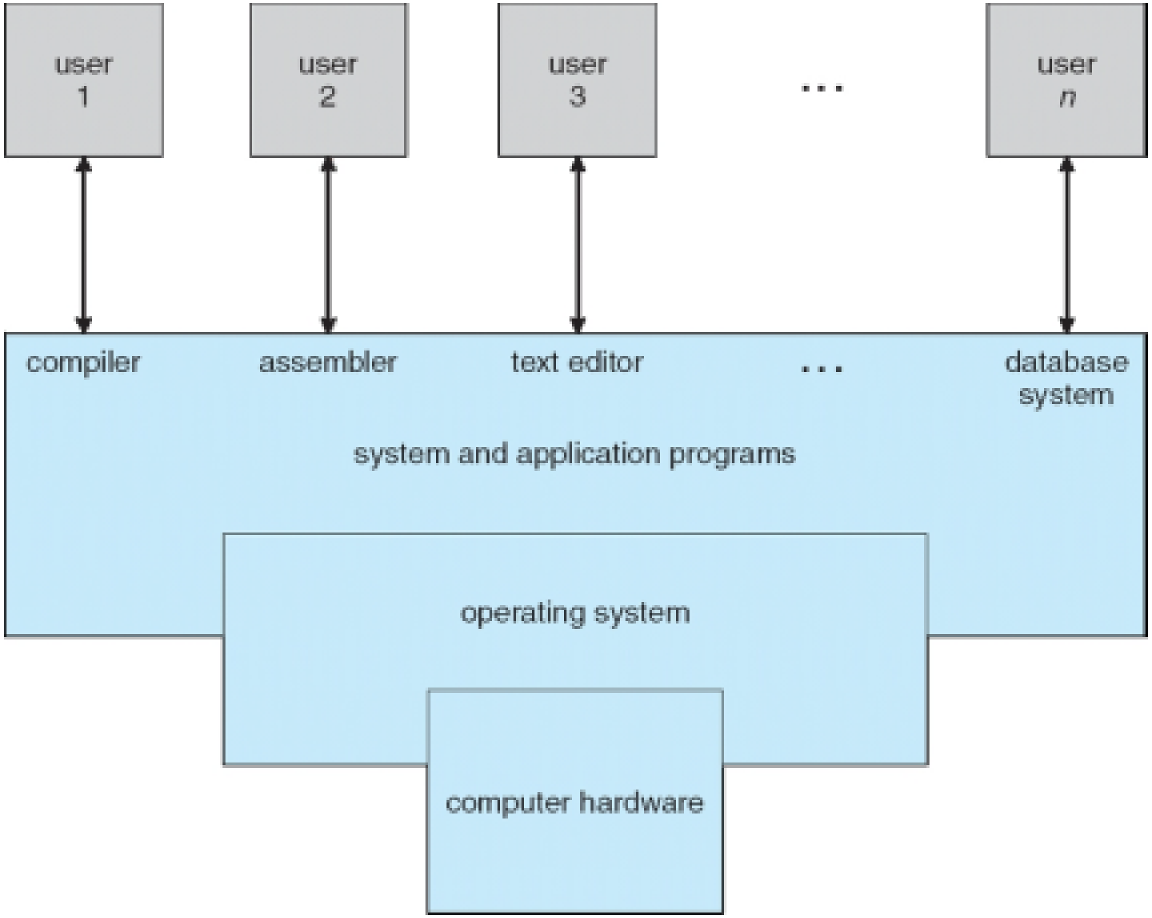
- Operating system goals:
  - Execute user programs and make solving user problemseasier
  - Make the computer system convenient to use • Use the computer hardware in an efficient manner

Computer SystemStructure

- Computer system can be divided into four components: • Hardware – provides basic computing resources 4 CPU, memory, I/O devices

- Operating system
  - 4 Controls and coordinates use of hardware amongvariousapplications and users
- Application programs – define the ways in which thesystemresources are used to solve the computing problems of theusers
  - 4 Word processors, compilers, web browsers, databasesystems, video games
- Users
  - 4 People, machines, other computers

Abstract View of ComponentsofComputer



Abstract View of Componentsof Computer

What OperatingSystemsDo

- Depends on the point of view
- User''s View (type of user)

 Standalone s/m - OS is designed for **ease of use** and **goodperformance**

 different terminals connected to **mainframe** or **minicomputer -** OSisdesigned to **maximize resource utilization** users of workstation - **ease of use** and **resource utilization** users of handheld devices - **ease of use** and **good performance**peramount of battery

- optimized for usability and battery life

 Embedded systems - designed for less user intearactionsand**easeofuse**

- little or no user interface
- designed to run primarily without user intervention

What OperatingSystemsDo(Cont.)

- System View:
  - OS is a **resource allocator**
  - Manages all resources
  - Decides between conflicting requests for efficient andfair resourceuse
  - OS is a **control program** • Controls execution of programs to prevent errors andimproperuseofthe computer

Operating SystemDefinition

- No universally accepted definition
- “Everything a vendor ships when you order an operatingsystem”isagood approximation
  - But varies wildly
- “The one program running at all times on the computer ” isthe**kernel,** which is part of the operating system
- Everything else is either
  - A **system program** (ships with the operating system, but not partofthe kernel) , or
  - An **application program**, all programs not associatedwiththe operating system
- Today’s OSes for general purpose and mobile computingalsoinclude**middleware** – a set of software frameworks that provideadditionservicesto application developers such as databases, multimedia, graphicsetc.

Computer SystemOrganization

Computer-system operation

- One or more CPUs, device controllers connect throughcommon**bus** providing access to shared memory
- Concurrent execution of CPUs and devices competingfor memorycycles

Computer-SystemOperation

- I/O devices and the CPU can execute concurrently
- Each device controller is in charge of a particular devicetype
- Each device controller has a local buffer

- Each device controller type has an operating system**devicedriver**to manage it
- CPU moves data from/to main memory to/fromlocal buffers
- I/O is from the device to local buffer of controller ▪ Device controller informs CPU that it has finished its operationbycausing an **interrupt**

Computer Startup

- **Bootstrap program** is loaded at power-up or reboot ▪ Typically stored in ROM or EPROM, generally known as **firmware**
  - Initializes all aspects of system: CPU registers, memory, device

controllesrs and other initial setups

- Loads operating system kernel and OS starts with the execution of ‘init’ process
- **Bootstrap program**
    - Initializes regs., memory, I/O devices
    - locates and loads kernel into memory
    - starts with ‘init’ process
    - wiats for interrupt from user

Interrupt Handling

- Occurence of an event is signalled by **interrupt**
- **Hardaware interrupt -** triggered by an event external totheprocessor

- **Software interrupt -** triggered by executing special intruction/operation (API or system calls) ▪ Interrupt transfers control to the interrupt service routinegenerally, through the **interrupt vector**, which contains theaddresses of all the service routines
- Interrupt architecture must save the address of the interruptedinstruction
- A **trap** or **exception** is a software-generated interrupt causedeither by an error or a user request
- An operating system is **interrupt driven**

Interrupt DrivenProgramExecution

Interrupt Timeline

Interrupt Handling

- The operating system preserves the state of the CPUby storing the registers and the program counter
- Determines which type of interrupt has occurred:
- Separate segments of code determine what action shouldbe taken for each type of interrupt

Interrupt-drivel/OCycle

Storage Structure

- Main memory – only large storage media that the CPUcanaccess directly
  - Typically, **volatile**
    - Typically, **random-access memory** in the formof **Dynamic Random-access Memory** (**DRAM**)
- Secondary storage – extension of main **memory** that provides large **nonvolatile** storage capacity

Storage Structure(Cont.)

- **Hard Disk Drives** (**HDD**) – rigid metal or glass platters covered with magnetic recording material
  - Disk surface is logically divided into **tracks**, which aresubdivided into **sectors**
  - The **disk controller** determines the logical interaction between the device and the computer

- **Non-volatile memory** (**NVM**) devices– faster than harddisks, nonvolatile
  - Various technologies
  - Becoming more popular as capacity and performanceincreases, price drops

Storage Definitions andNotationReview

The basic unit of computer storage is the **bit** . A bit can contain one of twovalues, 0 and 1. All other storage in a computer is based on collections of bits. Given enough bits, it is amazing how many things a computer can represent:numbers, letters, images, movies, sounds, documents, and programs, tonamea few. A **byte** is 8 bits, and on most computers, it is the smallest convenient chunk of storage. For example, most computers don’t have an instructiontomove a bit but do have one to move a byte. A less common termis **word**, which is a given computer architecture’s native unit of data. A word is madeup of one or more bytes. For example, a computer that has 64-bit registersand64-bit memory addressing typically has 64-bit (8-byte) words. A computer executes many operations in its native word size rather than a byte at atime.

Computer storage, along with most computer throughput, is generallymeasured and manipulated in bytes and collections of bytes. A **kilobyte**, or KB , is 1,024 bytes; a **megabyte** , or **MB** , is $1{,}024^2$ bytes; a **gigabyte**, or GB,is$1{,}024^3$ bytes; a **terabyte** , or **TB** , is $1{,}024^4$ bytes; and a **petabyte** , or **PB**, is$1{,}024^5$bytes. Computer manufacturers often round off these numbers and saythat a megabyte is 1 million bytes and a gigabyte is 1 billion bytes. Networkingmeasurements are an exception to this general rule; they are given inbits(because networks move data a bit at a time).

Storage Hierarchy

- Storage systems organized in hierarchy

- Speed
- Cost
- Volatility

- **Caching** – copying information into faster storage system; mainmemorycan be viewed as a cache for secondary storage
- **Device Driver** for each device controller to manage I/O• Provides uniform interface between controller and kernel

Storage-DeviceHierarchy

How a Modern ComputerWorks

A von Neumann architecture

I/OStructure

- Two methods for handling I/O
  - After I/O starts, control returns to user programonly upon I/O completion
  - After I/O starts, control returns to user programwithout waiting for I/O completion

I/O Structure(Cont.)

- After I/O starts, control returns to user programonly uponI/Ocompletion
  - Wait instruction idles the CPU until the next interrupt • Wait loop (contention for memory access)
  - At most one I/O request is outstanding at a time, nosimultaneous I/O processing
- After I/O starts, control returns to user programwithout waitingforI/O completion
  - **System call** – request to the OS to allow user to wait for I/Ocompletion
  - **Device-status table** contains entry for each I/Odeviceindicating its type, address, and state
  - OS indexes into I/O device table to determine devicestatusand to modify table entry to include interrupt

Direct Memory AccessStructure

- Used for high-speed I/O devices able to transmit informationat close to memory speeds
- Device controller transfers blocks of data frombuffer storagedirectly to main memory without CPU intervention
- Only one interrupt is generated per block, rather than theoneinterrupt per byte

Computer-SystemArchitecture

▪ Most systems use a single general-purpose processor • Most systems have special-purpose processors as well ▪ **Multiprocessors** systems growing in use and importance• Also known as **parallel systems**, **tightly-coupledsystems**• Advantages include:

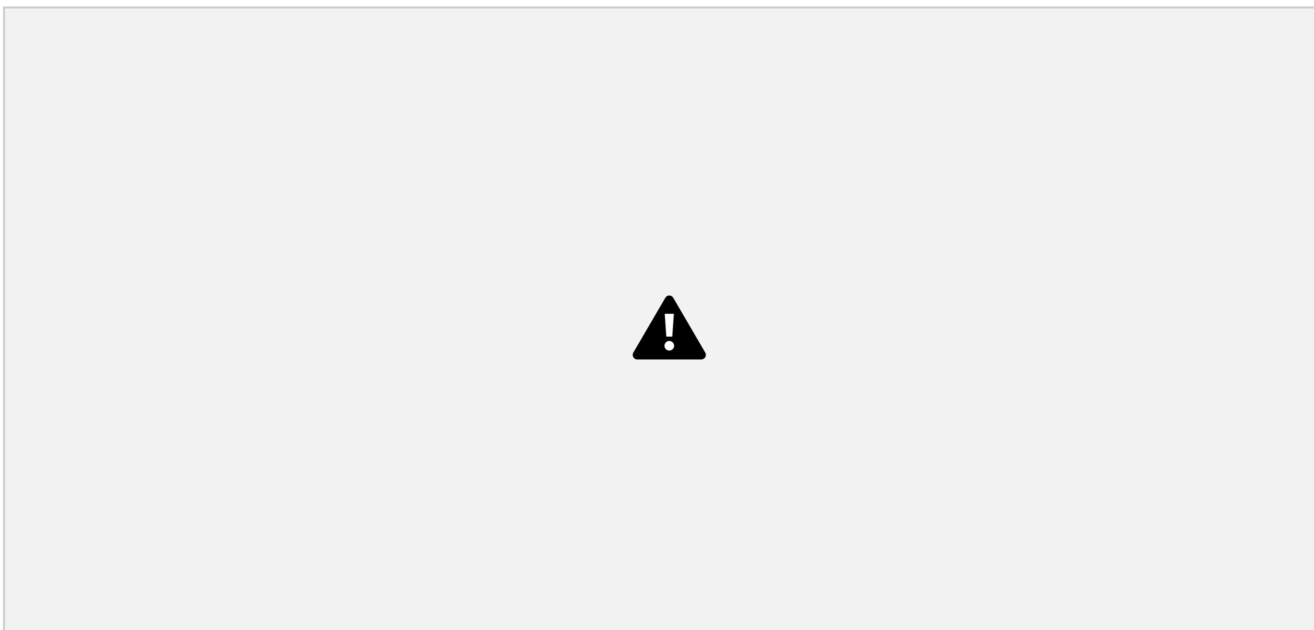
1. **Increased throughput :** No. programs executedper unit timeismore; increasing number of processors do not alwaysguaranteeincreased performance due to the overhead of comlexityandcost
2. **Economy of scale** – costs less than equivalent no. of manysingleprocessors due to sharing of peripherals, mass storageandpowersupplies
3. **Increased reliability** – if one processor fails, thes/misnothalted,itonly slows down; the job of the failed processor istakenupbyotherprocessor

- graceful degradation or fault tolerant

Symmetric MultiprocessingArchitecture

Two types of multiprocessor systems:

1. **Asymmetric Multiprocessing (Master/Slave)**–eachprocessor is assigned a specific task by master processor.Itschedules and allocates work to slave processors
2. **Symmetric Multiprocessing (SMP)**–eachprocessorperforms all tasks; all processors are consideredpeers;Windows, MacOS, Liux support SMP



A Dual-CoreDesign

- Multiple processing units (cores) fabricated onasinglechip - **multicore processors**
  - The term processor is then used for the completechip
- The communication b/w processors within a chipismorefaster than communication between two singleprocessors

ClusteredSystems

- Like multiprocessor systems, but multiple systems workingtogetherandsharing software resources
  - Usually sharing storage via a **storage-area network(SAN)** -whereresources can be shared with dozens of systems inacluster, thatareseparated by miles.
  - Provides a **high-availability** of resources and serviceswhichsurvivefailures;the service continues even if one or more systemsintheclusterfail; duplication of s/w resources

    4 **Asymmetric clustering** has one machine in hot-standbymode

    4 **Symmetric clustering** has multiple nodes runningapplications,monitoring each other
  - Some clusters are for **high-performance computing(HPC)**4

    Applications must be written to use **parallelization**
  - Parallel clusters allow multipe hosts to access

samedataonthesharedstorage

ClusteredSystems

Operating SystemStructure

- **Multiprogramming** (**Batch system**) needed for efficiency • Single user cannot keep CPU and I/O devices busy at all times• Multiprogramming organizes jobs (code and data) so CPUalwayshasoneto execute

- A 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 andexecutesanother job
- Multiprogrammed s/ms provide envoronment in which various s/mresourcesare utilized effectively, but they do not provide user interactionwiththe computer system

Memory Layout for

MultiprogrammedSystem

Operating SystemStructure

- **Timesharing** (**multitasking**) is logical extension in which a singleCPUexecutes multiple jobs by switching among them

switches jobs so frequently that users can interact with each job whileit isrunning,creating **interactive** computing

- **Response time** should be < 1 second
- Each user has at least one program executing in memory **process**
- If several jobs ready to run at the same time **CPUscheduling**
- If processes don’t fit in memory, **swapping** moves theminandout torun
- **Virtual memory** allows execution of processes not completelyinmemory

Operating-SystemOperations

- **Interrupt driven** (hardware and software)
  - Hardware interrupt by one of the devices
  - Software interrupt (**exception** or **trap):**
    - Software error (e.g., division by zero)
    - Request for operating system service
    - Other process problems include infinite loop, processes modifying each other or the operating system

Dual-modeOperation

- **Dual-mode** operation allows OS to protect itself and other systemcomponents
    - **User mode** and **kernel mode**
- **Mode bit** provided by hardware
    - Provides ability to distinguish when systemis runninguser codeorkernel code.
    - When a user is running mode bit is “user” (set to1) • When kernel code is executing mode bit is “kernel” (set to0)▪ Some instructions designated as **privileged**, only executableinkernelmode.
- Eg: changing the mode bit; changing the priority level etc.

Dual-mode Operation(Cont.)

- How do we guarantee that user does not explicitly set themode bit to "kernel"?
- When the system starts executing it is in kernel mode ▪ When control is given to a user program the mode-bit changesto "user mode".
- When a user issues a system call it results in an interrupt, which trap to the operating system. At that time, the mode–bit is set to "kernel mode".

Transition fromUser toKernelMode

Timer

- Operating system uses timer to control the CPU. Auser programcannot hold CPU for a long time, this is prevented with thehelpof timer.
- Timer to prevent infinite loop (or process hogging resources) • Timer is set to

  interrupt the computer after some timeperiod□ Fixed timer - After a fixed time, the process under executionisinterrupted.`

  □ Variable timer - Interrupt occurs after varying interval. • Fixed clock and a counter that is decremented every timetheclockticks

  • Operating system set the counter (privileged instruction) • When counter zero generate an interrupt

  • Set up before scheduling process to regain control or terminateprogram that exceeds allotted time

Process Management

▪ A process is a program in execution. It is a unit of work withinthesystem. Program is a ***passive entity;*** process is an ***activeentity***. ▪ Process needs resources to accomplish its task • CPU, memory, I/O, files

- Initialization data

▪ Process termination requires reclaim of any reusable resources

▪ Single-threaded process has one **programcounter** specifyinglocationof next instruction to execute

- Process executes instructions sequentially, one at atime, until completion

▪ Multi-threaded process has one program counter per thread▪ Typically, system has many processes, some user, someoperatingsystem running concurrently on one or more CPUs • Concurrency by multiplexing the CPUs among the processes/ threads

Process Management Activities

The operating system is responsible for the following activitiesinconnection with process management:

- Creating and deleting both user and systemprocesses
- Suspending and resuming processes
- Providing mechanisms for process synchronization
- Providing mechanisms for process communication
- Providing mechanisms for deadlock handling

Memory Management

- To execute a program all (or part) of the instructions must beinmemory
- All (or part) of the data that is needed by the programmust beinmemory
- To improve both the utilization of the CPU several programsarekeptinmemory, creating a need for memory management.
- Memory management determines what is in memory andwhen
  - Optimizing CPU utilization and computer responsetousers
- Memory management activities
  - Keeping track of which parts of memory are currentlybeingusedand by whom
  - Deciding which processes (or parts thereof) and datatomoveintoand out of memory
  - Allocating and de-allocating memory space as needed

File-systemManagement

- OS provides uniform, logical view of information storage• Abstracts physical properties to logical storage unit - **file**• Different types of physical media - Magnetic disk, optical disk, andmagnetic tape`
    - Each medium is controlled by device (i.e., disk drive, tapedrive)
        - 4 Varying properties include access speed, capacity, data-transferrate, access method (sequential or random)

- File-System management
    - Files usually organized into directories • Multiple users - Access control on most systems todeterminewhocanaccess what
    - OS activities include
        - 4 Creating and deleting files and directories 4 Supporting primitives to manipulate files and directories4 Mapping files onto secondary storage
        - 4 Backup files onto stable (non-volatile) storage media

Mass-StorageManagement

- Main memory is small and volatile – secondary storagebackupmain maemory
- Usually, disks used to store data that does not fit in mainmemory or data that must be kept for a "long"period of time
- Proper management is of central importance
- Entire speed of computer operation hinges on disk subsystemand its algorithms
- OS activities
  - Mounting and unmounting
  - Free-space management
  - Storage allocation
  - Disk scheduling
  - Partitioning
  - Protection

Caching

- Important principle, performed at many levels in a computer (in hardware, operating system, software)
- Information in use copied from slower to faster storagetemporarily
- Faster storage (cache) checked first to determine if information is there
  - If it is, information used directly from the cache (fast) • If not, data copied

to cache and used there ▪ Cache smaller than storage being cached • Cache management important design problem

  - Cache size and replacement policy

Characteristics of VariousTypesofStorage

Movement between levels of storage hierarchy can be explicit or implicit

Migration of data "A" fromDisktoRegister

- Multitasking environments must be careful to use most recent value,no matter where it is stored in the storage hierarchy

- Multiprocessor environment must provide **cache coherency**inhardware such that all CPUs have the most recent valueintheir cache
- Distributed environment situation even more complex
  - Several copies of a datum can exist
  - Various solutions covered in Chapter 19

I/OSubsystem

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

ProtectionandSecurity

- **Protection** – mechanism for controlling access of processesor users to resources defined by the OS
- **Security** – defense of the system against internal and external attacks
    - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service

Protection

- Systems generally first distinguish among users, to determinewhocan do what
  - User identities (**user IDs**, security IDs) include nameandassociated number, one per user

- User ID then associated with all files, processes of that user todetermine access control
- Group identifier (**group ID**) allows set of users to bedefinedandcontrols managed, then also associated with each process, file
- **Privilege escalation** allows user to change to effectiveIDwithmore rights

Computing Environments- Traditional

- Stand-alone general purpose machines
- But blurred as most systems interconnect with others (i.e., the Internet)
- **Portals** provide web access to internal systems
- **Network computers** (**thin**

clients) are like Web terminals ▪ Mobile computers interconnect via **wireless networks** ▪ Networking becoming ubiquitous – even home systems use **firewalls** to protect home computers from Internet attacks

Computing Environments- Mobile

- Handheld smartphones, tablets, etc
- What is the functional difference between themand a “traditional” laptop?
- Extra feature – more OS features (GPS, gyroscope) ▪ Allows new types of apps like ***augmented reality*** ▪ Use IEEE 802.11 wireless, or cellular data

networks for connectivity

- Leaders are **Apple iOS** and **Google Android**

Computing Environments–Distributed

Distributed computiing

- Collection of separate, possibly heterogeneous, systems networked together
  - **Network** is a communications path, **TCP/IP**most common
    - **Local Area Network** (**LAN**)

- Wide Area Network (WAN)
- Metropolitan Area Network (MAN)
- Personal Area Network (PAN)

- **Network Operating System** provides features betweensystems across network
  - Communication scheme allows systems to exchangemessages
  - Illusion of a single system

Computing Environments–Client-Server

- Client-Server Computing
  - Dumb terminals supplanted by smart PCs
  - Many systems now **servers**, responding to requestsgeneratedby **clients**
    - **Compute-server system** provides an interfacetoclient torequest

services (i.e., database)

4 **File-server system** provides interface for clients tostoreand retrieve files

Computing Environments- Peer-to-Peer

- Another model of distributed system
- P2P does not distinguish clients and servers
  - Instead all nodes are considered peers
  - May each act as client, server or both

- Node must join P2P network
  - 4 Registers its service with central lookup service on network, or
  - 4 Broadcast request for service ~~and~~ respond to requests for service via ***discovery protocol***
- Examples include Napster and Gnutella, **Voice over IP** (**VoIP**) such as Skype

Computing Environments – Real-Time EmbeddedSystems

- Real-time embedded systems most prevalent formof computers
  - Vary considerable, special purpose, limited purposeOS, **real-time OS**
  - Use expanding

- Many other special computing environments as well
  - Some have OSes, some perform tasks without an OS
- Real-time OS has well-defined fixed time constraints
  - Processing ***must*** be done within constraint
  - Correct operation only if constraints met

Computing Environments- Virtualization

- Allows operating systems to run applications within other OSes
  - Vast and growing industry
- **Emulation** - creates an environment that imitates the propertiesofonesystem onto another, mimics the properties of one processor toruninanother platform

efficiently

- hardware component is replaced by software based construct ▪ used when source CPU type different fromtarget type

  - run programs designed for a completely different architectureon an x86 PC

  - running an old game designed for obsolete platformsontodaysmodern systems (i.e. PowerPC to Intel x86)

  - Generally slowest method
  - When computer language not compiled to native code–**Interpretation**

Computing Environments- Virtualization

- **Virtualization** – OS natively compiled for CPU, running**guest** OSesalsonatively compiled
  - Consider VMware running WinXP guests, each runningapplications,all on native WinXP **host** OS

- **VMM** (virtual machine Manager) provides virtualizationservices▪ Use cases involve laptops and desktops running multipleOSesforexploration or compatibility
  - Apple laptop running Mac OS X host, Windows as aguest • Developing apps for multiple OSes without having multiplesystems• Executing and managing compute environments withindatacenters▪ VMM can run natively, in which case they are also the host

Computing Environments- Virtualization

Computing Environments- Virtualization

- Emulation requires software bridge ▪ With virtualization harware can be directly accessedwhereasinEmulator the GOS does not run on the physical hardware▪ Emulator requires interpreter to translate the sourcecodetohost’sreadable format, to further process it.
- Emulators are slower than VMs
- Emulators do not rely on CPU while VMs makeuseof CPU▪ VM solution is costlier and more complex than Emulationtechnoque
- VM provides more throughput, minimal overheadwithabetterbackup and recovery solution

Computing Environments – CloudComputing

▪ Delivers computing, storage, even apps as a service across a network▪ Logical extension of virtualization because it uses virtualization as thebasefor it functionality.

• Amazon **EC2** has thousands of servers, millions of virtual machines, petabytes of storage available across the Internet, pay basedonusage▪ Many types

• **Public cloud** – available via Internet to anyone willing to pay • **Private cloud** – run by a company for the company's own use• **Hybrid cloud** – includes both public and private cloud components• Software as a Service (**SaaS**) – one or more applications availableviathe Internet (i.e., word processor)

• Platform as a Service (**PaaS**) – software stack ready for applicationusevia the Internet (i.e., a database server)

• Infrastructure as a Service (**IaaS**) – servers or storage availableover Internet (i.e., storage available for backup use)

Computing Environments – CloudComputing

- Cloud computing environments composed of traditional OSes, plus VMMs, plus cloud management tools
  - Internet connectivity requires security like firewalls
  - Load balancers spread traffic across multiple applications

Open-Source OperatingSystems

- Operating systems made available in source-code format rather than just binary **closed-source**
- Counter to the **copy protection** and **Digital Rights Management (DRM)** movement
- Started by **Free Software Foundation (FSF)**, which has "copyleft" **GNU Public License (GPL)**
- Examples include **GNU/Linux** and **BSD UNIX** (includingcoreof **Mac OS X**), and many more
  - Can use VMM like VMware Player (Free on Windows), Virtualbox(open source and free on many platforms - http://www.virtualbox.com)
  - Use to run guest operating systems for exploration

The Study of OperatingSystems

There has never been a more interesting time to study operating systems, and it has never beeneasier. The open-source movement has overtaken operating systems, causing many of themtobemade available in both source and binary (executable) format. The list of operating systems available in both formats includes Linux, BUSD UNIX, Solaris, and part of macOS. The availability of source code allows us to study operating systems fromthe inside out. Questions that we could once answer only by looking at documentation or the behavior of anoperating system we can now answer by examining the code itself.

Operating systems that are no longer commercially viable have been open-sourced as well, enablingus to study how systems operated in a time of fewer CPU, memory, and storage resources. An extensive but incomplete list of open-source operating-system projects is available from https://curlie.org/Computers/Software/Operating_Systems/Open_Source/

In addition, the rise of virtualization as a mainstream (and frequently free) computer functionmakes it possible to run many operating systems on top of one core system. For example, VMware(http://www.vmware.com) providesa free "player" for Windows on which hundreds of free"virtual appliances" can run. Virtualbox (http://www.virtualbox.com) provides a free, open-sourcevirtual machine manager on many operating systems. Using such tools, students can tryout hundreds of operating systems without dedicated hardware.

The advent of open-source operating systems has also made it easier to make the movefromstudent to operating-system developer. With some knowledge, some effort, and an Internet connection, a student can even create a new operating-system distribution. Just a fewyearsago,it was difficult or impossible to get access to source code. Now, such access is limited onlybyhow much interest, time, and disk space a student has.

Chapter 2a: Operating-SystemServices

Outline

- Operating System Services
- User and Operating System-Interface
- System Calls
- System Services
- Linkers and Loaders
- Why Applications are Operating System Specific

Objectives

- Identify services provided by an operating system
- Illustrate how system calls are used to provide operatingsystem services

Operating SystemServices

- Operating systems provide an environment for executionof programsand services to programs and users

- One set of operating-system services provides functions that arehelpful to the user:
  - **User interface** - Almost all operating systems haveauser interface (**UI**).
    - 4 Varies between **Command-Line** (**CLI**), **GraphicsUser Interface** (**GUI**), **touch-screen, Batch** • **Program execution** - The system must be able toloadaprograminto memory and to run that program, end execution, either normally or abnormally (indicating error)
  - **I/O operations** - A running program may require I/O, whichmayinvolve a file or an I/O device

.

Operating SystemServices(Cont.)

- One set of operating-system services provides functions that arehelpful to the user (Cont.):

- **File-system manipulation** - The file systemis of particularinterest. Programs need to read and write files anddirectories,create and delete them, search them, list file Information, permission management.
- **Communications** – Processes may exchange information, onthesame computer or between computers over a network
  - 4 Communications may be via shared memory or throughmessage passing (packets moved by the OS)

Operating SystemServices(Cont.)

- One set of operating-system services provides functions that arehelpful to the user (Cont.):

- **Error detection** – OS needs to be constantly awareof possibleerrors

  4 May occur in the CPU and memory hardware, inI/Odevices,in user program

  4 For each type of error, OS should take the appropriateactionto ensure correct and consistent computing

  4 Debugging facilities can greatly enhance the user
  ’sand
  programmer’s abilities to efficiently use the system

Operating SystemServices(Cont.)

- Another set of OS function exists for ensuring the efficient operationofthe system itself via resource sharing • **Resource allocation -** When multiple users or multiplejobsrunning concurrently, resources must be allocatedtoeachof them

    4 Many types of resources - CPU cycles, main memory, filestorage, I/O devices.

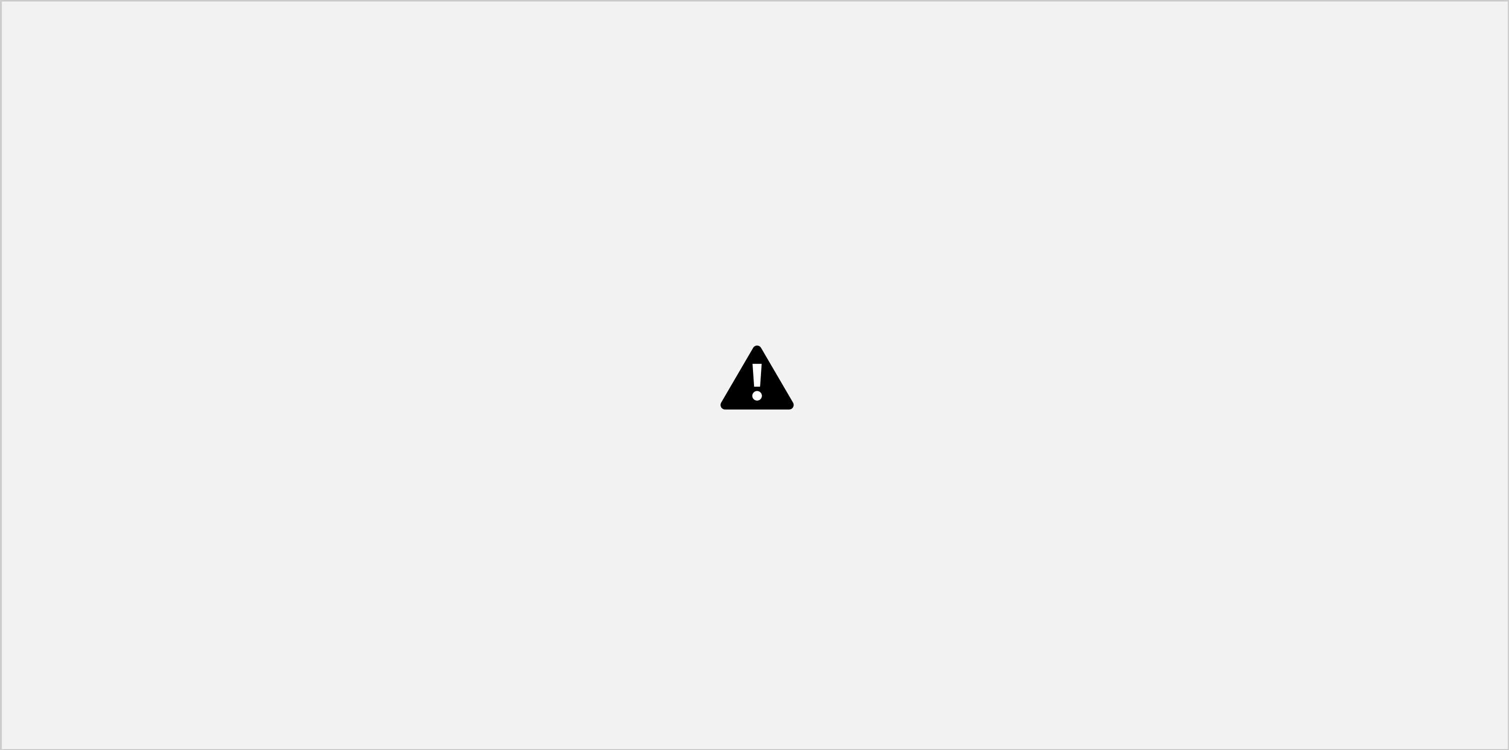
  - **Logging -** To keep track of which users use howmuchandwhatkinds of computer resources

  - **Protection and security -** The owners of informationstoredinamultiuser or networked computer systemmay want tocontrol useof that information, concurrent processes should not interferewitheach other

    4 **Protection** involves ensuring that all access tosystemresources is controlled

    4 **Security** of the system from outsiders requires user authentication, extends to defending external I/Odevicesfrom invalid access attempts

A View of OperatingSystemServices



User OperatingSystemInterface

- CLI -- command line interpreter
  - allows direct command entry
- GUI – graphical user interface
- Touchscreen Interfaces
- Batch

CLI

- Sometimes implemented in kernel, sometimes by systemsprogram
- Sometimes multiple flavors implemented – **shells** ▪ Primarily fetches a command from user and executes it ▪ Sometimes commands built-in, sometimes just names of programs

- If the latter, adding new features doesn't require shell modification

Bourne Shell CommandInterpreter

GUI

- User-friendly **desktop** metaphor interface
  - Usually mouse, keyboard, and monitor
  - **Icons** represent files, programs, actions, etc.
  - Various mouse buttons over objects in the interfacecausevariousactions (provide information, options, execute function, opendirectory (known as a **folder**)
  - Invented at Xerox PARC
- Many systems now include both CLI and GUI interfaces
  - Microsoft Windows is GUI with CLI “command”shell
  - Apple Mac OS X is “Aqua” GUI interface with UNIXkernel underneath and shells available
  - Unix and Linux have CLI with optional GUI interfaces(CDE, KDE,GNOME)