CS162
Operating Systems and Systems Programming
Lecture 2

History of the World Parts 1—5
Operating Systems Structures

September 1st, 2010

Prof. John Kubiatowicz

http://inst.eecs.berkeley.edu/~cs162

Review: Virtual Machine Abstraction

Application

Virtual Machine Interface

Operating System

Physical Machine Interface

Hardware

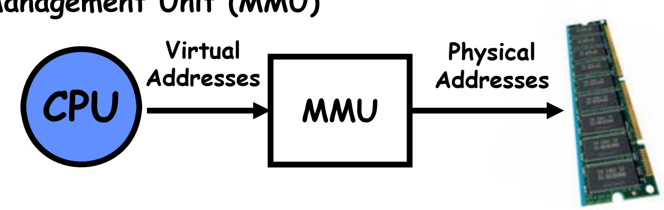
- Software Engineering Problem:
  - Turn hardware/software quirks ⇒ what programmers want/need
  - Optimize for convenience, utilization, security, reliability, etc…
- For Any OS area (e.g. file systems, virtual memory, networking, scheduling):
  - What's the hardware interface? (physical reality)
  - What's the application interface? (nicer abstraction)

Review: Protecting Processes from Each Other

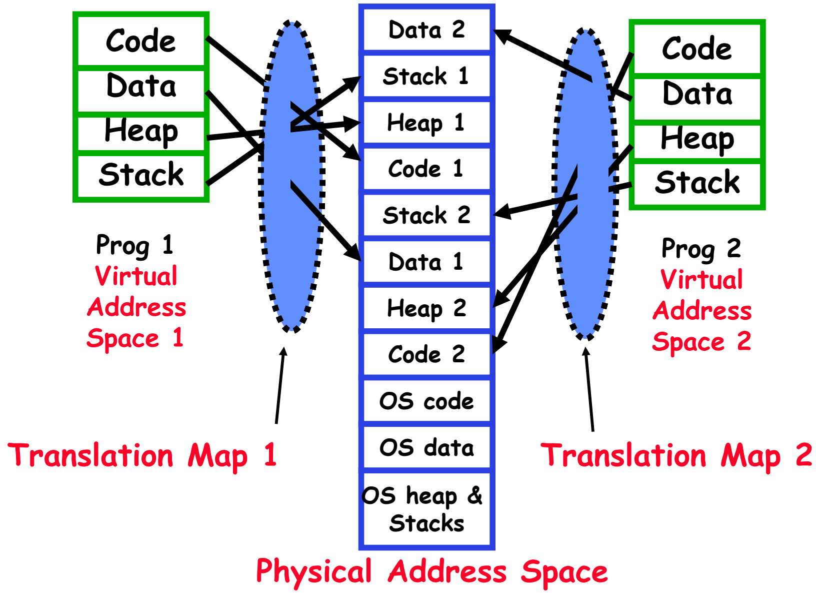
- Problem: Run multiple applications in such a way that they are protected from one another
- Goal:
  - Keep User Programs from Crashing OS
  - Keep User Programs from Crashing each other
  - [Keep Parts of OS from crashing other parts?]
- (Some of the required) Mechanisms:
  - Address Translation
  - Dual Mode Operation
- Simple Policy:
  - Programs are not allowed to read/write memory of other Programs or of Operating System

Review: Address Translation

- Address Space
  - A group of memory addresses usable by something
  - Each program (process) and kernel has potentially different address spaces.
- Address Translation:
  - Translate from Virtual Addresses (emitted by CPU) into Physical Addresses (of memory)
  - Mapping *often* performed in Hardware by Memory Management Unit (MMU)



Review: Example of Address Translation



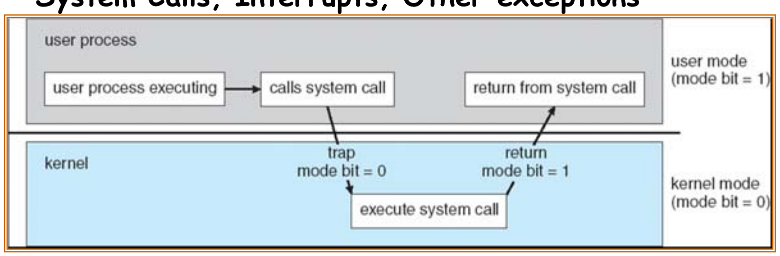
Goals for Today

- Finish Protection Example
- History of Operating Systems
  - Really a history of resource-driven choices
- Operating Systems Structures
- Operating Systems Organizations
- Abstractions and layering

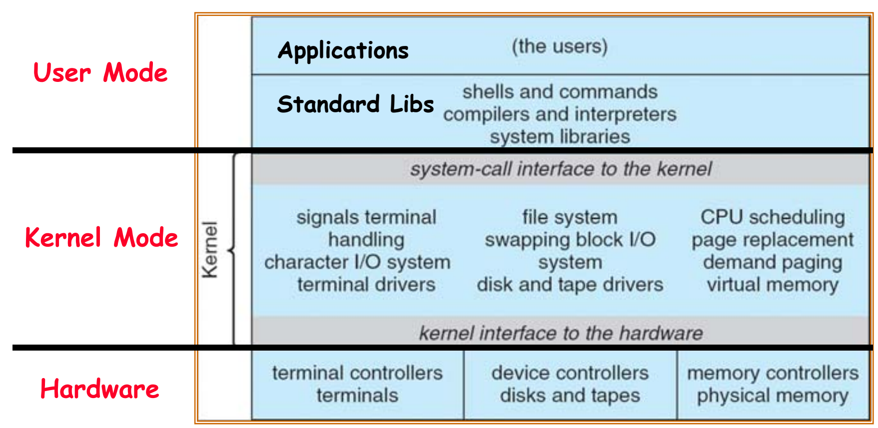
Note: Some slides and/or pictures in the following are adapted from slides ©2005 Silberschatz, Galvin, and Gagne. Many slides generated from lecture notes by Joseph.

The other half of protection: Dual Mode Operation

- Hardware provides at least two modes:
  - "Kernel" mode (or "supervisor" or "protected")
  - "User" mode: Normal programs executed
- Some instructions/ops prohibited in user mode:
  - Example: cannot modify page tables in user mode
    - » Attempt to modify ⇒ Exception generated
- Transitions from user mode to kernel mode:
  - System Calls, Interrupts, Other exceptions



UNIX System Structure



Moore's Law Change Drives OS Change

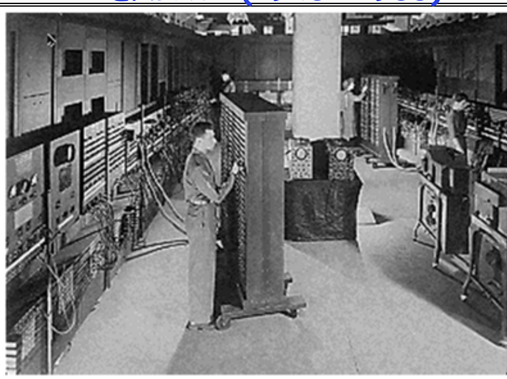
	1981	2010	Factor
CPU MHz, Cycles/inst	10 3—10	Quad 3G 0.25—0.5	1,200 6—40
DRAM capacity	128KB	8GB	65536
Disk capacity	10MB	2TB	200,000
Net bandwidth	9600 b/s	1 Gb/s	110,000
# addr bits	16	64	4
#users/machine	10s	≤ 1	≤ 0.1
Price	$25,000	$4,000	0.16

Typical academic computer 1981 vs 2010

Moore's law effects

- Nothing like this in any other area of business
- Transportation in over 200 years:
  - 2 orders of magnitude from horseback @10mph to Concorde @1000mph
  - Computers do this every decade (at least until 2002)!
- What does this mean for us?
  - Techniques have to vary over time to adapt to changing tradeoffs
- I place a lot more emphasis on principles
  - The key concepts underlying computer systems
  - Less emphasis on facts that are likely to change over the next few years…
- Let's examine the way changes in $/MIP has radically changed how OS's work

Dawn of time
ENIAC: (1945—1955)

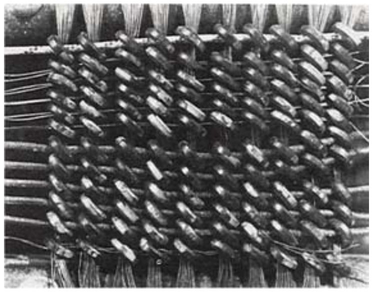


- "The machine designed by Drs. Eckert and Mauchly was a monstrosity. When it was finished, the ENIAC filled an entire room, weighed thirty tons, and consumed two hundred kilowatts of power."
- http://ei.cs.vt.edu/~history/ENIAC.Richey.HTML

History Phase 1 (1948—1970)
Hardware Expensive, Humans Cheap

- When computers cost millions of $'s, optimize for more efficient use of the hardware!
  - Lack of interaction between user and computer
- User at console: one user at a time
- Batch monitor: load program, run, print
- Optimize to better use hardware
  - When user thinking at console, computer idle⇒BAD!
  - Feed computer batches and make users wait
  - Autograder for this course is similar
- *No protection:* what if batch program has bug?

Core Memories (1950s & 60s)



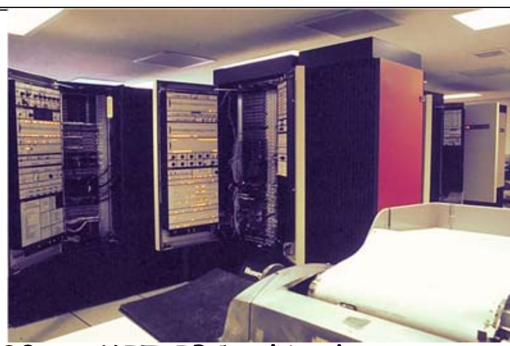
The first magnetic core memory, from the IBM 405 Alphabetical Accounting Machine.

- Core Memory stored data as magnetization in iron rings
  - Iron "cores" woven into a 2-dimensional mesh of wires
  - Origin of the term "Dump Core"
  - Rumor that IBM consulted Life Saver company
- See: http://www.columbia.edu/acis/history/core.html

History Phase 1½ (late 60s/early 70s)

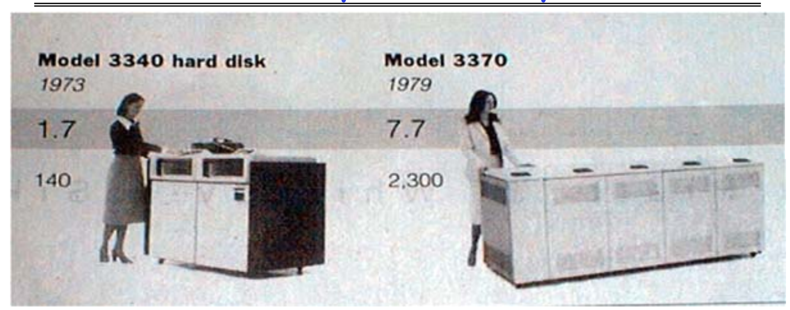
- **Data channels, Interrupts:** overlap I/O and compute
  - DMA – Direct Memory Access for I/O devices
  - I/O can be completed asynchronously
- **Multiprogramming:** several programs run simultaneously
  - Small jobs not delayed by large jobs
  - More overlap between I/O and CPU
  - Need memory protection between programs and/or OS
- **Complexity gets out of hand:**
  - Multics: announced in 1963, ran in 1969
    - » 1777 people "contributed to Multics" (30-40 core dev)
    - » Turing award lecture from Fernando Corbató (key researcher): "On building systems that will fail"
  - OS 360: released with 1000 known bugs (APARs)
    - » "Anomalous Program Activity Report"
- **OS finally becomes an important science:**
  - How to deal with complexity???
  - UNIX based on Multics, but vastly simplified

A Multics System (Circa 1976)



- The 6180 at MIT IPC, skin doors open, circa 1976:
  - "We usually ran the machine with doors open so the operators could see the AQ register display, which gave you an idea of the machine load, and for convenient access to the EXECUTE button, which the operator would push to enter BOS if the machine crashed."
- http://www.multicians.org/multics-stories.html

Early Disk History



1973:	1979:
1. 7 Mbit/sq. in	7. 7 Mbit/sq. in
140 MBytes	2,300 MBytes

Contrast: Seagate 2TB, 400 GB/SQ in, 3½ in disk, 4 platters

Administrivia

- Waitlist:
  - All CS/EECS seniors should be in the class
  - Remaining:
    - » 18 CS/EECS juniors,
    - » 4 grad students
    - » 2 non CS/EECS seniors
- Cs162-xx accounts:
  - We have more forms for those who didn't get one
  - If you haven't logged in yet, you need to do so
- Nachos readers:
  - TBA: Will be down at Copy Central on Hearst
  - Will include lectures and printouts of all of the code
- Video "Screencast" archives available off lectures page
  - If have mp4 player, just click on the title of a lecture
  - Otherwise, click on link at top middle of lecture page
- No slip days on first design document for each phase
  - Need to get design reviews in on time
- Don't know Java well?
  - Perhaps try CS 9G self-paced Java course

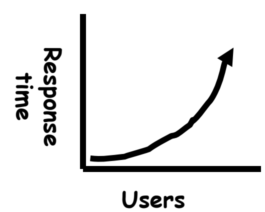
Administriva: Time to start thinking about groups

- Project Signup: Not quite ready, but will be
  - 4-5 members to a group
    - » Everyone in group must be able to *actually* attend same section
    - » The sections assigned to you by Telebears are temporary!
  - Only submit once per group!
    - » Everyone in group must have logged into their cs162-xx accounts once before you register the group
    - » Make sure that you select at least 2 potential sections
    - » Due Tuesday 9/7 by 11:59pm
- Sections:
  - Watch for section assignments next Wednesday/Thursday
  - Attend new sections next week: Telebears sections this Friday

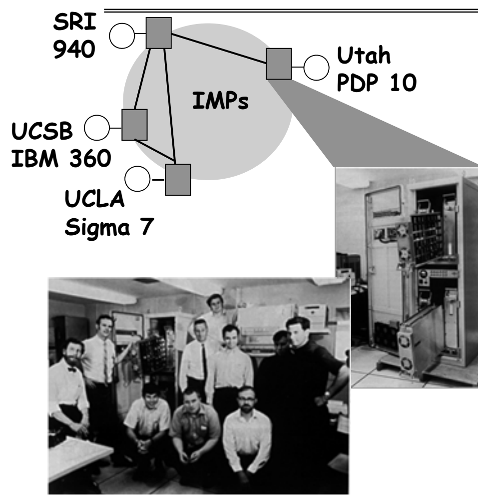
Section	Time	Location	TA
101	F 9:00A-10:00A	85 Evans	Christos Stergiou
102	F 10:00A-11:00A	6 Evans	Angela Juang
103	F 11:00A-12:00P	2 Evans	Angela Juang
104	F 12:00P-1:00P	75 Evans	Hilfi Alkaff
105 (New)	F 1:00P-2:00P	85 Evans	Christos Stergiou

History Phase 2 (1970 – 1985) Hardware Cheaper, Humans Expensive

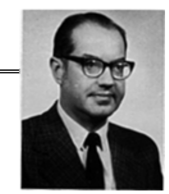
- Computers available for tens of thousands of dollars instead of millions
- OS Technology maturing/stabilizing
- *Interactive* timesharing:
  - Use cheap terminals (~$1000) to let multiple users interact with the system at the same time
  - Sacrifice CPU time to get better response time
  - Users do debugging, editing, and email online
- Problem: Thrashing
  - Performance very non-linear response with load
  - Thrashing caused by many factors including
    - » Swapping, queueing



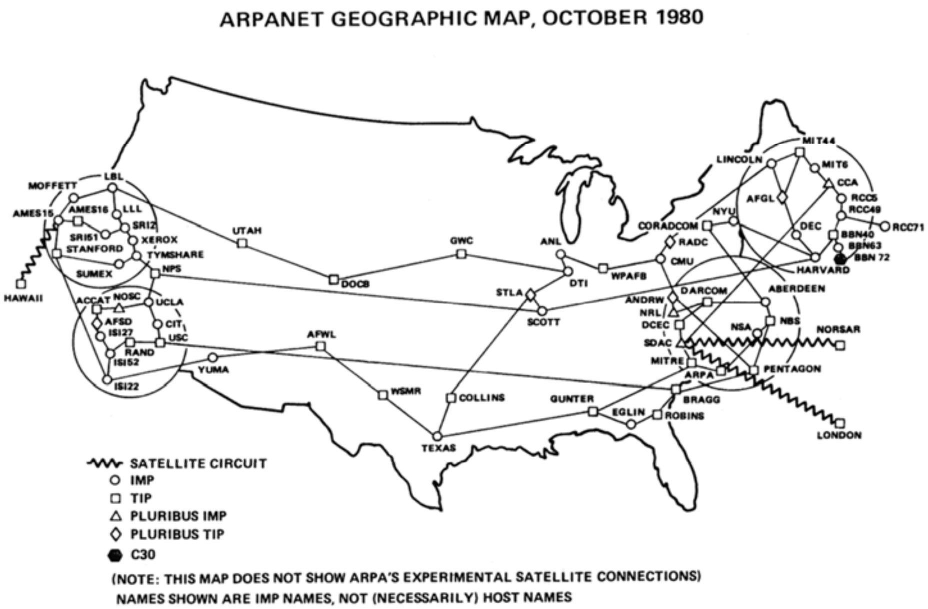
The ARPANet (1968-1970's)



BBN team that implemented the interface message processor (IMP)



- Paul Baran
  - RAND Corp, early 1960s
  - Communications networks that would survive a major enemy attack
- ARPANet: Research vehicle for "Resource Sharing Computer Networks"
  - 2 September 1969: UCLA first node on the ARPANet
  - December 1969: 4 nodes connected by 56 kbps phone lines
  - 1971: First Email
  - 1970's: <100 computers



History Phase 3 (1981—)
Hardware Very Cheap, Humans Very Expensive

- Computer costs $1K, Programmer costs $100K/year
  - If you can make someone 1% more efficient by giving them a computer, it's worth it!
  - Use computers to make people more efficient
- Personal computing:
  - Computers cheap, so give everyone a PC
- Limited Hardware Resources Initially:
  - OS becomes a subroutine library
  - One application at a time (MSDOS, CP/M, …)
- Eventually PCs become powerful:
  - OS regains all the complexity of a "big" OS
  - multiprogramming, memory protection, etc (NT,OS/2)
- Question: As hardware gets cheaper does need for OS go away?

History Phase 3 (con't)
Graphical User Interfaces

- CS160 ⇒ All about GUIs
- Xerox Star: 1981
  - Originally a research project (Alto)
  - First "mice", "windows"
- Apple Lisa/Machintosh: 1984
  - "Look and Feel" suit 1988
- Microsoft Windows:
  - Win 1.0 (1985)
  - Win 3.1 (1990)
  - Win 95 (1995)
  - Win NT (1993)
  - Win 2000 (2000)
  - Win XP (2001)
  - Win Vista (2007)

$\left.\begin{array}{l}\text{Win 1.0 (1985)}\\ \text{Win 3.1 (1990)}\\ \text{Win 95 (1995)}\end{array}\right\}$ Single Level

$\left.\text{Win NT (1993)}\right\}$ HAL/Protection

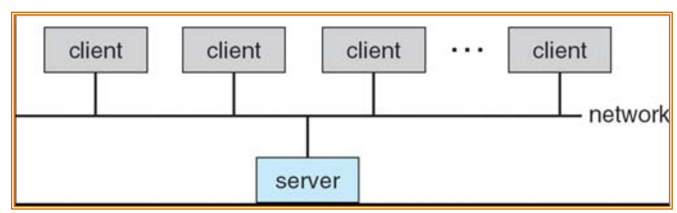
$\left.\begin{array}{l}\text{Win 2000 (2000)}\\ \text{Win XP (2001)}\\ \text{Win Vista (2007)}\end{array}\right\}$ No HAL/ Full Prot

Xerox Star

Windows 3.1

History Phase 4 (1988—): Distributed Systems

- Networking (Local Area Networking)
  - Different machines share resources
  - Printers, File Servers, Web Servers
  - Client – Server Model
- Services
  - Computing
  - File Storage

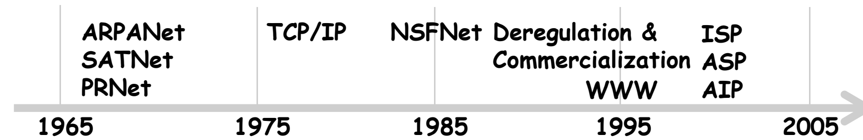


History Phase 4 (1988—): Internet

- Developed by the research community
  - Based on open standard: Internet Protocol
  - Internet Engineering Task Force (IETF)
- Technical basis for many other types of networks
  - Intranet: enterprise IP network
- Services Provided by the Internet
  - Shared access to computing resources: telnet (1970's)
  - Shared access to data/files: FTP, NFS, AFS (1980's)
  - Communication medium over which people interact
    - » email (1980's), on-line chat rooms, instant messaging (1990's)
    - » audio, video (1990's, early 00's)
  - Medium for information dissemination
    - » USENET (1980's)
    - » WWW (1990's)
    - » Audio, video (late 90's, early 00's) – replacing radio, TV?
    - » File sharing (late 90's, early 00's)

ARPANet Evolves into Internet

- First E-mail SPAM message: 1 May 1978 12:33 EDT
- 80-83: TCP/IP, DNS; ARPANET and MILNET split
- 85-86: NSF builds NSFNET as backbone, links 6 Supercomputer centers, 1.5 Mbps, 10,000 computers
- 87-90: link regional networks, NSI (NASA), ESNet (DOE), DARTnet, TWBNet (DARPA), 100,000 computers



SATNet: Satelite network
PRNet: Radio Network

What is a Communication Network? (End-system Centric View)

- Network offers one basic service: move information
  - Bird, fire, messenger, truck, telegraph, telephone, Internet …
  - Another example, transportation service: move objects
    - » Horse, train, truck, airplane ...
- What distinguish different types of networks?
  - The services they provide
- What distinguish the services?
  - Latency
  - Bandwidth (Highest BW? "Sneakernet")
  - Loss rate
  - Number of end systems
  - Service interface (how to invoke the service?)
  - Others
    - » Reliability, unicast vs. multicast, real-time...

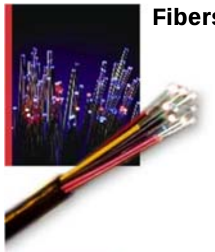
What is a Communication Network? (Infrastructure Centric View)

- Communication medium: electron, photon
- Network components:
  - Links – carry bits from one place to another (or maybe multiple places): fiber, copper, satellite, …
  - Interfaces – attach devices to links
  - Switches/routers – interconnect links: electronic/optic, crossbar/Banyan
  - Hosts – communication endpoints: workstations, PDAs, cell phones, toasters
- Protocols – rules governing communication between nodes
  - TCP/IP, ATM, MPLS, SONET, Ethernet, X.25
- Applications: Web browser, X Windows, FTP, ...

Network Components (Examples)

Links

Fibers



Coaxial Cable

Interfaces

Ethernet card

Wireless card

Switches/routers

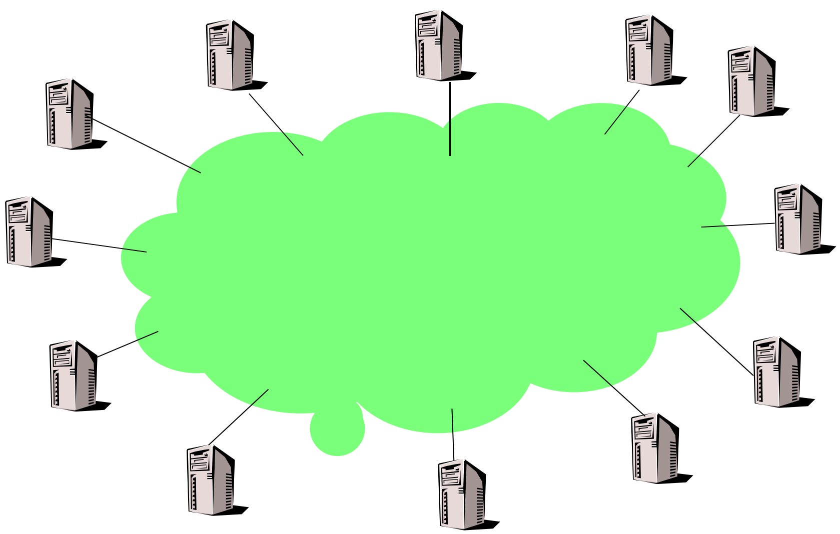
Large router

Telephone switch

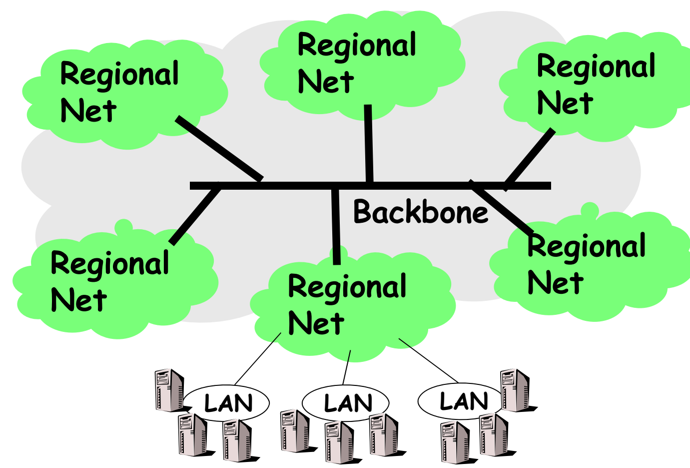
Types of Networks

- Geographical distance
  - Local Area Networks (LAN): Ethernet, Token ring, FDDI
  - Metropolitan Area Networks (MAN): DQDB, SMDS
  - Wide Area Networks (WAN): X.25, ATM, frame relay
  - Caveat: LAN, MAN, WAN may mean different things
    - » Service, network technology, networks
- Information type
  - Data networks vs. telecommunication networks
- Application type
  - Special purpose networks: airline reservation network, banking network, credit card network, telephony
  - General purpose network: Internet

Network "Cloud"

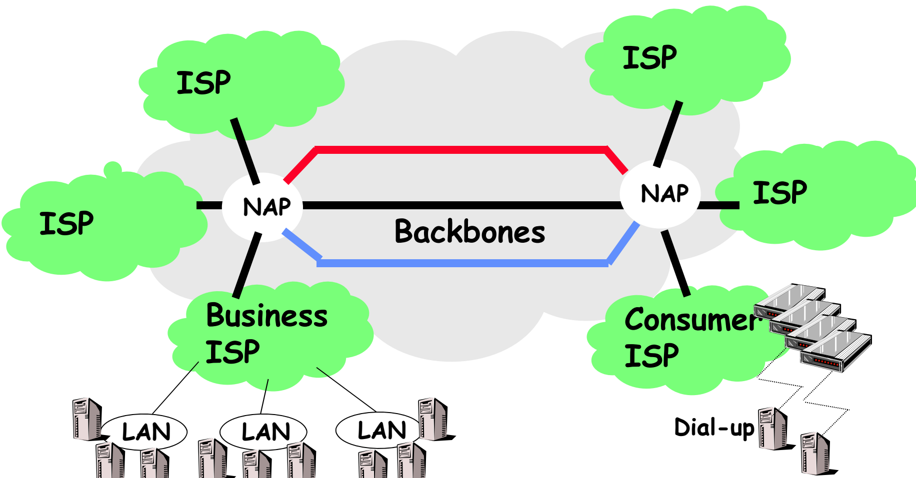


Regional Nets + Backbone



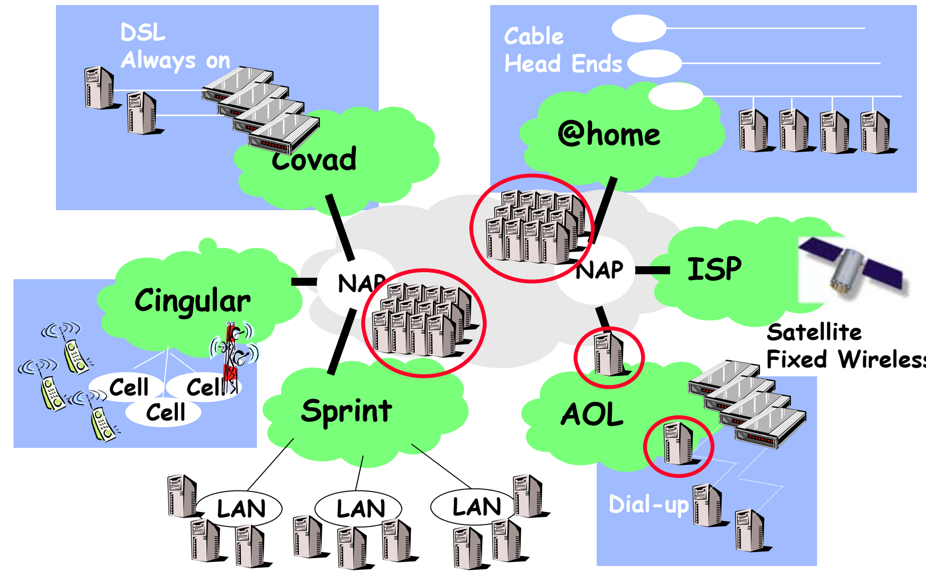
LAN: Local Area Network

Backbones + NAPs + ISPs



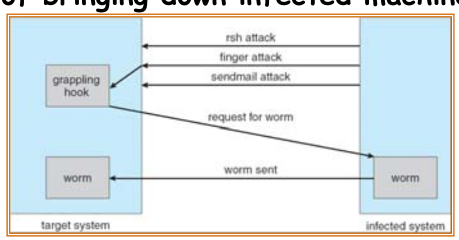
ISP: Internet Service Provide
NAP: Network Access Point

Computers Inside the Core



The Morris Internet Worm (1988)

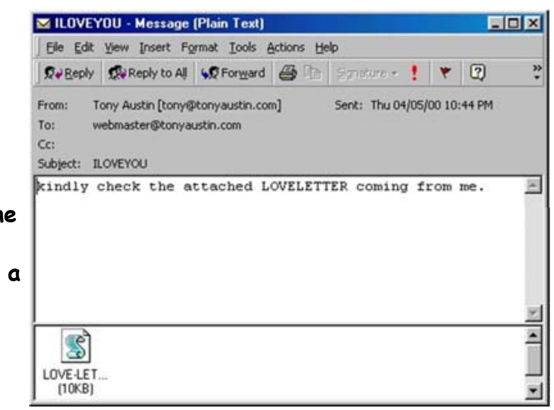
- Internet worm (Self-reproducing)
  - Author Robert Morris, a first-year Cornell grad student
  - Launched close of Workday on November 2, 1988
  - Within a few hours of release, it consumed resources to the point of bringing down infected machines



- Techniques
  - Exploited UNIX networking features (remote access)
  - Bugs in *finger* (buffer overflow) and *sendmail* programs (debug mode allowed remote login)
  - Dictionary lookup-based password cracking
  - Grappling hook program uploaded main worm program

LoveLetter Virus (May 2000)

- E-mail message with VBScript (simplified Visual Basic)
- Relies on Windows Scripting Host
  - Enabled by default in Win98/2000
- User clicks on attachment➔ infected!
  - E-mails itself to everyone in Outlook address book
  - Replaces some files with a copy of itself
  - Searches all drives
  - Downloads password cracking program
- 60-80% of US companies infected and 100K European servers



History Phase 5 (1995—): Mobile Systems

- Ubiquitous Mobile Devices
  - Laptops, PDAs, phones
  - Small, portable, and inexpensive
    - » Recently twice as many smart phones as PDAs
    - » Many computers/person!
  - Limited capabilities (memory, CPU, power, etc…)
- Wireless/Wide Area Networking
  - Leveraging the infrastructure
  - Huge distributed pool of resources extend devices
  - Traditional computers split into pieces. Wireless keyboards/mice, CPU distributed, storage remote
- Peer-to-peer systems
  - Many devices with equal responsibilities work together
  - Components of "Operating System" spread across globe

CITRIS's Model: A Societal Scale Information System

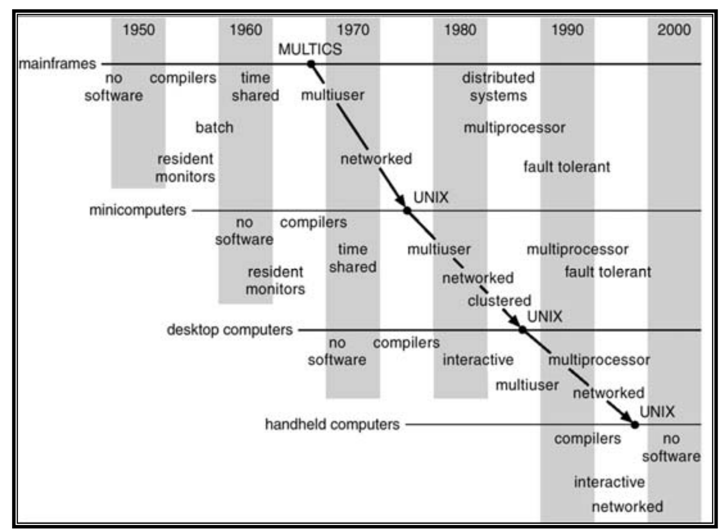
- Center for Information Technology Research in the Interest of Society
- The Network is the OS
  - Functionality spread throughout network

Datacenter is the Computer

- *(From Luiz Barroso's talk at RAD Lab 12/11)*
- Google *program* == Web search, Gmail,…
- Google *computer* ==

  - Thousands of computers, networking, storage
- Warehouse-sized facilities and workloads may be unusual today but are likely to be more common in the next few years

Migration of Operating-System Concepts and Features



History of OS: Summary

- Change is continuous and OSs should adapt
  - Not: look how stupid batch processing was
  - But: Made sense at the time
- Situation today is much like the late 60s
  - Small OS: 100K lines
  - Large OS: 10M lines (5M for the browser!)
    - » 100-1000 people-years
- Complexity still reigns
  - NT developed (early to late 90's): Never worked well
  - Windows 2000/XP: Very successful
  - Windows Vista (aka "Longhorn") delayed many times
    - » Finally released in January 2007
    - » Promised by removing some of the intended technology
    - » Slow adoption rate, even in 2008/2009
- CS162: understand OSs to simplify them

Now for a quick tour of OS Structures

Operating Systems Components (What are the pieces of the OS)

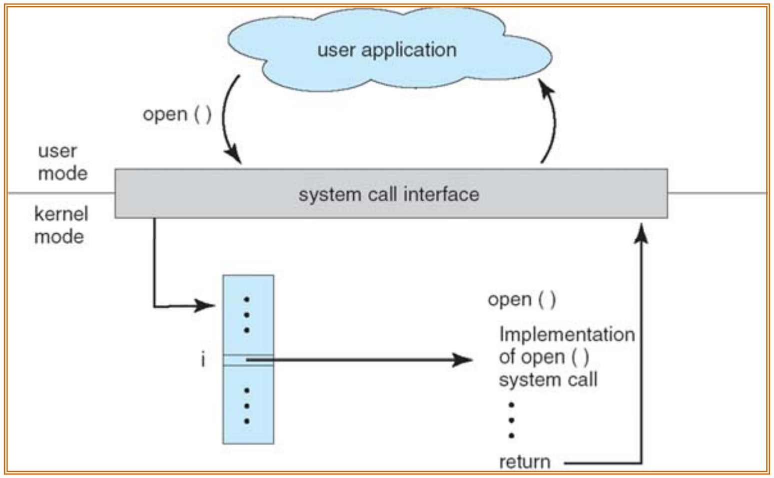
- Process Management
- Main-Memory Management
- I/O System management
- File Management
- Networking
- User Interfaces

Operating System Services (What things does the OS do?)

- Services that (more-or-less) map onto components
  - Program execution
    - » How do you execute concurrent sequences of instructions?
  - I/O operations
    - » Standardized interfaces to extremely diverse devices
  - File system manipulation
    - » How do you read/write/preserve files?
    - » Looming concern: How do you even find files???
  - Communications
    - » Networking protocols/Interface with CyberSpace?
- Cross-cutting capabilities
  - Error detection & recovery
  - Resource allocation
  - Accounting
  - Protection

System Calls (What is the API)

- See Chapter 2 of 7th edition or Chapter 3 of 6th

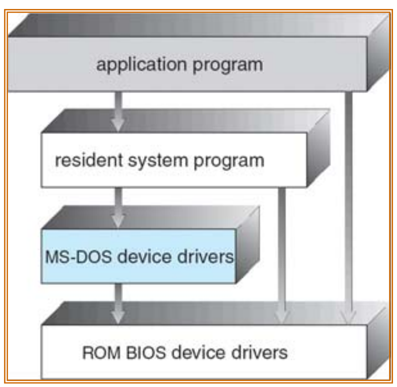


Operating Systems Structure (What is the organizational Principle?)

- Simple
  - Only one or two levels of code
- Layered
  - Lower levels independent of upper levels
- Microkernel
  - OS built from many user-level processes
- Modular
  - Core kernel with Dynamically loadable modules

Simple Structure

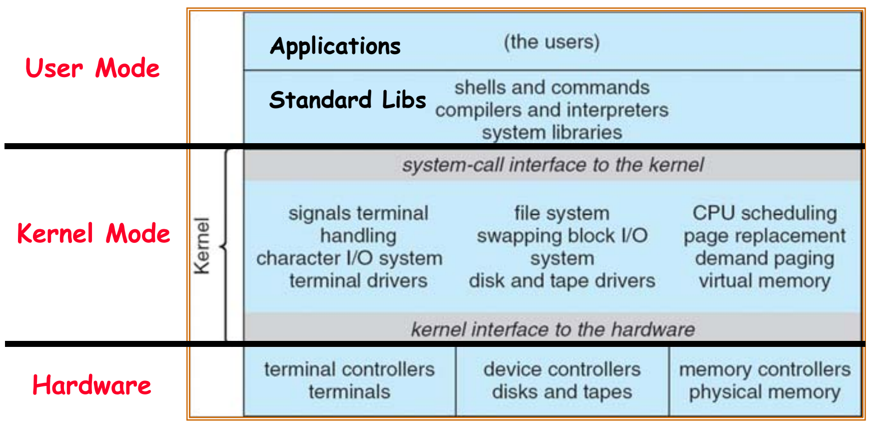
- MS-DOS – written to provide the most functionality in the least space
  - Not divided into modules
  - Interfaces and levels of functionality not well separated



UNIX: Also "Simple" Structure

- UNIX – limited by hardware functionality
- Original UNIX operating system consists of two separable parts:
  - Systems programs
  - The kernel
    - » Consists of everything below the system-call interface and above the physical hardware
    - » Provides the file system, CPU scheduling, memory management, and other operating-system functions;
    - » Many interacting functions for one level

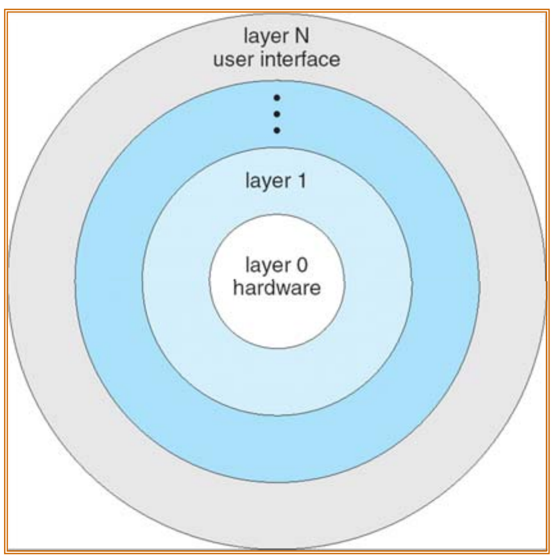
UNIX System Structure



Layered Structure

- Operating system is divided many layers (levels)
  - Each built on top of lower layers
  - Bottom layer (layer 0) is hardware
  - Highest layer (layer N) is the user interface
- Each layer uses functions (operations) and services of only lower-level layers
  - Advantage: modularity ⇒ Easier debugging/Maintenance
  - Not always possible: Does process scheduler lie above or below virtual memory layer?
    - » Need to reschedule processor while waiting for paging
    - » May need to page in information about tasks
- Important: Machine-dependent vs independent layers
  - Easier migration between platforms
  - Easier evolution of hardware platform
  - Good idea for you as well!

Layered Operating System

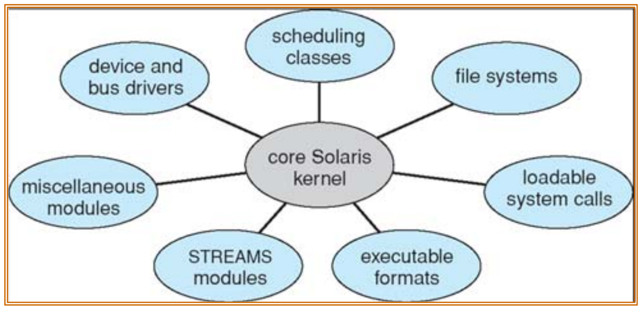


Microkernel Structure

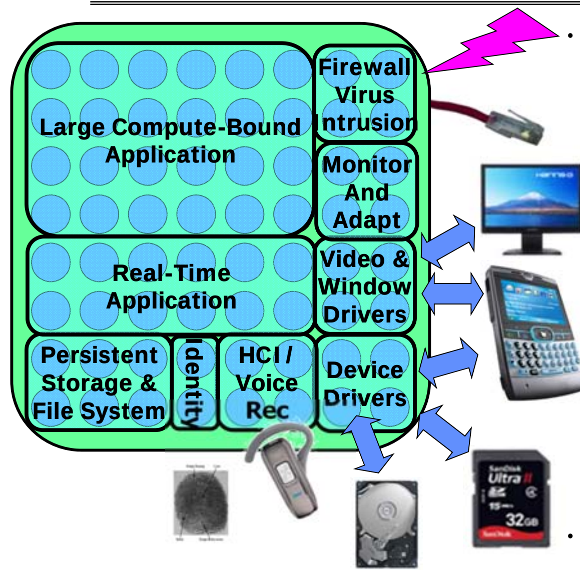
- Moves as much from the kernel into "*user*" space
  - Small core OS running at kernel level
  - OS Services built from many independent user-level processes
- Communication between modules with message passing
- Benefits:
  - Easier to extend a microkernel
  - Easier to port OS to new architectures
  - More reliable (less code is running in kernel mode)
  - Fault Isolation (parts of kernel protected from other parts)
  - More secure
- Detriments:
  - Performance overhead severe for naïve implementation

Modules-based Structure

- Most modern operating systems implement modules
  - Uses object-oriented approach
  - Each core component is separate
  - Each talks to the others over known interfaces
  - Each is loadable as needed within the kernel
- Overall, similar to layers but with more flexible



Partition Based Structure for Multicore chips?



- Normal Components split into pieces
  - Device drivers (Security/Reliability)
  - Network Services (Performance)
    - » TCP/IP stack
    - » Firewall
    - » Virus Checking
    - » Intrusion Detection
  - Persistent Storage (Performance, Security, Reliability)
  - Monitoring services
    - » Performance counters
    - » Introspection
  - Identity/Environment services (Security)
    - » Biometric, GPS, Possession Tracking
- Applications Given Larger Partitions
  - Freedom to use resources arbitrarily

Implementation Issues (How is the OS implemented?)

- Policy vs. Mechanism
  - Policy: What do you want to do?
  - Mechanism: How are you going to do it?
  - Should be separated, since both change
- Algorithms used
  - Linear, Tree-based, Log Structured, etc…
- Event models used
  - threads vs event loops
- Backward compatability issues
  - Very important for Windows 2000/XP
- System generation/configuration
  - How to make generic OS fit on specific hardware

Conclusion

- Rapid Change in Hardware Leads to changing OS
  - Batch ⇒ Multiprogramming ⇒ Timeshare ⇒ Graphical UI ⇒ Ubiquitous Devices ⇒ Cyberspace/Metaverse/??
- OS features migrated from mainframes ⇒ PCs
- Standard Components and Services
  - Process Control
  - Main Memory
  - I/O
  - File System
  - UI
- Policy vs Mechanism
  - Crucial division: not always properly separated!
- Complexity is always out of control
  - However, "Resistance is NOT Useless!"