Lecture Topics

• Today: Operating System Overview  
  (Stallings, chapter 2.1-2.4, 2.8-2.10)

• Next: continued

Announcements

• Consulting hours posted

• Self-Study Exercise #2 posted

• Project #1 posted
What is an operating system?

- Software which:
  - Controls the resources of a computer
  - Provides an interface for user applications to hardware

- Advantages over working with the bare machine:
  - Programmer productivity
  - Better utilization of resources
  - Fair access to resources
  - Correctness, security

- Disadvantage: OS consumes resources

Limited Direct Execution

- A modern OS does not monitor or interpret the execution of individual instructions in a user process

- Instead, the OS relinquished control of the CPU to user processes (user processes run directly on the hardware)

- Potential problem: OS must have ability to regain control of the CPU (interrupts)

- Called *limited direct execution*
Virtual Memory

- When a process reads and writes to computer memory, the addresses it uses are not physical memory addresses
- Rather, the OS and architecture collaborate to map user’s logical addresses to physical addresses
- Called virtual memory

Major OS Functions

- Process and thread management
  - process control: creation, termination, suspension, resumption, scheduling, resource allocation, etc.
- Memory management
  - allocation/deallocation of computer memory, implementation of virtual memory, demand paging
- Interprocess (or interthread) communication
  - signals, pipes, messages, events, semaphores, shared memory, etc.
Major OS Functions (cont.)

- I/O and device drivers
  - character and block device management, handling interrupts, direct memory access (DMA), etc.

- File management
  - file system implementation, memory-resident data structures, buffer cache, device files, etc.

- Network management
  - sockets or equivalent, TCP/IP protocols, etc.

Major Advances (Denning)

- Processes
- Memory management
- Information protection and security
- Scheduling and resource management
Processes

Memory Management
Scheduling and Resource Management

Origins of Operating Systems
Serial Processing

• Earliest computers had no operating system
  – programmers interacted directly with the hardware

• Computers ran from a console with display lights, toggle switches, some input device, and a printer
  – Load compiler, program (via card reader and/or tapes)
  – Compile, link, run, in steps
  – Errors indicated by lights on front panel

• Users had access to the computer in “series”
  (sign-up sheet!)

Batch Processing

• Much time was lost configuring a computer between one job and the next (setup and teardown time)

• The earliest “operating systems” were designed to address this problem
  – batches of jobs loaded onto tape, executed sequentially by the OS
  – first OS: GM Research Labs developed OS for IBM 701
Major shift: Multiprogramming

- System resources underutilized

- Multiprogramming
  - also called time sharing, multitasking
  - Sharing a resource (e.g., computer) among users, such that each seems to have the whole machine
  - Pauses by one user filled with work for another
  - Need to store program state, quickly switch among user applications
  - Key mechanism: interrupts
Multiprogrammed Batch Systems

- Can be used to handle multiple interactive jobs
- Processor time is shared among multiple users
- Multiple users simultaneously access the system through terminals, with the OS interleaving the execution of each user program in a short burst or quantum of computation
Comparison

• Multiprogrammed batch vs time sharing systems

<table>
<thead>
<tr>
<th>Principal objective</th>
<th>Batch Multiprogramming</th>
<th>Time Sharing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source of directives to operating system</td>
<td>Maximize processor use</td>
<td>Minimize response time</td>
</tr>
<tr>
<td>Job control language commands provided with the job</td>
<td>Commands entered at the terminal</td>
<td></td>
</tr>
</tbody>
</table>

Unix Beginnings

• 1969 - first Unix system
  – (mostly) Thompson at Bell Labs
  – implemented in assembly language on DEC PDP-7

• 1970 - development of C
  – Thompson and Dennis Ritchie port Unix to a PDP-11/20
  – Ritchie designs and writes the first C compiler
  – Goal: “high-level” language for writing a portable OS

• 1972 - C implementation of Unix
  – Ritchie and Thompson rewrite the Unix kernel in C.
AT&T’s Unix

• 1970s - free distribution of Unix
  – UNIX source code distributed freely to universities, due to marketing restrictions on AT&T, parent of Bell Labs
  – UNIX gains favor within the academic/research community

• 1984 - AT&T Unix goes commercial
  – January 1, 1984 - Divestiture of Bell System
  – AT&T can enter new markets, including computers
  – AT&T releases the commercial UNIX System V

Traditional UNIX Architecture
Traditional UNIX Kernel

- Monolithic kernel (simple!)
  - Clean design (originally)
  - No complex recovery scheme
- Designed for programmers
  - Many system calls, libraries
  - Shells, scripts, standard I/O
  - Fork/exec, pipes, signals, IPC
  - Windowing with X11 (later)
  - Tree-structured file system
  - Simple file access descriptors
  - And many others…

Unix Features
UNIX and derivatives

- Originally developed at AT&T Bell Labs
- Derivatives:
  - BSD – UC Berkeley
  - Mac OS – Apple
  - AIX – IBM
  - Solaris – Sun Microsystems
  - HP-UX – Hewlett-Packard
- UNIX-like:
  - Minix – Tannenbaum
  - Linix – Torvalds
GNU (GNU’s Not Unix!)

- 1983 - In response to commercialization of Unix, Richard Stallman starts the GNU Project
  - Goal: create “complete Unix-compatible software system” composed entirely of free software
  - 1985 - Stallman creates Free Software Foundation
  - 1989 - Stallman writes GNU General Public License
    - Copyleft license: requires derived works to be available under the same copyleft.
  - By early 1990’s, many programs (libraries, shells, etc) available, but kernel work was stalled.

Linux

- Andrew Tanenbaum had developed MINIX as a “microkernel-based, Unix-like” OS for educational purposes.
- 1991 - Linus Torvalds develops Linux kernel as a student project
  - Based in part on MINIX and Unix System V
  - Wanted a free Unix-like, industrial strength OS
  - Originally developed only for Intel x86 hardware; since then, has been ported to more platforms than any other OS
Linux Impact

• Linux made available under GNU Public License
• Many, many distributions of Linux, packaged with various configurations, have been available over the past 20 years
• Torvalds continues to oversee kernel work, and Stallman still runs Free Software Foundation
• Linux continues to evolve:
  – SMP kernel, demand-loadable modules
  – Support for 64-bit processors
  – Ever-expanding set of modules, new file systems
  – And many others…

Linux Impact

• Although desktop OS market is dominated by Windows versions, Linux has major share of other areas:
  – Top 10 fastest supercomputers run Linux
  – 95% of top 500 supercomputers run Linux
  – 2nd quarter 2013: 79% of smartphones sold in world used Android
  – 33% of server market (65% for Unix overall)
Big vs. Small

One of the major historical debates is how OS should be designed:

- Monolithic kernel: kernel provides most OS services

  or

- Microkernel: kernel provides as little as possible; user-level servers provide most OS services
Monolithic Kernel vs. Microkernel

Monolithic Kernel based Operating System

- Application
- VFS
- IPC, Rte System
- Scheduler, Virtual Memory
- Device Drivers, Dispatcher, ...
- Hardware

Microkernel based Operating System

- Application
- UNIX Server
- Device Driver
- Rte Server
- Basic IPC, Virtual Memory, Scheduling
- Hardware

Monolithic Kernel

- All of the OS shares a common, protected address space, which executes in privileged mode
- Kernel performs all major functions of the OS
- System calls are made by trapping to kernel, which performs work on behalf of the process
- Traditional UNIX has a monolithic kernel
Microkernel-Based Systems

• Very small kernel provides only the most basic services to support processes and threads
  – Low-level process management, memory management, interprocess communication, interrupt handling

• Servers (often running at user-level, but with high priority) provide all other OS functions
  – full process management, virtual memory management and paging, file service, I/O, networking, system calls
  – user process sends message to server, which does work and returns results

Implications for OS Design

• Unix/Linux - monolithic kernel

• Mach – microkernel OS developed at Carnegie Mellon University in 1980s

• Windows (NT) – basis of current Windows versions
  – Heavily influenced initially by microkernel concepts
  – Has evolved toward monolithic kernel over the years

• Mac OS X
  – Combines Mach microkernel with BSD Unix
History

• The following slides are a brief history of some of the major developments in operating systems

ENIAC (1946)

• First general-purpose electronic computer
• Developed by UPenn for Army Research Lab
• Project led by
• Input: card reader
• Output: card punch; data printed by an accounting machine
• Used vacuum tubes – transistors not invented yet
• Stored data in decimal format
• 357 10-digit multiplications/sec, 35 divisions/sec
Atlas

University of Manchester batch system, late '50s early '60s
- designed for Ferranti computer
- Used spooling for scheduling jobs according to need for peripherals
  - (magnetic tape, paper tape reader, paper tape punches, line printers, card readers, card punches)
- innovations: device drivers, system calls, demand paging between core memory (cache) and a drum (main memory)

CTSS

- 1962 at MIT, Corbato et al.
  - implemented on IBM 7090 (32K memory)
  - experimental timesharing system eventually supported up to 32 interactive users via terminals
  - user processes were swapped to drum
  - multilevel feedback queue scheduling system
  - demonstrated convenience of timesharing
  - Follow-on project: Multics!
Multics Project

- 1966, joint project between MIT, GE, and Bell Labs
- Idea: design a timesharing computer system that could be accessed via telephone from anywhere
  - written mostly in PL/1 (not assembly language!)
  - implemented on GE 645, which supported paged-segmentation memory hardware.
  - virtual address: 18-bit segment number and 16-bit word offset (large!)
- Last known installation shut down in 2000!

Multics Features/Innovations

- Virtual address space merged into tree-like file system
- Per process stacks
- Concept of a shell, a process you can “replace” with some other process
- Multi-level feedback queue scheduling
- Security: access list associated with each file and set of protection rings for executing processes
- Dynamic linking of library routines (on demand)
- Supported multiple CPUs
Multics Ring Structure

- Protected domains numbered 0 to 7
- Processes executing in inner domains have more privileges than those in outer domains
- Separate (kernel) stack for each ring

Unintended Multics Contribution: Unix!

- Ken Thompson and Dennis Ritchie were Bell Labs engineers who worked on the Multics Project
- They (and others) felt Multics was overly complex and had too many layers
- Began work on a simpler operating system
  - OS comprises one layer
  - All application code (end applications, libraries, shells) run at another layer
Unix Beginnings

- **1969** - first Unix system
  - (mostly) Thompson at Bell Labs
  - implemented in assembly language on DEC PDP-7

- **1970** - development of C
  - Thompson and Dennis Ritchie port Unix to a PDP-11/20
  - Ritchie designs and writes the first C compiler
  - Goal: “high-level” language for writing a portable OS

- **1972** - C implementation of Unix
  - Ritchie and Thompson rewrite the Unix kernel in C.

---

PDP-11

---
AT&T’s Unix

• 1970s - free distribution of Unix
  – UNIX source code distributed freely to universities, due to marketing restrictions on AT&T, parent of Bell Labs
  – UNIX gains favor within the academic/research community

• 1984 - AT&T Unix goes commercial
  – January 1, 1984 - Divestiture of Bell System
  – AT&T can enter new markets, including computers
  – AT&T releases the commercial UNIX System V

BSD Unix

• 1979-1980 - Berkeley enters the picture
  – Ken Thompson had spent a sabbatical at Berkeley, among other things, teaching Unix
  – Bill Joy, a graduate student, becomes a disciple
  – Joy invents vi, adds demand paging to Unix - 3BSD
  – A DARPA research grant supports the implementation of TCP/IP and the Unix interface, sockets
  – Berkeley Software Distribution (BSD Unix) released
  – Most influential BSD versions: 4.2 (1983) and 4.3 (1987)
  – BSD Unix will form the basis of many commercial Unix spinoffs, such as SunOS and DEC Ultrix
Back at the Phone Company…

• 1987 - UNIX System V Release 3
  – Major features include: IPC facilities, Transport Level Interface, Remote File Sharing, STREAMS communication facility
  – Various major hardware vendors, who felt business pressure to have a “Unix” product, based their OS on this version. Examples: HP-UX IBM AIX.

• Late 1980s - AT&T and Sun Microsystems agree to cooperate on UNIX development to merge/unify System V and BSD.

GNU (GNU’s Not Unix!)

• 1983 - In response to commercialization of Unix, Richard Stallman starts the GNU Project
  – Goal: create “complete Unix-compatible software system” composed entirely of free software
  – 1985 - Stallman creates Free Software Foundation
  – 1989 - Stallman writes GNU General Public License
    • Copyleft license: requires derived works to be available under the same copyleft.
  – By early 1990’s, many programs (libraries, shells, etc) available, but kernel work was stalled.
Linux

• Andrew Tanenbaum had developed MINIX as a “microkernel-based, Unix-like” OS for educational purposes.

• 1991 - Linus Torvalds develops Linux kernel as a student project
  – Based in part on MINIX and Unix System V
  – Wanted a free Unix-like, industrial strength OS (note: google(Tanenbaum-Torvalds debate) on monolithic vs microkernels)
  – Originally developed only for Intel x86 hardware; since then, has been ported to more platforms than any other OS

Linux Impact

• Linux made available under GNU Public License
• Many, many distributions of Linux, packaged with various configurations, have been available over the past 20 years
• Torvalds continues to oversee kernel work, and Stallman still runs Free Software Foundation
• Linux continues to evolve:
  – SMP kernel, demand-loadable modules
  – Support for 64-bit processors
  – Ever-expanding set of modules, new file systems
  – And many others…
Linux Impact

• Although desktop OS market is dominated by Windows versions, Linux has major share of other areas:
  – Top 10 fastest supercomputers run Linux
  – 95% of top 500 supercomputers run Linux
  – 2nd quarter 2013: 79% of smartphones sold in world used Android
  – 33% of server market (65% for Unix overall)

Mach Project (Carnegie Mellon, mid ‘80s)

• Some researchers considered a monolithic kernel to be unwieldy and inherently insecure/brittle.
• RIG project at U.Rochester, 1975
  – goal to develop a modular, message-based OS
  – Key designer, Rick Rashid, moved to CMU in 1979
• Accent Project (CMU, Rashid et al, early 80’s)
  – OS designed for PERQ, an early workstation
  – similar to RIG, but with 32-bit VM, network transparency, etc.
  – by 1984, running on 150 PERQs, but still no competition to Unix
Mach History

- Mach Project started in 1986
  - Features: microkernel, Unix compatibility, better IPC, SMP support, new VM system
  - supported by DARPA
  - OSF chose Mach 2.5 as basis for OSF/1, later used by NeXT
  - kernel grew quite large by including BSD code
  - Mach 3 took moved this code to user-level and went back to microkernel

Derivative XNU kernel used in NeXTSTEP OS, integrated Mach and BSD code

Windows NT Architecture

- Client-server design
- Different subsystems support applications designed for other operating systems
- NT kernel very similar to Mach microkernel; inherent thread and SMP support
- System resources represented as objects
- NT Executive handles most conventional OS work; runs in kernel mode
- Hardware abstraction layer (HAL): maps generic hardware interaction to specific platforms
Original Architecture

1996 Apple acquires NeXT!
- Access to Mach, Objective-C, and everything NeXTSTEP
- Jobs again CEO in 2000

New OS development
- Apple “Rhapsody” project
- Later renamed Mac OS X
- Core components (Mach, Cocoa API, IOKit, Xcode, etc) are all inherited from NeXT

Darwin core: Merging of Mach and BSD Unix, plus OO driver framework, in a single address space OS

Mac OS X
Original Mac OS X Architecture

- Lower entities support higher ones
- Entities may cross layers (e.g., Java support, etc.)
- XNU combines Mach/BSD
- XNU kernel looks like?

Key Players

- Fernando Corbato (1926 - )
  - MIT professor (retired)
  - Turing Award 1990

- Jack B. Dennis (1931 - )
  - MIT professor (retired), Member NAE

- Peter Denning (1942 - )
  - PhD, MIT
  - Prof at Purdue, then NASA, GMU, NPS...