Operating System

Introduction
(Ch 1)

Topics

• What is an OS?
• OS History
• OS Concepts
• OS Structures
Let’s Get Started!

• What are some OSes you know?
  – Guess if you are not sure
• Pick an OS you know:
  – What are some things you like about it?
  – What are some things you don’t like about it?

What is an Operating System?

<table>
<thead>
<tr>
<th>Bank Program</th>
<th>Reservation</th>
<th>Game</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compilers</td>
<td>Editors</td>
<td>Shell</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operating System</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Machine Language</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microprogramming</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Physical Devices</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Applications
System Programs
Hardware
What is an Operating System?

• An Extended Machine (Top-down)
  – Transforming - new resource
    + ex: Windows device manager
• A Resource Manager (Bottom-up)
  – Multiplexing - illusion of several resources
    + ex: browse the web AND read email
  – Scheduling - deciding who gets what when
    + ex: compile fast OR edit fast
• Why have an OS?
  – Convenient and Efficient
    + Programming hardware difficult
    + Idle hardware “wasteful”

Topics

• What is an OS? (done)
• OS History (next)
• OS Concepts
• OS Structures
OS History

• Helps understand key requirements
  – Not one brilliant design
    + (despite what Gates or Torvalds might say)
  – Fixed previous problems, added new ones
  – Tradeoffs

• Closely tied to:
  – Hardware history
  – User history

Hardware History

<table>
<thead>
<tr>
<th></th>
<th>1981</th>
<th>1999</th>
<th>Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Power</strong></td>
<td>1</td>
<td>250</td>
<td>250</td>
</tr>
<tr>
<td><strong>$/Power</strong></td>
<td>$100K</td>
<td>$45</td>
<td>2200</td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>128K</td>
<td>128M</td>
<td>1000</td>
</tr>
<tr>
<td><strong>Disk Capacity</strong></td>
<td>10M</td>
<td>10G</td>
<td>1000</td>
</tr>
<tr>
<td><strong>Net Bandwidth</strong></td>
<td>9600b/s</td>
<td>155Mb/s</td>
<td>15K</td>
</tr>
<tr>
<td><strong>Users / Mach.</strong></td>
<td>10s</td>
<td>&lt;=1</td>
<td>10</td>
</tr>
</tbody>
</table>

• Comments? Change!
OS History

- Supplement to book
- My version is a brief narrative

Hardware Very Expensive
Humans Cheap

- Single program execution (no OS)
- Hardwire “programming”
- Programming slow, not “offline”!
  - Punch cards
Hardware Very Expensive
Humans Cheap

- Punch cards
- Fortran or assembler
- Waste computer time walking!
  - Batch programs on tape

Hardware Very Expensive
Humans Cheap

- Programs read in from tape
- Two applications:
  - Scientific
  - Data processing
- CPU idle during I/O!
  - Multiprogramming with partitions
  - Spooling as jobs finished
Hardware is Cheap
Humans Expensive

• Turn around time 1/2 day
• Programmer time wasted!
  “Sigh. In the good old days….”
  – Time-sharing
  – Multics (sorta)
  – New problems
    + response time
    + thrashing
    + file-systems

Hardware Very Cheap
Humans Very Expensive

• Personal computers
  – Network operating systems
  – Distributed operating systems
• OSes today
  – size
    + small == 1 million
    + large == 10 million
  – need to evolve quickly
    + hardware upgrades, new user services, bug fixes
  – efficient and/or modular kernels
Windows XP History

- 1988, v1
  - split from joint work with IBM OS/2
  - Win32 API
- 1990, v3.1
  - Server and Workstation versions
- 1997(?), v4
  - Win95 interface
  - Graphics to kernel
  - More NT licenses sold than all Unix combined

Windows 2000 History

- 2000 v5, called “Windows 2000”
  - Micro-kernel
  - Multi-user (with terminal services)
- Four versions (all use same core code)
  - Professional
    + desktop
  - Server and Advanced Server
    + Client-server application servers
  - Datacenter Server
    + Up to 32 processors, 64 GB RAM
Windows XP History

- Released in 2001
- Replacement to 95/98
- Same code-base as Windows 2000
- Client version and Server version
  - Server version called .NET
- Simultaneous users via Windows Terminal Services
- 64-bit version

Windows Today

- Microsoft has 80% to 90% of OS market
  - mostly PC’s
- 800 MHz Intel Pentium
- Aiming at robust, server market
  - network, web and database
- Platforms
  - Intel 386+ only (IA32 or IA64)
- WinNT is 12 million lines of code
- Win2000 is 18 million lines of code
- WinXP is 20 million lines of code
**Linux History**

- **Open Source**
  - Release Early, Release Often, Delegate
  - “The Cathedral or the Baazar”
- **Bday 1991, Linus Torvalds, 80386 processor**
  - v.01, limited devices, no networking,
  - with proper Unix process support!
- **1994, v1.0**
  - networking (Internet)
  - enhanced file system (over Minix)
  - many devices, dynamic kernel modules

<table>
<thead>
<tr>
<th>Year</th>
<th>Version</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>1991</td>
<td>v0.1</td>
<td>limited devices, no networking, with proper Unix process support!</td>
</tr>
<tr>
<td>1994</td>
<td>v1.0</td>
<td>networking (Internet), enhanced file system (over Minix), many devices, dynamic kernel modules</td>
</tr>
<tr>
<td>1995</td>
<td>v1.2</td>
<td>more hardware, 8086 mode (DOS emulation) included, Sparc, Alpha, MIPS support started</td>
</tr>
<tr>
<td>1996</td>
<td>v2.0</td>
<td>multiple architectures, multiple processors, threads, memory management ….</td>
</tr>
</tbody>
</table>

**Development convention**

- Odd numbered minor versions “development”
- Even numbered minor versions “stable”

- **1995, v1.2**
  - more hardware
  - 8086 mode (DOS emulation) included
  - Sparc, Alpha, MIPS support started
- **1996, v2.0**
  - multiple architectures, multiple processors
  - threads, memory management …. 
Linux Today

- v2.6
- 3 million lines of code
- 29 million users
- Growing by 25%/year through 2003
  - all others, 10% combined

Outline

- Operating System Concepts
  - Processes
  - Memory management
  - Input/Output
  - Files
  - System Calls
  - Shells
- Operating System Structures
The Process

- Program in execution
- Running -> Suspended -> Running
- Example: the Shell
- Process “Tree”
- Signals
- UID (GID)
- (4 hours)

Memory Management

- One chunk of physical memory
- Needs to be shared with all processes
  - multiprocessing
- 32 bit architecture, $2^{32}$ bytes $\Rightarrow$ 4GB!
  - virtual memory
- (8 hours)
Input/Output

- OS manage resources, including other devices
- Significant fraction of code
  - Up to 90% (complete distribution source)
- Want to be simple to use
- (2 hours)

Files

- Store data on disk
- Directory “Tree”
- Working directory
- Protection bits
  - 9 in Unix: **rw* bits**, ex: `rw-x-x-x-x`  
- Abstraction of I/O device
  - terminal, printer, network, modem
- Pipe
- (4 hours)
System Calls

• Way processes communicate with OS
• example:
  \[\text{write(file, string, size)}\]
• OS specific!
• POSIX (1980s)
  – Portable Operating System (UNIX-ish)
• (Most of the projects use them)

Shells

• User’s interface to OS
• Simple commands
  “cd”, “cat”, “top”
• Modifiers
  ‘&’, ‘|’, ‘>’
• (Hey, do some process and shell examples!)
Outline

• Operating System Structure
  – Simple Systems
  – Virtual Machines
  – Micro Kernels

Simple Systems

• Started small and grew, no hardware support
• MS-DOS

- Application
  - Resident system program
  - Device drivers
  - ROM BIOS device drivers

• Protection!
Simple Systems

- Unix (see /vmunix)
- “The Big Mess”
- Some move towards a more modular kernel

Applications

Signals, File Sys, Swapping, Scheduling ...

Terminal    Device    Memory

Virtual Machines

- IBM VM/370 → VMWare

Process | Process | Process
--------|---------|---------
Process | Process | Process
Operating Sys | Operating Sys | Operating Sys
Virtual Machine
Hardware

- Complete protection
- OS development, emulation
- Performance!
- (Exokernel says can have subset of kernel)
Virtual Machines

• Java Virtual Machine

<table>
<thead>
<tr>
<th>Java program</th>
<th>Java OS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java VM</td>
<td>Process</td>
</tr>
<tr>
<td>Operating System</td>
<td>Process</td>
</tr>
<tr>
<td>Hardware</td>
<td></td>
</tr>
</tbody>
</table>

• Platform independence!

Micro Kernel

• Mach

<table>
<thead>
<tr>
<th>User Process</th>
<th>File Server</th>
<th>Mem Server</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Kernel</td>
</tr>
</tbody>
</table>

• Client-Server
• Good performance
• Adaptable to distributed OS
• Robust
• Careful about mechanism!
WinXP Structure

User Level Space
- Netscape
- Win32 Subsystem
- Security

Executive / Privileged Space
- I/O
- File System

Kernel Space
- Graphics
- Scheduler
- Memory Manager
- IPC
- Window Manager

“Micro Kernel?”
- Hardware Abstraction Layer

Linux Structure

- “Simple” system
  - Applications, User Space
  - System Libraries
  - Kernel
  - Terminal

- Loadable Modules
  - done after “boot”
  - allow 3rd party vendors
  - easier for development