Lecture 1

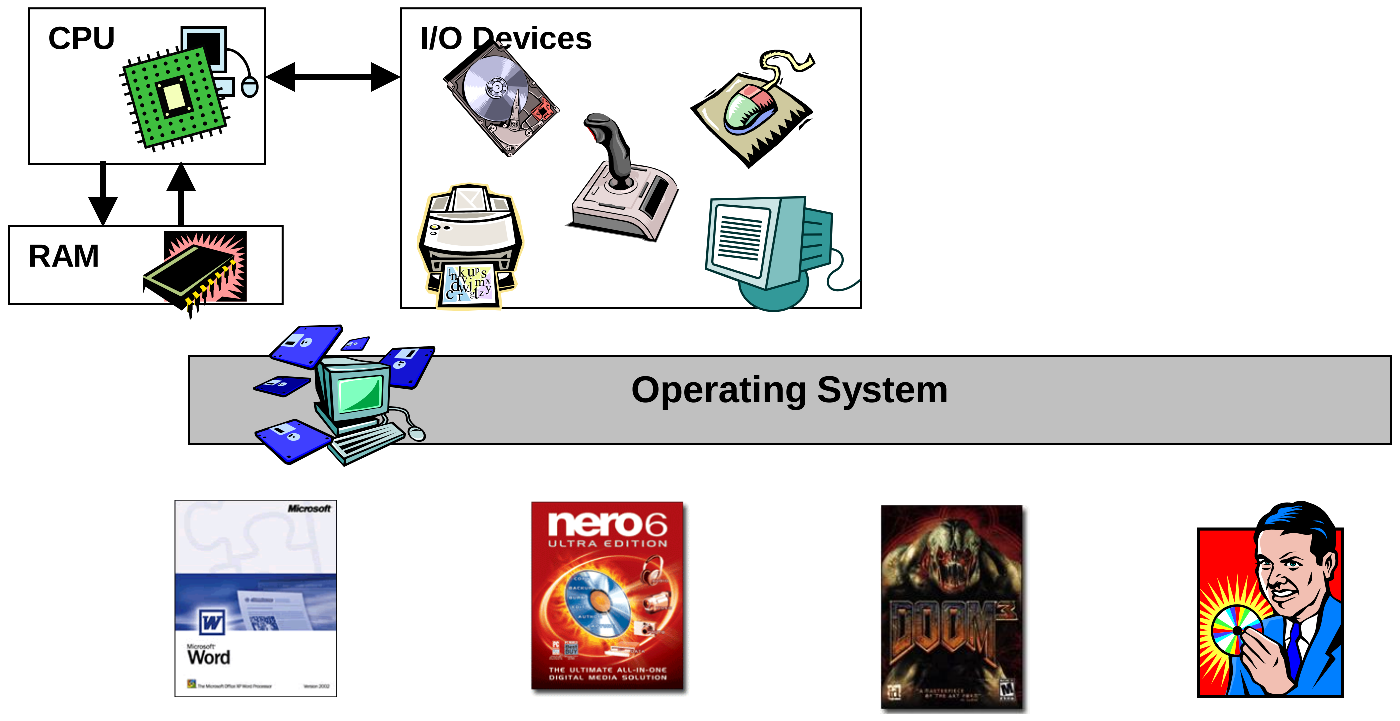
Introduction & Getting Started

COP 3353 Introduction to UNIX

Brief History

- Dennis Ritchie and Ken Thompson of Bell Laboratories developed the Unix operating system in the early 1970's
  - Unix is a "pun" on Multics. Multics was a joint project of many companies and universities designed to be a leap forward in OSs. Multics contributed many ideas to OS development but failed as a useful OS.
  - Thompson needed to build an OS for a PDP-7 (9 Kbytes of main memory) and did so with the help of Ritchie (who also developed the C language with Brian Kernighan). This became Unics, and then Unix.

Basic System Components & OS



Basic Components

- CPU (Central Processing Unit, "Processor")
  - Brain
- Main Memory (RAM)
  - Temporary Workspace
- I/O (Input/Output)
  - Keyboard, Mouse
  - Monitor
  - Mass Storage (Hard Drives, CD-ROM)
- Operating System
  - Oversees interaction of hardware components
  - Provides interface between software and hardware
  - Provides interface to user
  - Most common use is running programs and managing "files"

Major Components of the Unix OS

- Kernel
  - The master control program
  - Schedules tasks and switching to provide multitasking and multi-user operation
  - Manages resources
- Shell
  - Interprets user commands
  - Passes user commands to the kernel for execution (executes programs)
- File System
  - Information organized as files and specialized files called directories
- Utilities
  - Software tools provided as part of the OS. Often called commands

Some Definitions

- Executable
  - A program in a form that can be executed by the OS
- Process
  - The activation or instantiation of an executable
- Daemons
  - Processes spawned by the kernel (OS) to perform tasks on behalf of OS to manage system resource
- Filters
  - General purpose utilities transforming an input stream to an output stream while doing well-defined processing

Varieties of Unix

- Developed at Bell Labs and AT&T
- University of California Berkeley
  - BSD Unix
- Commercial versions
  - SunOS, Solaris, SCO Unix, Aix, HP/UX, Ultrix
- Freely available version
  - GNU (Gnu's not Unix) & Free Software Foundation
  - Linux (Linus Torvalds created for PCs), NetBSD, FreeBSD
- Linux Distributions (Linux kernel core + parts of Gnu etc.)
  - Fedora Core (Red Hat), SUSE Linux (Novell), Ubuntu, Mandriva, Gentoo, Debian
- Posix – a standard
  - A standard for Unix like operating systems

Logging on to a CS Machine

- Machines
  - Diablo (“diablo.cs.fsu.edu”) - Faculty only (do not use)
  - Shell (“shell.cs.fsu.edu”) - Use this one generally (Linux OS)
  - Linprog (“linprog.cs.fsu.edu”) - Use for programming (actually a stack of “linprog1” – “linprog4”, Linux OS)
  - Program (“program.cs.fsu.edu”) –Also for programming (“program1” – “program4”, Solaris OS)
- SSH (Secure Shell)
  - Use an SSH client program to connect to CS machines
  - Info from CS Systems Group on accessing CS servers, including Tectia download
    - https://system.cs.fsu.edu/newuser/ssh-how-to/
- New Account Application
  - http://system.cs.fsu.edu/info/newuser/index.html
  - Use SSH Client to connect to "shell.cs.fsu.edu"
  - username: **newacct**
  - password: **newacct**
  - Carefully follow *rules* for creating your password.
  - Remember to record / remember your username and remember your password

Variety of Shells

- Some aspects
  - Prompt ($, %, >, machine you are on, etc)
  - History mechanism (arrow keys), string completion (tab)
- Different shells
  - sh: Bourne shell, (S.R. Bourne, good scripting capabilities)
  - csh: C shell, (UC Berkeley, closer to C syntax)
  - ksh: Korn shell, (David Korn, better interactivity)
  - bash: Bourne-again shell (built on sh with more features)
  - tcsh: T shell: (Tenex shell) similar to C shell, default on Linux /Intel installations, default on CS accounts

Editors

- Common text editors that are available (none have many of the features available on word processors) for plain text files such as programs, shell scripts, etc.
  - vi (vee-eye)
    - Available on almost all Unix machines
    - Fairly powerful and sophisticated
  - emacs (ee-macs)
    - Also widely available
    - Powerful and popular
  - nano (updated version of pico)
    - Easier to learn but simpler and not as powerful

Starting pico

- The command “nano" at a shell prompt will start the “nano" text editor with an empty buffer

```
$ nano
```

- Specifying a file name will have "nano" open that file (or start a new file)

```
$ nano testfile1
```

- Basic Command
  - Arrow keys are used to navigate around the document
  - Typing will insert text at the point of the cursor
  - The caret sysmbol (^) indicates you must press and hold the control (ctrl) key first, then press the command key
  - Some available commands are at the bottom of the nano window
  - ^o writes “out” the text to a file (a prompt will let you specify the name)
  - ^x exits nano

Marking and cutting and pasting in nano

- You cannot use your mouse in "nano" (actually, the mouse works to cut and paste because of the SSHClient program, but you must learn how to work without it)
- ^^ (ctrl-*shift*-^) begins marking text at the current cursor position
- Use the arrow keys to mark text
- ^k cuts text (kills),
- ^u then brings the text back at the current cursor position

nano command summary

`(arrows)`	Move cursor
`(bksp)`	Move cursor left one space, deleting character
`^a`	Move to beginning of line
`^b`	Move back one character (same as left arrow)
`^e`	Move to end of line
`^f`	Move forward one character (same as right arrow)
`^n`	Move to next line (same as down arrow)
`^p`	Move to previous line (same as up arrow)
`^v`	Move forward one page
`^y`	Move back one page
`^(space)`	Move to next word

nano command summary continued

`^c`	Shows current position
`^d`	Delete character at current position
`^g`	Display help file (^V and ^Y to scroll through)
`^h`	Delete previous character (same as bksp)
`^i`	Insert TAB character (same as tab)
`^j`	Justify paragraph
`^^`	Begin selecting text at current cursor position
`^k`	Cut selected text
`^l`	Redraw screen
`^o`	Output current buffer to a file (save)
`^r`	Insert text from a file
`^u`	Undelete last line, series of lines, or marked block you deleted. Can also "unjustify"
`^w`	Search file for text
`^x`	Exit nano