Introduction to UNIX

CS 2204
Class meeting 1

*Notes by Doug Bowman and other members of the CS faculty at Virginia Tech. Copyright 2001-2003.

What is UNIX?

- A modern computer operating system
- Operating system:
  - "a program that acts as an intermediary between a user of the computer and the computer hardware"
  - Software that manages your computer’s resources (files, programs, disks, network, …)
  - e.g. Windows, MacOS
- Modern: features for stability, flexibility, multiple users and programs, configurability, etc.

Why UNIX?

- Used in many scientific and industrial settings
- Huge number of free and well-written software programs
- Open-source OS
- Internet servers and services run on UNIX
- Largely hardware-independent
- Based on standards

Brief history of UNIX

- Ken Thompson & Dennis Ritchie originally developed the earliest versions of UNIX at Bell Labs for internal use in 1970s
  - Borrowed best ideas from other Oss
  - Meant for programmers and computer experts
  - Meant to run on “mini computers”

Early UNIX History

- Thompson also rewrote the operating system in high level language of his own design which he called B.
  - The B language lacked many features and Ritchie decided to design a successor to B which he called C.
  - They then rewrote UNIX in the C programming language to aid in portability.

UNIX variants

- Two main threads of development:
  - Berkeley software distribution (BSD)
  - Unix System Laboratories System V
  - Sun: SunOS, Solaris
  - GNU: Linux (many flavors)
  - SGI: Irix
  - FreeBSD
  - Hewlett-Packard: HP-UX
  - Apple: OS X (Darwin)
  - …
Layers in the UNIX System

- **Hardware** (CPU, memory, disks, terminals, etc.)
- **UNIX Operating System** (process management, memory management, the file system, I/O, etc.)
- **Standard Library** (open, close, read, write, etc.)
- **Standard Utility Programs** (shell, editors, compilers, etc.)
- **Users**

System Interface calls

Library Interface

User Interface

User Mode

Kernel Mode

UNIX Structure

- The **kernel** is the core of the UNIX system, controlling the system hardware and performing various low-level functions. The other parts of the UNIX system, as well as user programs, call on the kernel to perform services for them.

UNIX Structure

- The **shell** is the command interpreter for the UNIX system. The shell accepts user commands and is responsible for seeing that they are carried out.

UNIX Structure

- Over two hundred **utility programs or tools** are supplied with the UNIX system. These utilities (or commands) support a variety of tasks such as copying files, editing text, performing calculations, and developing software.

- This course will only attempt to introduce a limited number of these utilities or tools!

Getting started

- You must have an "account" on the UNIX machine you're trying to use
- "login" is your user name (usually some variant of your real name)
- Your password will not echo as you type
- Remember good password practices
The shell prompt
- After logging in, you get some information about the system, then a shell prompt
- shell: The program you use to send commands to the UNIX system
- Prompts take many forms:
  - $  
  - %  
  - username@hostname>  
  - hostname # %  
- Must beat the "Tyranny of the blank screen"

Entering commands
- Anytime you see a prompt, you can enter a command for the shell
- Some commands are a single word
  - who  
  - date  
  - ls  
- Others use additional information
  - cat textfile  
  - ls -l

Command syntax
- Commands must be entered exactly. If you make a mistake before entering, delete/backspace to fix it. Be careful!
  - command options argument(s)
- Options: modify command's execution
- Arguments: often filenames that tell a command what to operate on

Example commands: ls (list)
- ls -l  
- ls -a -l  
- ls -al  
- ls -a; ls -l  
- ls textfile  
- ls folder  
- ls textfile1 textfile2  
- ls -al textfile

If you don’t get a normal shell prompt...
- A program is probably running
- If you see a special program prompt, try to quit the program (quit, bye, exit)
- If you see nothing, you can
  - Stop the program with CTRL-z (program will wait until started again)
  - Interrupt the program with CTRL-c (program will usually die)

Ending your session
- Always log out when you are done
- Use the command exit (also sometimes logout or CTRL-d)
- Note: if you are running a window system, logging out of the shell only ends that shell. You must also log out of the window system using a menu.