Introduction to Linux

Week 1 - Wednesday

A Brief History of Operating Systems

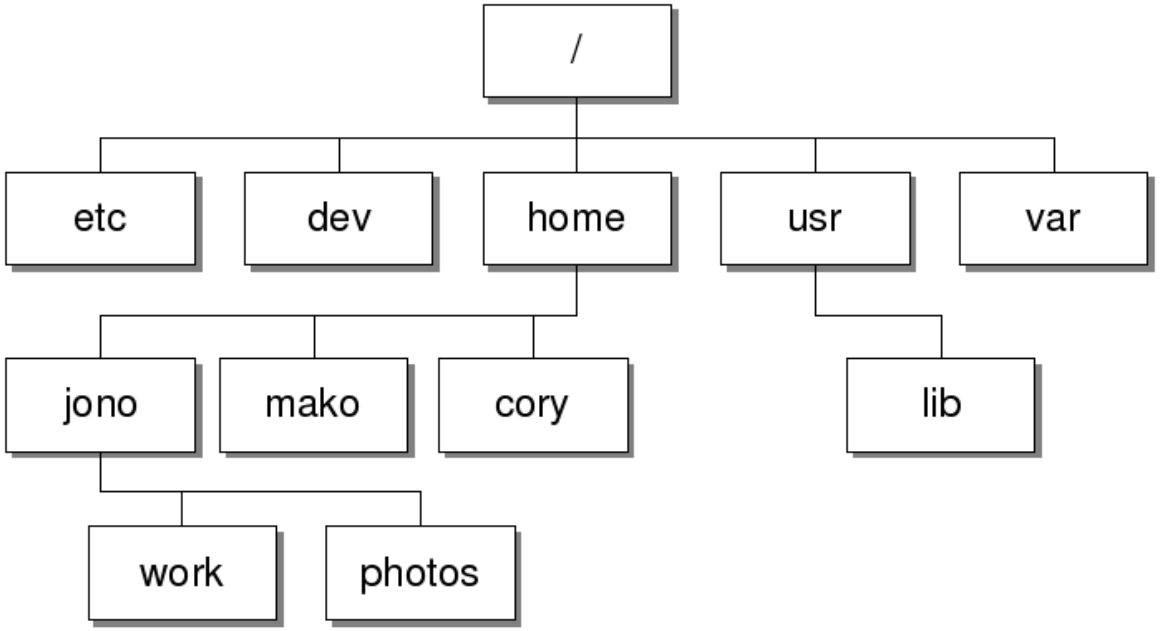
- The Dark Ages
  - No OS until 1960s
  - Manually loaded programs
  - Reboot after each program
- Batch OS
  - Unified application development across systems
  - Output via printer, later via monitor
  - I/O via magnetic tape or disk
  - Written in assembler (e.g., OS/360)
  - Multiprocess

A Brief History of Operating Systems

- Timesharing OS
  - Multiuser
  - Multics (1964)
    - Segmented memory
    - Paged virtual memory
    - Applications written in many languages
    - Shared multiprocess memory
- Personal Computer
  - Single machine for single user
  - OS must manage screen and input devices
  - Window, Icon, Menu, Pointing Device (WIMP, e.g., MacOS, 1984)
- Cutting-Edge OS
  - High performance computer (HPC) clusters
  - Cell phones, video
  - Video games
  - Browsers

Unix File System Layout

- Everything is a file (including devices)
- Tree structured hierarchy (with some exceptions)
- Lost?
  - man: get manual or man pages



Process: ps and kill

- Process
  - An instance of a computer program that is being executed
- ps
  - List processes that are currently running
- kill
  - Terminate certain process(es)
  - Usage
    - kill PID

CLI(Command Line Interface) vs. GUI(Graphic User Interface)

CLI

- Steep learning curve
- Pure control (e.g., scripting)
- Cumbersome multitasking
- Speed: Hack away at keys
- Convenient remote access

GUI

- Intuitive
- Limited Control
- Easy multitasking
- Limited by pointing
- Bulky remote access

The Basics: File Name Matching

- ?: matches any single character in a filename
- *: matches one or more characters in a filename
- []*:* matches any one of the characters between the brackets. Use ‘-’ to separate a range of consecutive characters.

The Basics: History

- <up arrow>: previous command
- <tab>: auto-complete
- !!: replace with previous command
- ![*str*]: refer to previous command with str
- ^[*str*]: replace with command referred to as str

The Basics: Redirection

- > *file*: write stdout to a file
- >> *file*: append stdout to a file
- < *file*: use contents of a file as stdin

The Basics: Changing File Attributes

- ln: create a link
  - Hard links: points to physical data
  - Soft links aka symbolic links (-s): points to a file
- touch: update access & modification time to current time
  - touch *filename*
  - touch -t 201101311759.30 *filename*
    - Change filename's access & modification time to (year 2011 January day 31 time 17:59:30)

The Basics: chmod

- chmod
  - read (r), write (w), executable (x)
  - User, group, others

Reference	Class	Description
u	user	the owner of the file
g	group	users who are members of the file's group
o	others	users who are not the owner of the file or members of the group
a	all	all three of the above, is the same as *ugo*

The Basics: chmod

Operator	Description
+	adds the specified modes to the specified classes
-	removes the specified modes from the specified classes
=	the modes specified are to be made the exact modes for the specified classes

Mode	Name	Description
r	read	**r**ead a file or list a directory's contents
w	write	**w**rite to a file or directory
x	execute	e**x**ecute a file or recurse a directory tree

The Basics: chmod

#	Permission
7	full
6	read and write
5	read and execute
4	read only
3	write and execute
2	write only
1	execute only
0	none

- Usage
  - chmod ["references"]["operator"]["modes"] "file1" ...
    - Ex) **chmod** ug+rw mydir, **chmod** a-w myfile,
    - Ex) **chmod** ug=rx mydir, **chmod** 664 myfile

The Basics: find

- -type: type of a file (e.g,, directory, symbolic link)
- -perm: permission of a file
- -name: name of a file
- -prune: don’t descend into a directory
- -ls: list current file

The Basics: find

- Examples
  - find . -name my*
  - find . -name my* -type f
  - find / -type f -name myfile –print
  - find / -path excluded_folder -prune -o -type f -name myfile –print

The Basics: locate

- Find files by name
- -b: Match only the base name against the specified patterns.

The Basics: whereis

- locate the binary, source, and manual page files for a command

Submission Rules - Newline

UNIX/Linux

- Only uses
  - LF (Line feed, '\n', 0x0A, 10 in decimal)

Windows

- Uses
  - CR followed by LF (CR+LF, '\r\n', 0x0D0A).
  - CR (Carriage return, '\r', 0x0D, 13 in decimal)
  - LF (Line feed, '\n', 0x0A, 10 in decimal)

Homework Submission Rule: The .txt files should be ASCII text files, with no carriage returns, and with no more than 80 columns per line.

→ **Don't do your homework on Windows.**

Submission Rules – Column number check

- vi editor – edit mode

  – Number inside the box is the column number

Submission Rules – Column number check

- Emacs
  - M-x column-number-mode or
  - Options → Show/Hide → Check “Column Numbers”
  - The number inside the box is the Column Number.