Software Construction Laboratory 35L

Mushi Zhou

Winter 2017

Lab 3

Accessing the Servers

- Use Putty on Lab Machines:
  - Host Name: lnxsrv07.seas.ucla.edu
  - Port: 22
  - Your SEASNet Account
- On UCLA Network
  - For Windows: The same as lab machines
  - For Mac/Linux: ssh *your_SEASNetAcc@lnxsrv07.seas.ucla.edu*
- Remotely
  - Use Cisco AnyConnect and UCLA VPN with your MyUCLA account to get on UCLA Network. (ssl.vpn.ucla.edu)

Account Set Up

- Homework Graded on Server: lnxsrv07.seas.ucla.edu
- Servers 06 or 09 works the same

- PATH=/usr/local/cs/bin:$PATH (every login)

- Add PATH=/usr/local/cs/bin:$PATH to ~/.profile (once for all)

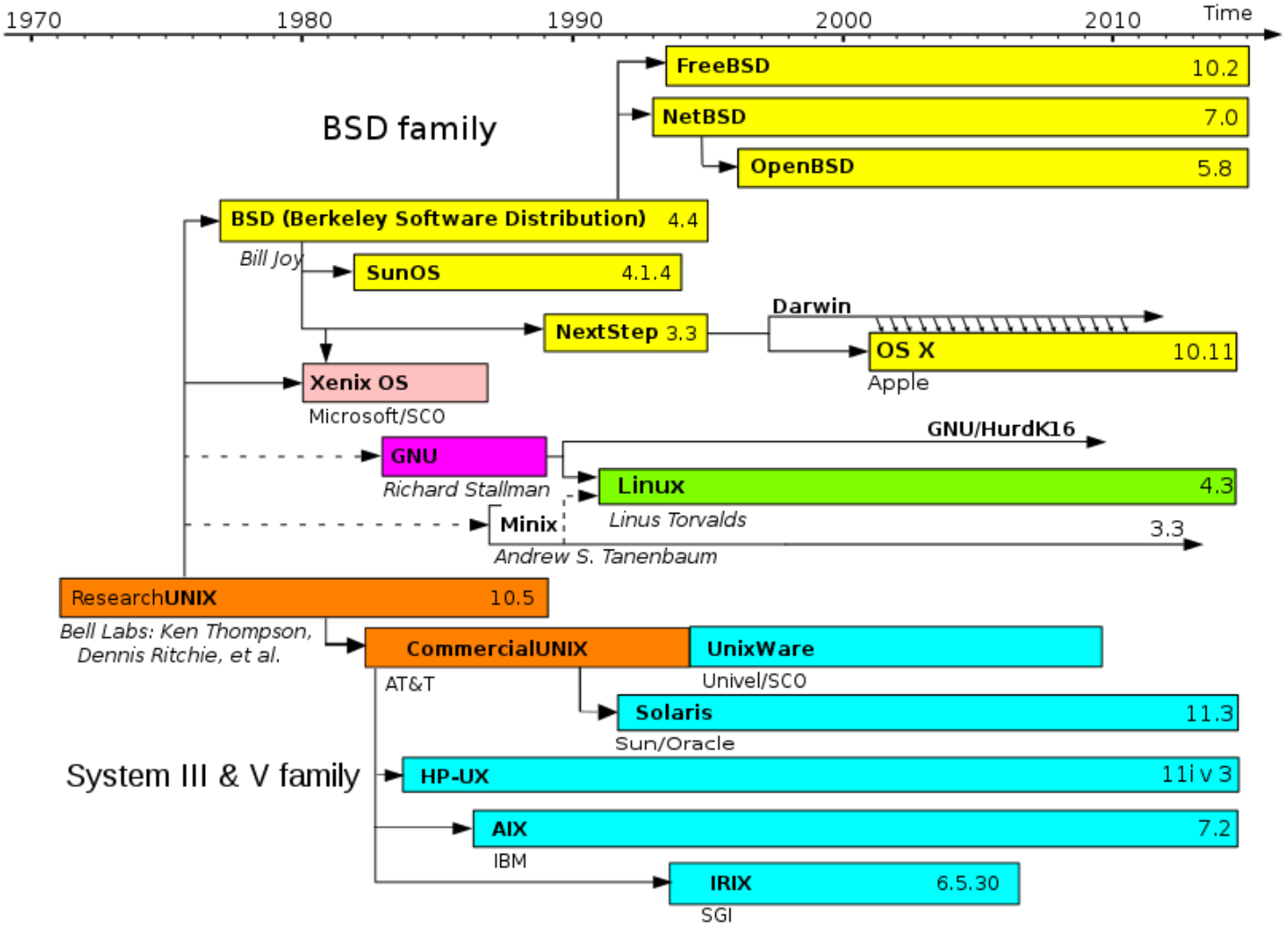
- Use “echo $PATH” to test

Lecture 1
Introduction to Linux & Emacs

Unix-Like system

- Developed since 1970s
- Bell Labs research center by Ken Thompson, Dennis Ritchie
- Multiuser
- Multitask/Multi-process
- Conform to a standard behavior of Unix command or shell
- Examples: Apple OS X, Linux

Unix-Like System Family Timeline



Linux

- Unix Like
- Assembled under the model of free and open-source software development and distribution
- Linux kernel
- The largest installed base of all general-purpose operating systems
- Linux is also the leading operating system on servers
- Runs on embedded systems
- Linux is packaged in a form known as a *Linux distribution*

Ubuntu & Red Hat

Ubuntu

- Debain based,
- Mostly for personal computers, smartphone platforms, and some servers
- Mostly non-commercial purposes
- Most popular operating systems running in hosted environments, so–called "clouds", as it is the most popular server Linux distribution.
- People are encouraged to use free software, study how it works, improve upon it, and distribute it
- 16.04.1 LTS released 21 July 2016

Ubuntu & Red Hat

Red Hat

- Red Hat Enterprise Linux
- Targeted towards the commercial market
- Third-party derivatives can be built and redistributed by stripping away non-free components of the distribution.

GUI vs CLI

GUI: Graphic User Interface

- Introduced in reaction to the perceived steep learning curve of command-line interfaces (CLI)
- Allows the use of icons or other visual indicators to interact with electronic devices

Examples of GUI:

- Microsoft windows systems
- X, Wayland, GNOME, KDE (You can read more on these from the links on course website syllabus)

GUI vs CLI

CLI: command-line interfaces

- A way to interact with a computer by typing text commands into a terminal window
- Usually, Unix-like systems have similar CLI for users.
- All Linux systems have the same CLI for users.

Examples of CLI:

- MS-DOS system
- Bash (We will focus on this)
- Xterm (You can read more on this from the link on course website syllabus)

Bash

Shell

- A shell is a user interface for access to an operating system's services
- It is named a shell because it is a layer around the operating system kernel
- It can either be GUI or CLI

Bash

- A Linux shell that typically runs in a text window (CLI), where the user types commands that cause system actions.
- Bash can also read commands from a file, called a script. (This is what assignment 2 is about)

Unix File System Layout

- The file system is considered a central component of the operating system

- Provides information storage and retrieval, and one of several forms of interprocess communication.

Everything is a file

In Linux, everything is a file

- Documents are files
- Logs are files
- Directories (folders) are files
- Device drivers are files
- Programs are files

They are just in different formats, and you can open any of them using a text editor.

Important Directories

Below are important directories for Linux systems:

/ The slash / character alone denotes the root of the file system tree.
/bin Stands for *binaries* and contains certain fundamental utilities
/boot Contains all the files needed for successful booting process
/dev Stands for devices.
/etc Contains system-wide configuration files and system databases
/home Contains user home directories on Linux and some other systems
/lib Essential libraries for the system
/opt Contains locally installed software
/usr The user file system (User's files)
/var Stands for variable
/log Contains system log files.

How to Use Bash to Interact with the System

- Bash predefines a list of commands.
- The list is very long, we will go over some basic common ones
- For each command, there is a manual page that describes everything about that command
- Type: man <command_name>, the manual will show up in the terminal.
- So “man” is a command for showing the manual page for a given command (It is very useful for learning new commands)

Basic Commands: Moving Around

- pwd print working directory
- cd change working directory
- ~ home directory
- . current directory
- / root directory, or directory separator
- .. parent directory

Basic Commands

- cat
- head
- tail
- cp
- diff
- cmp
- wc
- sort
- find

Example of Command Options

Use command “find” as an example:

- find command will find all files in the current and sub-directories or in a given root directory
- Some options for find command (start with “-” following the command)
- -type: type of a file (e.g,, directory, symbolic link)
- -perm: permission of a file
- -name: name of a file
- -ls: list all files including their properties

For example:

find –type d

will list all sub-directories in the current directory

Executing a Program

- If a program is executable, it can usually be run using the command:
- ./program_name
- The output of the program will show in the terminal

Basic Commands: Dealing with Files

- mv: move a file (no undos!)
- cp: copy a file
- rm: remove a file
- mkdir: make a directory
- rmdir: remove a directory
- ls: list contents of a directory
  - -d: list only directories
  - -a: list all files including hidden ones
  - -l: show long listing including permission info

Basic Commands: 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.

Unix Permissions

- **Owner permissions** − The owner's permissions determine what actions the owner of the file can perform on the file.
- **Group permissions** − The group's permissions determine what actions a user, who is a member of the group that a file belongs to, can perform on the file.
- **Others permissions** − The permissions for others indicate what action all other users can perform on the file

Unix Permissions

- ls -l command displays various information related to file permission
- 10 digits
- First digit is type (- for regular file, d for directory)
- Read (R) write (W) execute (X) for each kind
- Chmod u/g/o/a =/-/+ (To change permissions)
- Chown [user] file (To change owner)
- Set permission with numbers

Unix Permissions

- The SUID and SGID bits will appear as the letter "s"
- Located at the permission bit where the owner/group execute permission bit would normally reside.
- A capital letter S in the execute position instead of a lowercase s indicates that the execute bit is not set.
- Executes as if run by the owner of the file/the group of the file belongs to with corresponding permissions (as opposed to the user who runs the file/ the group the user who runs the file belongs to)
- Set SUID/SGID with numbers (2/4/6)

$ ls -l /usr/bin/passwd

-r-sr-xr-x 1 root bin 19031 Feb 7 13:47 /usr/bin/passwd*

Vi Editor

VIM

- Normal: Enter commands

– Insert: Insert text

– ESC

– :qw

– !

Emacs

- Much more powerful compares to VIM
- Can do almost everything without using mouse
- show multiple files at once
- A little bit hard to start
- Available to almost all systems including embedded systems, just like vim

Intro to Emacs

- Three keys: Ctrl, Alt, ESC

For example:

- C-c = Hold Ctrl and enter the c
- M-c = Hold Alt and enter c

  or Press ESC once, then enter c
- ESC C-c = Press ESC once, Hold Ctrl, then enter c
- Combination of commands
- Sequential commands

Intro to Emacs

- **emacs** *filename*
- **CTRL-x CTRL-c** exit
- **CTRL-x CTRL-s** save
- **CTRL-a** go to beginning of the line
- **CTRL-e** go to end of the line
- **ESC <** go to the beginning of the file
- **ESC >** go to the end of the file
- **CTRL-y** re-insert ('yank') the last text that was killed
- **ESC w** copy the marked region

Intro to Emacs

- **CTRL-h a *FUNCTION*** lists commands related to ***FUNCTION***
- **CTRL-x 1** get rid of Help window
- **ESC CTRL-v** scroll Help window forward
- **CTRL-g** cancel partially typed or accidental command
- **CTRL-x u** undo the last change
- **ESC-x list-matching-lines** lists all the lines matching your pattern in a separate buffer, along with their numbers.
- Use **ESC-x goto-line** to go to the occurrence you're interested in.
- **ESC v** scroll to previous screen

Helpful Emacs Resources

Emacs Basics

http://mally.stanford.edu/~sr/computing/emacs.html

- Colorado State University CS Department

https://www.cs.colostate.edu/helpdocs/emacs.html

Brief Overview of Assignment 1

- Follow all instructions on the assignment page, except key stroke recording.
- Record ‘emacs SP exer1.html Enter” instead of “e m a c s SP e x e r 1 . h t m l Enter” for readability.
- key1.txt is for HW part (Everything)
- ans1.txt is for lab part (Everything)

Assignment Specifics

- What you want to submit in your .txt files:

For each lab part:

1. Your answers to the question
2. The command you used to get the answer
3. Very brief description of how you get your answers (If applicable)

For each HW part:

1. You answers to the question (if applicable)
2. Key strokes you used to achieve the goal
3. Brief explanations of what you did and the reasons for it (if applicable)

Some Things to Be Aware of

- 1.5.2 Since there is not a Makefile (We will talk about this later in the course), you need to use “gcc filename” to compile
- To run the compiled program and save the output to an Emacs buffer is kind of complicated, you can just run your program in command line.

Some Things to Be Aware of

- For 1.6.1 C-x b is the command to open an Emacs buffer
- You need to type parentheses in 1.6 questions
- For 1.6.7
- I'm not asking you to calculate the exact probability, as it is very difficult.
- You do need to state the probability likehood (i.e. very close to 1, very close to 0, between 0.2-0.3, etc)
- You do need to give very good reasons for your answers.

Some Things to Be Aware of

- *awk '/\r/ || 80 < length' key1.txt ans1.txt*
- This command is to make sure the lines in your file are less than 80 characters each.
- Carriage Return -> Enter/Return (\r) on a windows systems. (same as “\r\n”)
- Unix use only “\n” (new line)
- Be very careful when saving files in windows and transfer them into Unix systems.

For Week 2

- Next Monday is a holiday.
- We will have to cover the entire second week in one class.
- Unfortunately, assignment 2 seems to be the hardest in respect to the stage into the course compared to other assignments.
- I recommend you start working on assignment 2 before next Wednesday class.