CS35L Software Construction Laboratory

Lab 5 – Sneha Shankar

Week 1; Lecture 1

About this course

- Course Syllabus : https://web.cs.ucla.edu/classes/winter18/cs35L/syllabus.html
- Why this course?
  - To get accustomed to the most commonly used software environments and tools to be used in upper division CS classes (especially CS111)
  - Linux, scripting, VMs, version control management, systems programming, low-level construction, parallelism, etc.

Course Logistics

- **3 credit course**
- Structure: 9 assignments (Lab + HW), 1 report , 1 Final exam
- PTEs
- SEASnet account mandatory!
- Use piazza for questions
- Office Hours:
  - Prof: Mondays and Thursdays 14:15–15:15. Engineering VI 363
  - Sneha: Wednesdays 9:30 – 11:30 am; 2432 BH

Course Logistics contd…

- Grading : 50% HW and 50% Final exam
- Lateness penalty: N days late -> 2^N % of assignment deducted
- All assignments due by 23:55 of the specified date
- Assignment 1 due on 13th Jan 23:55
- Instructions for 3760 BH
  - Do not carry food or liquid inside.
  - Always logout if you use the computers in lab
- **My email id : snehashankar@cs.ucla.edu**

About SEAS account and its connection

- Create account from https://www.seas.ucla.edu/acctapp/
- Connect to lxnsrv server
- Install PuTTY SSH client (highly recommended) : Follow instructions on http://www.seasnet.ucla.edu/lnxsrv/
- Remember your SEAS username and password! (will mostly be different from your UCLA login and password)
- Use host lnxsrv.seas.ucla.edu and port 22 in PuTTY

What is an Operating System?

- Most important software that runs on a computer
- Manages memory, processes, other softwares and hardwares
- Makes human to computer communication easy
- Computer is useless without an OS!
- Brief history of Operating Systems: http://www.informit.com/articles/article.aspx?p=24972

Multiuser and Multi-process Operating System

- Allow many users to work on the same computer at the same time (as long as they have their own terminal)
- Allows many processes, programs and applications to run simultaneously.
- Supports multiprocessing and multitasking
- Multitasking OS examples
  - Windows
  - Linux
  - Unix

Open Source Software

- What is an open source software?
  - Source code is publicly available
  - Modification by any individual allowed on a global scale
  - It is free for use
- Examples: Firefox, Android, Linux

User Interfaces: CLI v/s GUI

CLI

- Steep learning curve
- Pure control (e.g., scripting)
- Speed: Only keyboard, faster performance
- Consumes less resources
- Remote access possible but cumbersome
- No change; less diverse

GUI

- Intuitive
- Limited Control
- Mouse + keyboard; Slower
- More resources; e.g. loading icons, fonts, etc.
- Easy remote access
- Frequent changes; More diverse

Debian GNU/Linux

- Clone of UNIX
- Linux is just a kernel.
- What is a kernel?
  - Core of any OS
  - Allocates time and memory to programs
  - Interfaces applications with the physical hardware
  - Allows communication between different processes: inter-process communication (IPC)
- Linux distribution make the Linux kernel a completely usable OS by adding various applications
- Linux distribution = GUI + GNU utilities (cp,mv,ls,etc) + installation and management tools + GNU compilers (c/c++) + Editors(vi/emacs) + ….
- Shell : Interface between the user and kernel

Basics of Shell

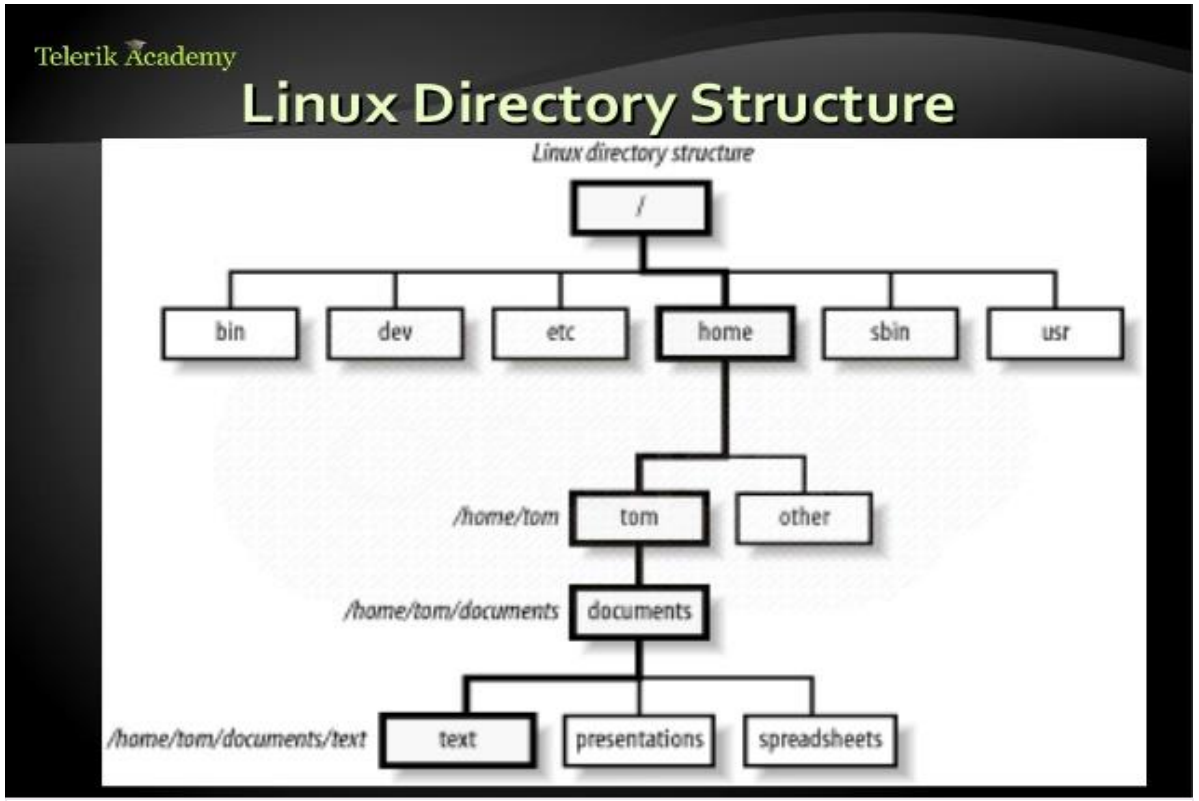
- Outermost layer around the kernel; hence called shell !
- Can be used as CLI as well as GUI depending upon the task/operation
- Examples:
  - CLI shell in Windows :
    - Command Prompt
  - CLI shell in UNIX :
    - Bash
- Basic shell commands:
  - **<up arrow>**: previous command
  - **<tab>**: auto-complete
  - **!!**: replace with previous command
  - **![*str*]**: refer to previous command with str
  - **^[*str*]**: replace with command referred to as str

Files and Processes

- Everything is either a process or a file
- **Process**: an executing program identified by PID
- **File**: collection of data
  - A document
  - Text of program written in high-level language
  - Executable
  - Directory
  - Devices

Linux File System Layout

- Tree Structure Hierarchy



Source: https://www.slideshare.net/azilian/4-linux-file-systems-18175783

- Only One Root- '/'
- Directories are also files
  - E.g. home, tom
- Regular files can only be leaves
  - E.g. text, spreadsheets, etc

The Basics: Moving Around

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

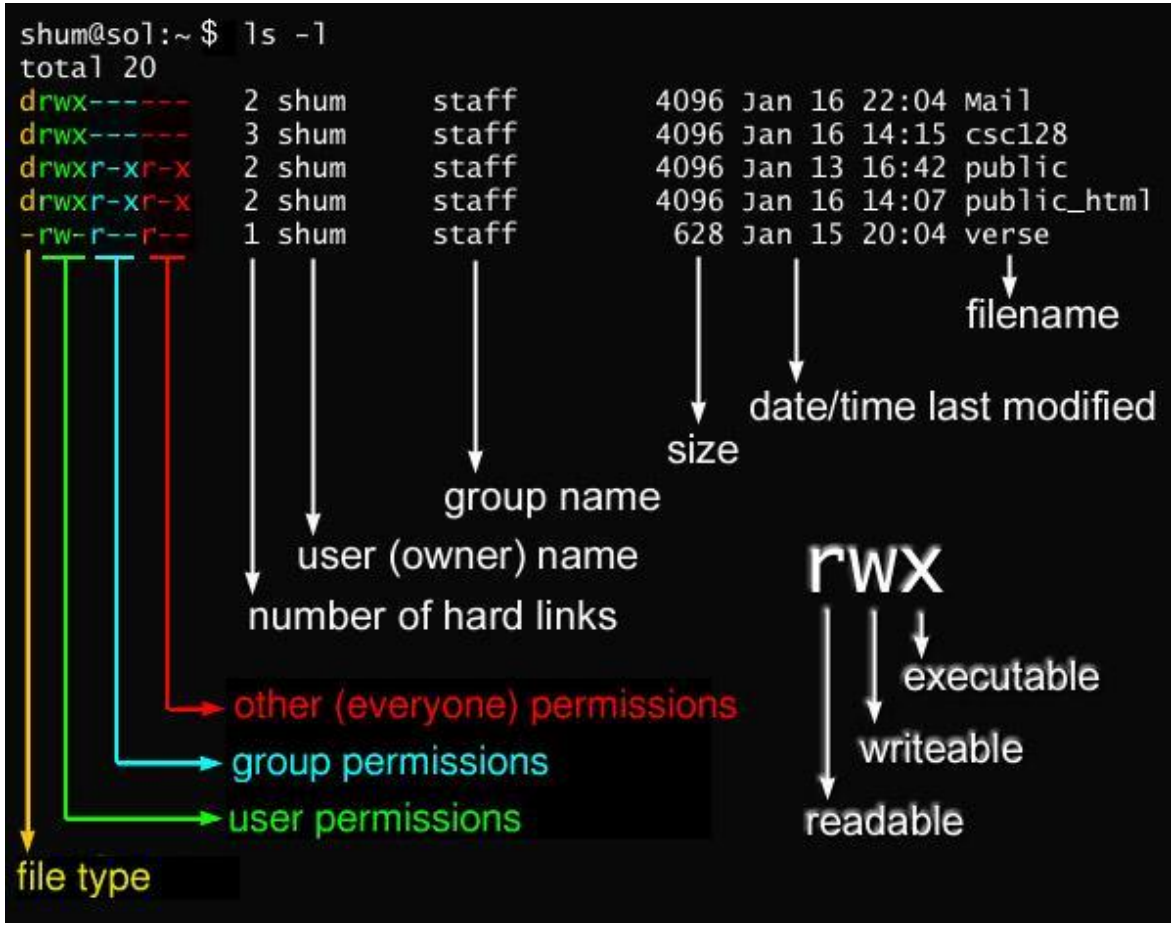
The Basics: Dealing with Files

- **mv**: move/rename a file
- **cp**: copy a file
- **rm**: remove a file
  - r: remove directories and their contents recursively
- **mkdir**: make a directory
- **rmdir**: remove an empty 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
  - s: show size of each file, in blocks

The Basics: Changing File Attributes

- **ln :** creates a link
  - **Hard links** : Point to physical Data
  - Additional name for an existing file
    - ln file1 hlink1
  - **Soft Links/ Symbolic Links (-s):** Point to file
    - ln –s <source file> <my 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: File Permissions



File Permissions

- 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*

chmod contd…

- **Numeric**

#	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

- **Symbolic**

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	read a file or list a directory's contents
w	write	write to a file or directory
x	execute	execute a file or recurse a directory tree

Special permissions

- **setuid** : set user ID on execution
- Permits users to run certain programs with escalated privileges
- E.g. : chmod u+s file1
- When an executable file's setuid permission is set, users may access the program with a level of access that matches the owner
- E.g. passwd command

```
ls -l /usr/bin/passwd
```

```
-rwsr-xr-x 1 root 54192 Nov 20 17:03 /usr/bin/passwd
```

Special permissions contd…

- **setgid** : Grants permission of the group which owns the file
- E.g. : chmod g+s file2

```
ls -l myfile2
```

```
-rw-r-sr-- 1 user 0 Mar 6 10:46 myfile2
```

Basic Shell Commands

- man
- cat
- head
- tail
- du
- ps
- kill
- diff
- cmp
- wc
- sort
- echo

The Basics: Redirection

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

Handling Files Example

1. Create two files

   $ touch blah1 $ touch blah2

2. Fill contents into the files and print them

   $ echo "Cat" > blah1 $ echo "Dog" > blah2

   $cat blah1; cat blah2

   Cat

   Dog

3. Create links: blah1-hard and blah2-soft

$ ln blah1 blah1-hard

$ ln -s blah2 blah2-soft

4. Change the original file : blah1-new, nlah2-new

$ mv blah1 blah1-new

$ cat blah1-hard => Cat

$ mv blah2 blah2-new

$ cat blah2-soft =>

cat: blah2-soft: No such file or directory

find command

- -type: type of a file (e.g: directory, symbolic link)
- -perm: permission of a file
- -name: name of a file
- -user: owner of a file
- -maxdepth: how many levels to search

find contd...

- ?: 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.
- Examples:
  - find . -name my*
  - find . -name my* -type f
  - find / -type f -name myfile

man command

- Extensive documentation that comes preinstalled with almost all substantial Unix and Unix-like operating systems
- Usage
  - read a manual page for a Linux command
    - **man <**command_name**>**
- Hit “q” to get out of man page

wh commands

- `whatis` <command>: returns Name section of man page
- `whereis` <command>: locates the binary, source, and manual page files for a command
- `which` <command>: locates only the binary

diff command

- A file comparison utility that outputs the differences between two files.
- Usage:
  - diff file1 file2
  - Diff –u file1 file2 (unified format)

wget command

- A computer program that retrieves content from web servers
- Usage
  - wget <URL>

vi editor

- Open a file- **vi <filename> or vim <filename>**
- Close a file- **:q**
- Save a file- **:w**
- Save and close a file- **:wq**
- Modes:
  - Normal: Enter commands
  - Insert: Insert text
  - Visual: Like normal, but you can highlight
  - Replace: Like insert, but you replace characters as you type
  - Recording: Record a sequence of key sequences

Emacs editor

- "The customizable, extensible, self documenting display editor"
- Customizable (no programming)
  - Users can customize font, colors, etc
- Extensible (programming required)
  - Run Lisp scripts to define new commands
- Self-documenting
  - C-h r (manual) and C-h t (tutorial)
- https://www.gnu.org/software/emacs/refcards/pdf/refcard.pdf
- http://bit.ly/2CQy3H8 (some basic commands)
- http://stanford.io/2CTWNyl

Getting started with emacs

- emacs already installed in linux servers
- Type emacs to enter into the editor
- Emacs has both GUI and CLI
- All emacs commands start with ‘C’ or ‘M’
  - ‘C’ = ctrl; ‘M’ = alt (windows)/ option (Mac)
- Start emacs
  - emacs <filename>
- Exit emacs
  - C-x C-c

Basic emacs editing

- **Insert text** by simply typing it
- **Undo** by typing C-x u
- **Save changes** by typing C-x C-s
- **Copy, cut, paste**
  - C-space (starts selecting region)
  - M-w (copy a region)
  - C-w (cuts a region)
  - C-k (kill a line)
  - C-y (yank/paste)

Moving around

Keystrokes	Action
C-p	Up one line
C-n	Down one line
C-f	Forward one character
C-b	Backward one character
C-a	Beginning of line
C-e	End of line
C-v	Down one page
M-v	Up one page
M-f	Forward one word
M-b	Backward one word
M-<	Beginning of buffer
M->	End of buffer
C-g	Quit current operation

More emacs commands

- Search – C-s
- Replace – M-%
- Accessing menu – F10
- Switch buffer – C-x b
- Switch current window – C-x o
- Kill the current window – C-x 0 (zero)
- Help – C-h

Directory edit (dired) (C-x d)

- Creates an Emacs buffer containing list of directory contents
- Allows you to operate on files
- Allows you to navigate filesystem
- + - new directory, C-x C-f new file in directory, g - refresh dired buffer
- ! - run shell command
- https://www.gnu.org/software/emacs/refcards/pdf/dired-ref.pdf

Other features

- Emacs as lisp interpreter
- Use scratch buffer
  - type (random) or (+ 1 2) or (setq x 2) then C-j
  - M-: and an expression to evaluate, e.g. (* 1 2 3)
- Run shell command - M-!

Assignment 1: Example key1.txt

key1.txt is for both LABORATORY and HOMEWORK section

1. C-s H E L L O W O R L D
2. C-s H T M L
3. C-d
4. C-n
5. M-x goto-line Enter 1 2 3 Enter

Assignment 1: Example ans.txt

ans1.txt is specifically for LABORATORY section

- 1. Here is the answer to question 1
- 2. Here is the answer to question 2
- 3. Here is the answer to question 3
- …..