

---

# **CS 35L Software Construction Lab**

Fall 2018  
TA: Guangyu Zhou  
Lab 3

---

---

# Introduction to Linux

Week 1

---

# Outline

---

- **Overview of Linux**
  - **Basic commands for Linux**
  - The vim and Emacs editor
-

# CLI vs. GUI

---

## **Client (CLI)**

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

## **Graphics User Interface (GUI)**

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

# Which Linux for this course?

---

## Ubuntu Linux Distribution

- Most popular
- Frequently updated, fixed release cycle (6 months)
- Simple installation and booting
- Nice set of pre-installed packages

## Seasnet servers:

- Red Hat
-

# GNU/Linux

---

- Open-source operating system
    - **Kernel:** core of operating system
      - Allocates time and memory to programs
      - Handles file system and communication between software and hardware
    - **Shell:** interface between user and kernel
      - Interprets commands user types in
      - Takes necessary action to cause commands to be carried out
    - **Programs**
-

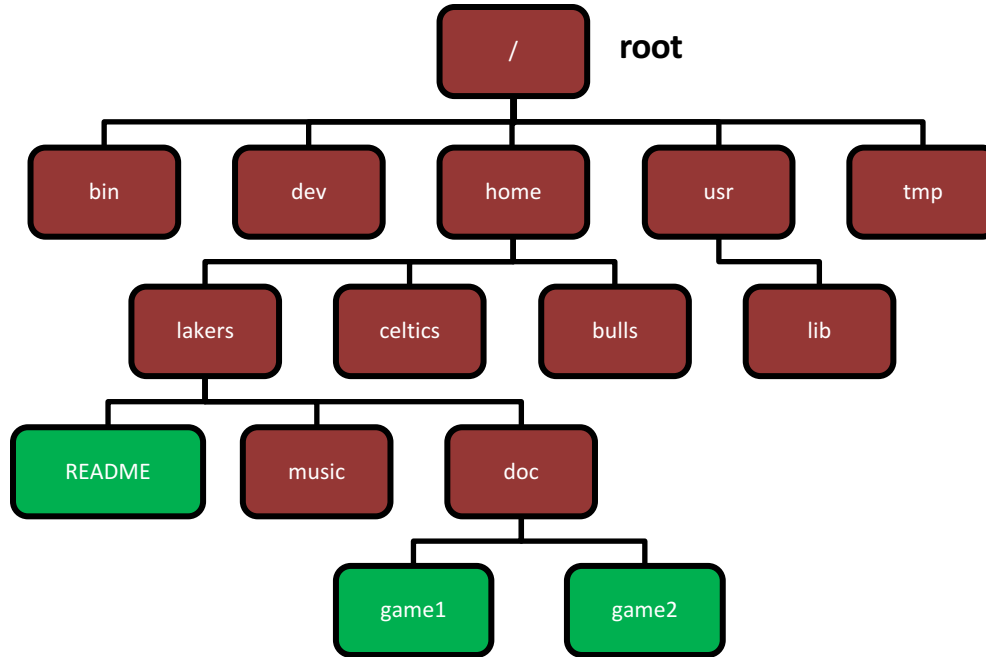
# Files and Processes

---

- 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
-

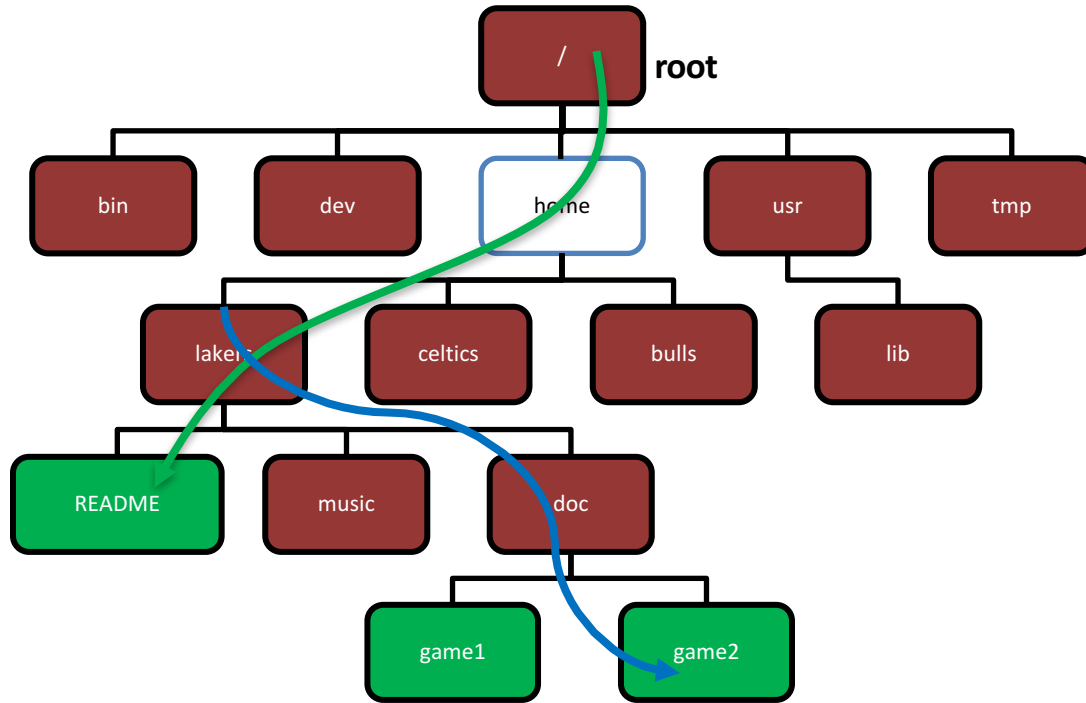
# Linux File System Layout

- Tree structured hierarchy





# Absolute Path vs. Relative Path



Current directory: home

# The Basics: Moving Around

---

- Lost?
    - man: get manual or man pages
  - **pwd**: print working directory
  - **cd**: change working directory
  - ~: home directory
  - .: current directory
  - /: root directory, or directory separator
  - ..: parent directory
-

# The Basics: Look These Up

---

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

Use **man** command!

e.g.    man cat

---

# 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: Dealing with Files

---

- The basics continued...
  - 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
    - -s: show size of each file, in blocks

# 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. i.e. `[1-9][a-z]`

Detailed usage will be covered in Regular Expression (Week 2)

---

# The Basics: Redirection

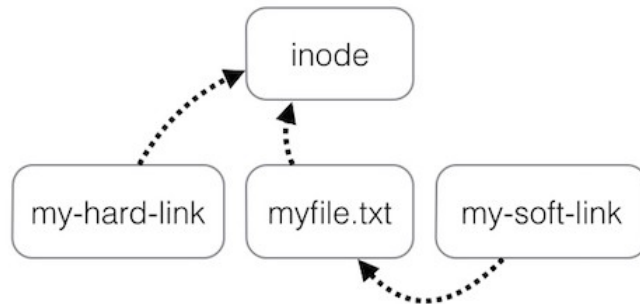
---

- `> 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

---

- **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)
- **ln**: create a link
  - Hard links: point to physical data
  - Soft links aka symbolic links (-s): point to a file

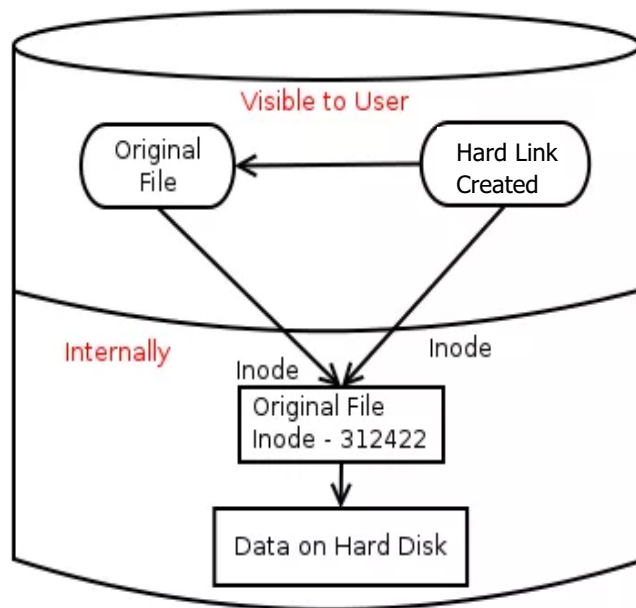




# Hard link vs Soft link

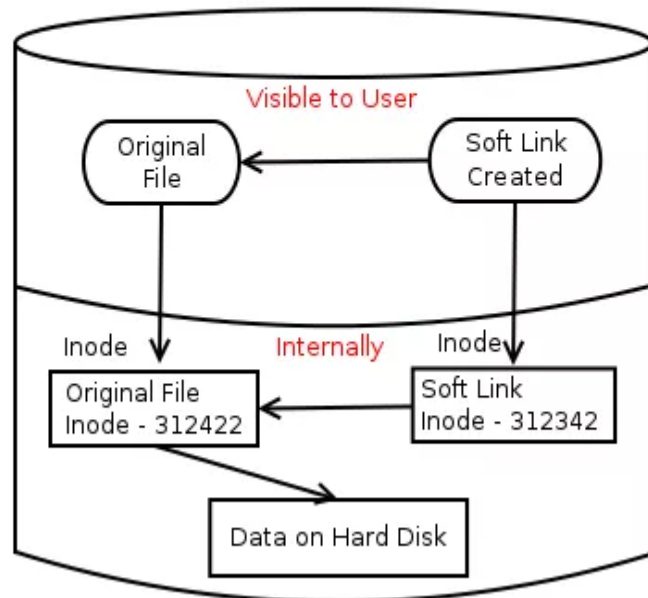
## Hard Link

Hard Link is direct pointer to the original inode of the original file. If you compare the original file with hard link, there won't be any differences.



## Soft Link / Symlink

A softlink is a file that have the information to point to another file/inode. That inode points to the data on the hard drive.



# 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 <i>ugo</i>

---

# The Basics: chmod (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

---

# Linux File Permissions

```
shum@sol:~$ ls -l
total 20
drwx----- 2 shum  staff  4096 Jan 16 22:04 Mail
drwx----- 3 shum  staff  4096 Jan 16 14:15 csc128
drwxr-xr-x  2 shum  staff  4096 Jan 13 16:42 public
drwxr-xr-x  2 shum  staff  4096 Jan 16 14:07 public_html
-rw-r--r--  1 shum  staff   628 Jan 15 20:04 verse
```

Annotations for the `ls -l` output:

- file type**: Indicated by the first character of the permissions (e.g., `d` for directory, `-` for regular file).
- number of hard links**: The number following the permissions (e.g., `2`, `3`, `2`, `2`, `1`).
- user (owner) name**: The user name following the number (e.g., `shum`).
- group name**: The group name following the user name (e.g., `staff`).
- size**: The file size in bytes (e.g., `4096`, `628`).
- date/time last modified**: The date and time the file was last modified (e.g., `Jan 16 22:04`).
- filename**: The name of the file (e.g., `Mail`, `csc128`, `public`, `public_html`, `verse`).

Permissions breakdown for `drwxr-xr-x`:

- file type**: `d` (directory)
- user permissions**: `rwx` (read, write, execute)
- group permissions**: `r-x` (read, execute)
- other (everyone) permissions**: `r-x` (read, execute)

Permissions breakdown for `-rwx`:

- file type**: `-` (regular file)
- permissions**: `rwx` (read, write, execute)
- readable**: `r`
- writable**: `w`
- executable**: `x`

# Special Permissions

---

- sticky bit (o+t)
    - On shared directories, it locks files within the directory from being modified/deleted by users other than the file creator, owner of the directory, or root, even if others have write permissions (Example: /tmp)
  - setuid, setgid (u+s, g+s)
    - “set user ID upon execution”
    - Run an executable with the permissions of the executable’s owner or group
-

# The basics: dealing with process

---

- Process: An instance of a computer program in execution

- ps

- List processes that are currently running

- kill

- Terminate a certain process
  - Usage
    - kill PID

```
[root@tecmint ~]# ps
  PID TTY          TIME CMD
 2232 pts/0        00:00:00 bash
 2256 pts/0        00:00:00 ps
[root@tecmint ~]#
```

# 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
  - -o: or
  - -ls: list current file
-

# find Examples

---

- Examples
  - `find . -name my*`
  - `find . -name my* -type f`
  - `find / -type f -name myfile -print`



# The Basics

---

- First: learn to use man command!
  - Supplement materials: Linux command cheat sheet  
<http://www.rain.org/~mkummel/unix.html>
  - Online document:  
<http://www.linuxdevcenter.com/cmd/>
-

# Assignment 1 is available

---

- Check: <http://web.cs.ucla.edu/classes/fall18/cs35L/assign/assign1.html>

- Deadline:

11:55 PM on Oct 6<sup>th</sup>, 2018

---

# Lab1: hints

---

- For lab questions(ans1.txt)
    - Answer 15 questions using natural language
    - For each question, list all the commands used to solve it
    - Give some explanations about your choice of commands
    - Will be graded manually
-