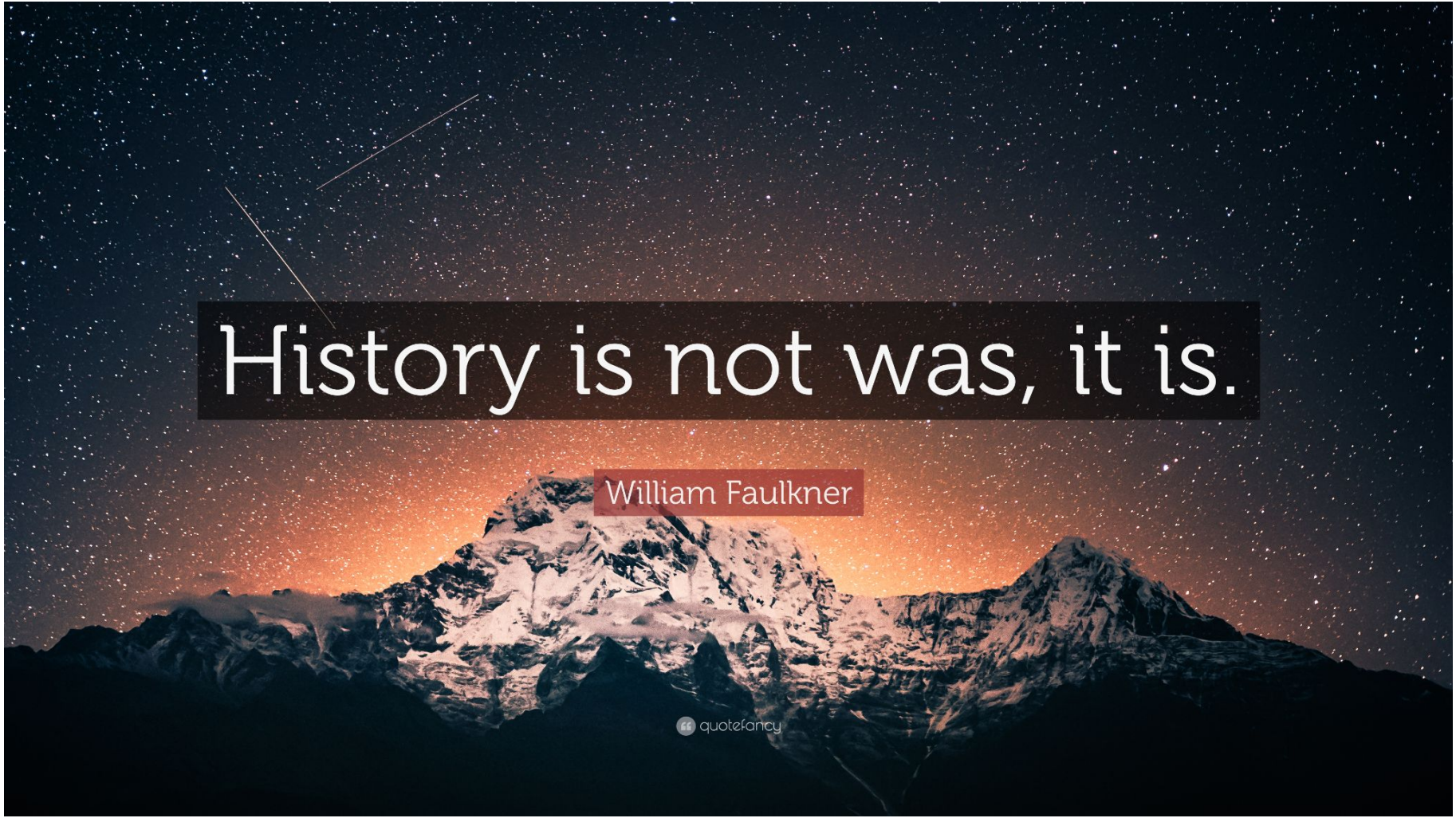
Unix Command Line

Kameswari Chebrolu
Department of CSE, IIT Bombay

https://pbs.twimg.com/media/E-YJGozUUAA6rUU.jpg

Outline

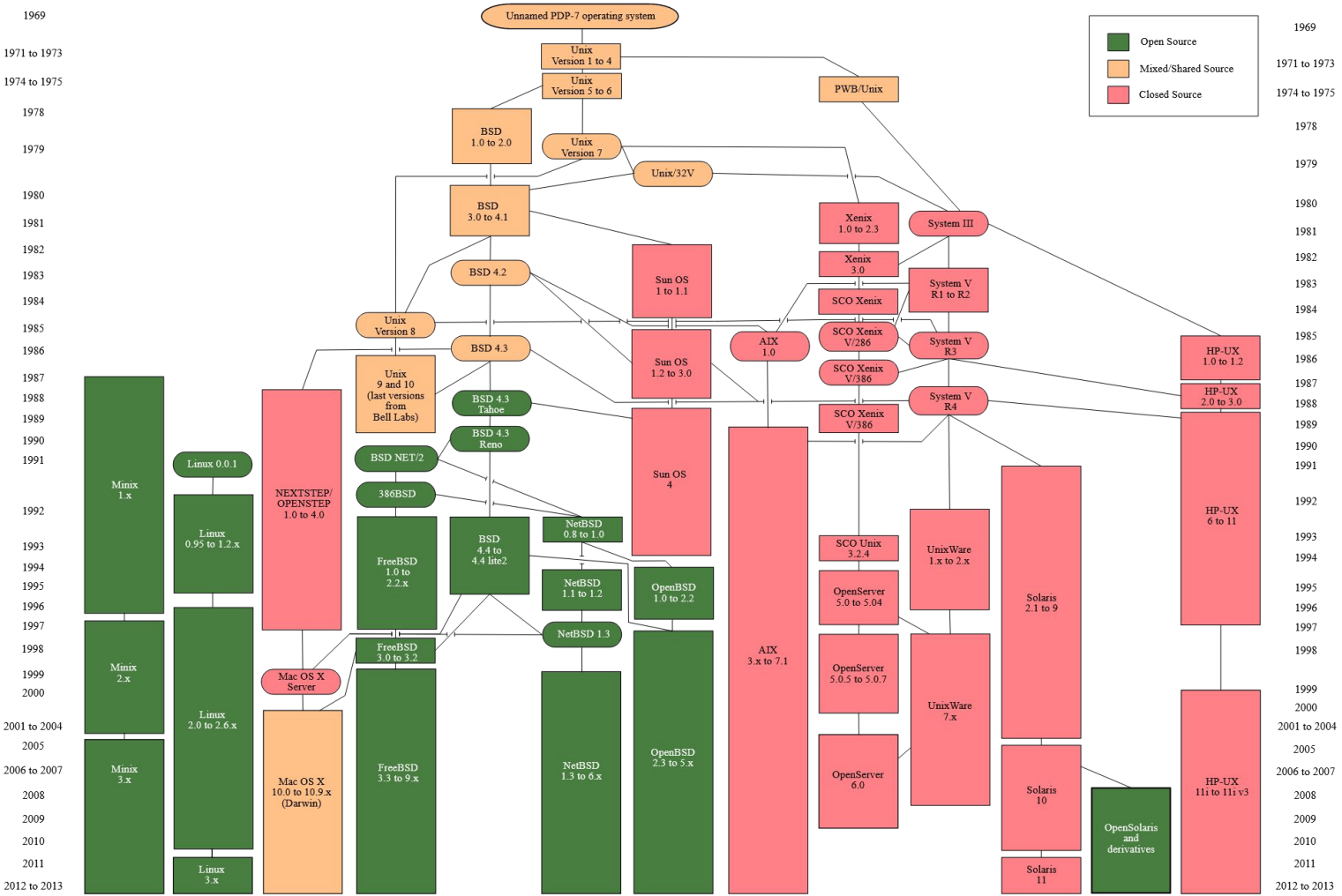
- Unix history and why popular?
- Command line vs GUI
- What is a Shell?
- Linux File System
- Various commands


History is not was, it is.
William Faulkner
quotefancy

Unix/Linux OS

- Unix: Proprietary OS created in late 1960s at AT&T Bell Labs
- Linux: a clone of Unix, free and open source
  - Written from scratch by Linus Torvalds in 1991

- Distributions of Linux: Linux OS packaged with lot of additional free software
  - Fedora, Ubuntu, CentOS, SuSe etc
  - Differ wrt to desktop environment, package installation, display server etc
  - Other Unix clones: FreeBSD and Mac OS X (its kernel Darwin, is based on BSD)
- A user on one Unix system can move to another easily wrt to command-line



https://sosheskaz.github.io/technology/2017/05/12/Adventures-In-Bsd.html

Popularity of *nix

- “Since we are programmers, we naturally designed the system to make it easy to write, test, and run programs” – Unix Creators, Dennis M. Ritchie and Ken Thompson
  - Very server and programmer-friendly OS
  - Linux (FREE) is for developers!
  - Easy to do scripting
  - Lot of scientific libraries and programs are written for *nix

- Open source (some versions) and exposes you to an ecosystem of open-source software
  - Helps bridge the concepts you learn with how they're applied in practice.
    - Interested in OS? Dig into details of open source linux and interaction with device drivers
    - Interested in Compilers? Clone gcc source
    - Interested in distributed systems? Clone Hadoop and run a cluster on your laptop
    - Interested in cloud computing? Containers origins in linux

Command Line vs GUI

Windows GUI: use pre-programmed interface ⇒ set of possible actions pre-decided

```
chebrolu@silmaril: ~/web-development-demo
chebrolu@silmaril:~$ mkdir web-development-demo
chebrolu@silmaril:~$ cd web-development-demo/
chebrolu@silmaril:~/web-development-demo$ mkdir dir1 dir2 dir3
chebrolu@silmaril:~/web-development-demo$ ls
dir1  dir2  dir3
chebrolu@silmaril:~/web-development-demo$ mkdir
mkdir: missing operand
Try 'mkdir --help' for more information.
chebrolu@silmaril:~/web-development-demo$
```

Command-line Shell: a prog. (scripting) language ⇒ use pre-written programs **AND** compose new scripts!

Power of the Shell

Alias: shell, terminal, console, prompt etc

1. Rename a set of files
2. Number of lines in all C files in a directory
3. Top five files with maximum number of lines

Demo!

A Brief History of the Shell

- Unix: OS for mainframe computers
  - Users connecting remotely via individual terminals (keyboard and screen)
  - No local programs, send text and receive text
  - Terminals based on text since text is light on resources
  - Commands kept very terse to reduce the number of keystrokes needed

- Need to support all kinds of file management tasks
  - Create files, list files, rename, move to folders etc
  - Each task required its own program (or command)
  - **Master program to coordinate execution of all these programs → shell**
- Original Unix shell called sh (Bourne shell)
  - Extended with better features and syntax is BASH (Bourne Again SHell)
  - Other shells also: zsh (mac OS), csh, fish etc

Basic Instructions

- Open shell: Click on “Activities” top left of the screen + type shell in the search box (or) use Ctrl-Alt-T
- Type a command in the same line as where $ (prompt) appears (command line ;-)
- Commands sometimes have number of arguments (command-line arguments)
  - tar -zcvf lab1.tgz lab1/

- The shell does not execute commands until the “Enter key” is pressed
- Any output the shell produces will usually be printed directly in the terminal
  - Another prompt is shown once finished
- Commands are case sensitive (ls vs LS)

Demo!

my folder : `Downloads`

me : `cd downloads`

Linux :

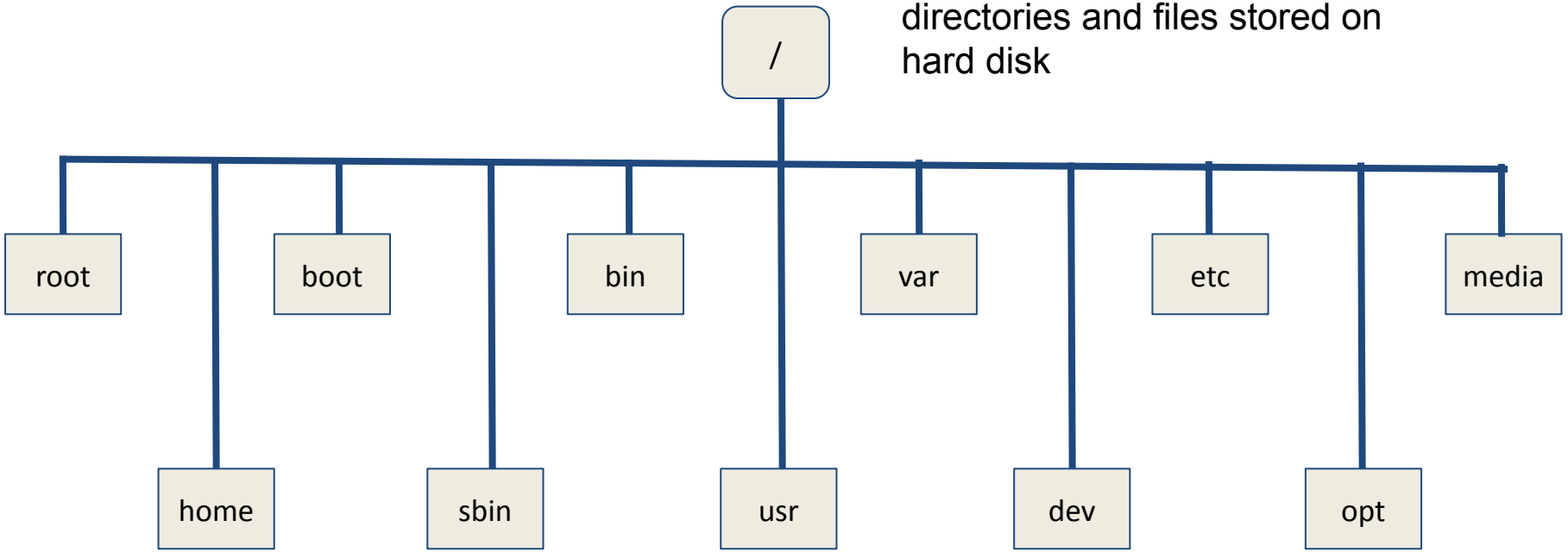
https://preview.redd.it/3zoe90gmq7y51.jpg?auto=webp&s=791da8fa068769a9e73f494f63e730c7f354fb86

Outline

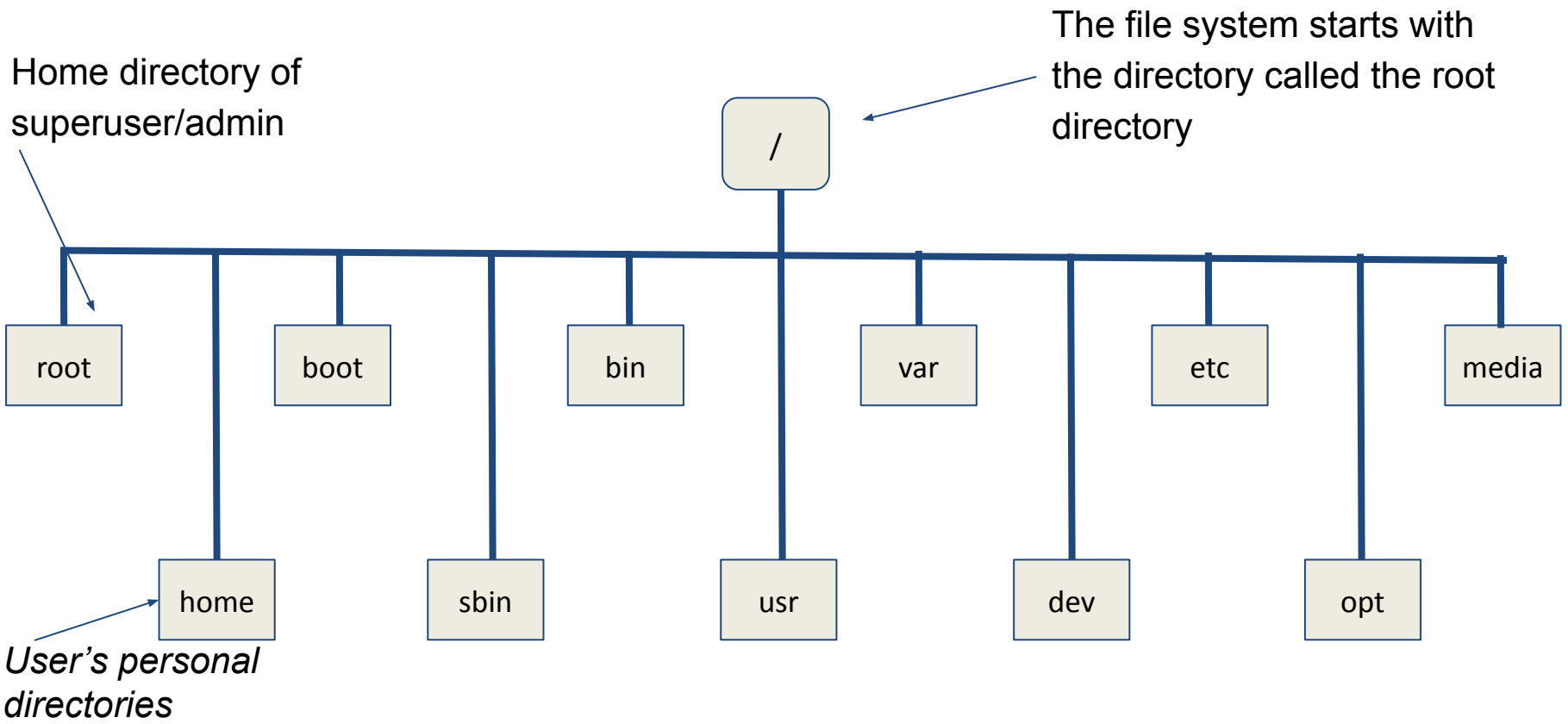
- ~~Unix history and why popular?~~
- ~~Command line vs GUI~~
- ~~What is a Shell?~~
- Linux File System
- Various commands

Linux Filesystem

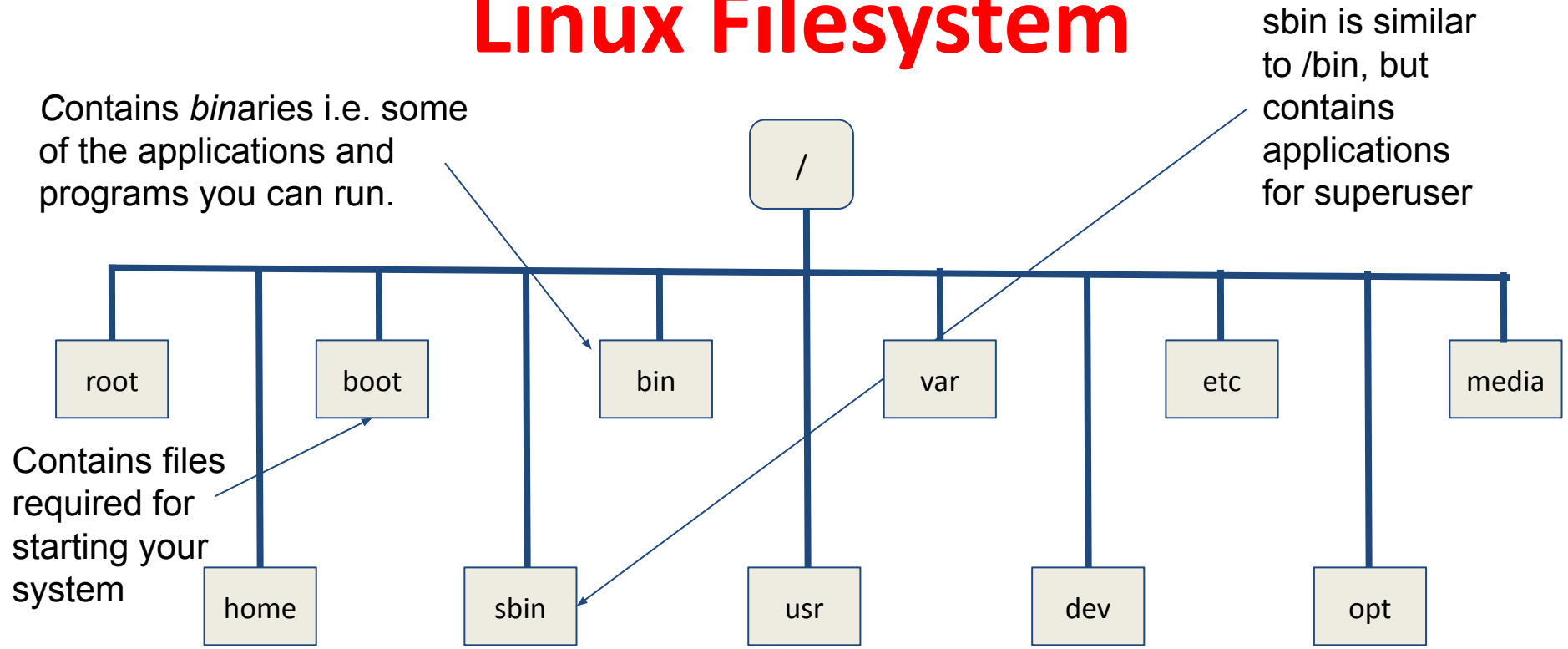
A filesystem is a hierarchy of directories and files stored on hard disk



Linux Filesystem



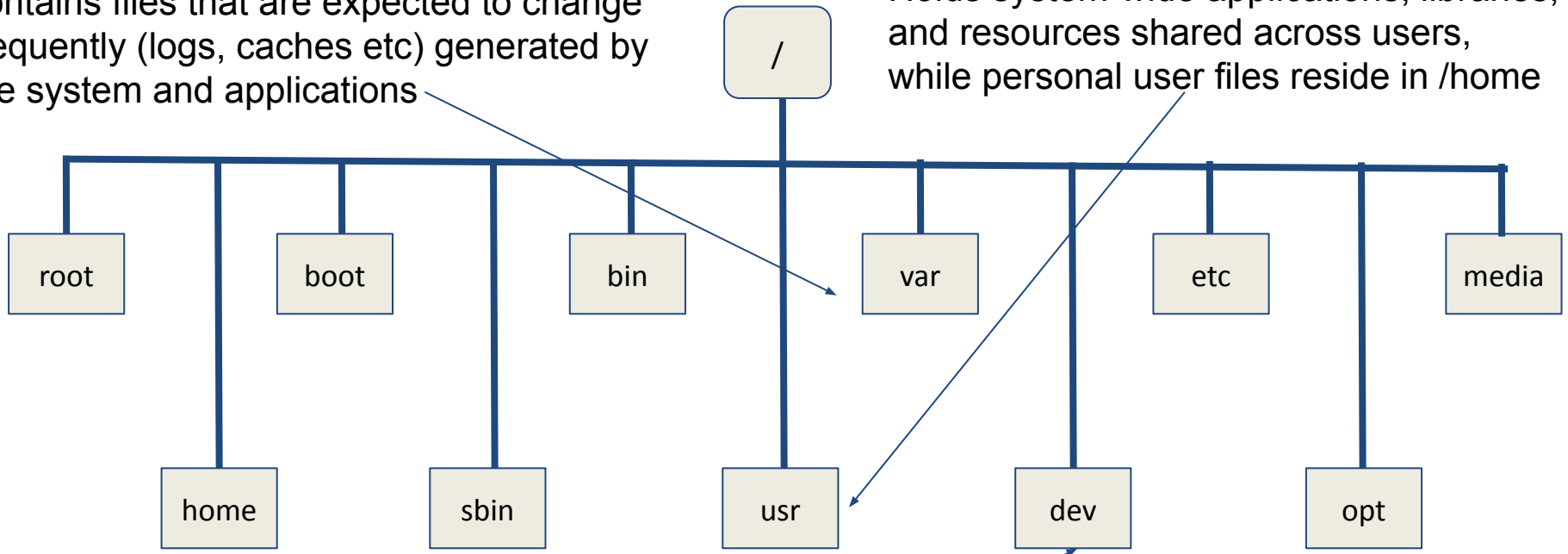
Linux Filesystem



Linux Filesystem

contains files that are expected to change frequently (logs, caches etc) generated by the system and applications

Holds system-wide applications, libraries, and resources shared across users, while personal user files reside in /home

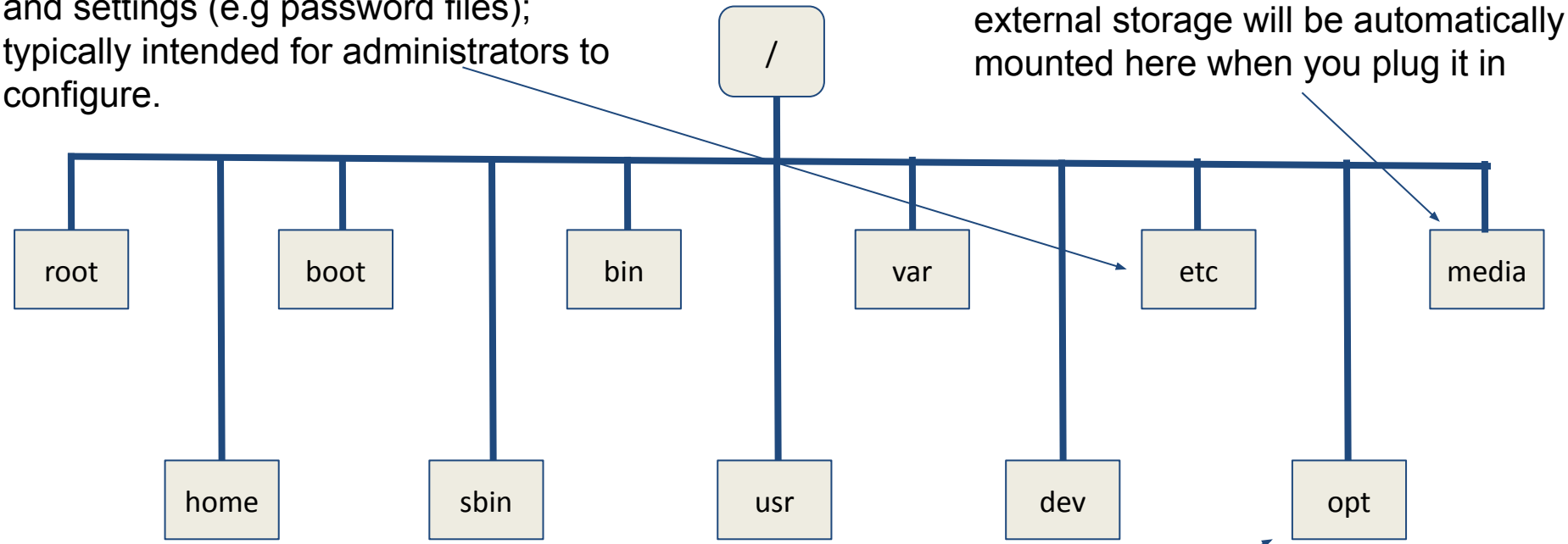


Contains device files (hard drives, USB devices, webcams etc). These files allow software to interact with hardware as if they were standard files

Linux Filesystem

holds system-wide configuration files and settings (e.g password files); typically intended for administrators to configure.

external storage will be automatically mounted here when you plug it in



optional or third-party software packages that aren't part of the default Linux distribution

Outline

- File and Directory Commands
- File Viewing and Editing Commands
- Commands for File Analysis
- Process Management
- Security and Permissions

File and Directory Commands

- clear
- man
- pwd
- ls
- cd
- mkdir
- rmdir
- cp
- mv
- rm

- These commands enable users to
  - Navigate the file system
  - Create, move or remove files and directories
- Provide a powerful interface for interacting with the operating system

Clear

- Clears the terminal screen
  - Terminal cursor moves to the top-left corner
- Helps enhance readability
  - Use before running new commands to avoid clutter and improve focus
- Note: Doesn't delete history or affect running programs
  - Only affects visual display

man

- Displays manual (help) pages for Unix commands
- Useful when learning new commands or options you are unfamiliar with
- Provides detailed documentation
  - Descriptions, options, usage examples, and technical details

- Usually formatted with a consistent structure
  - NAME, SYNOPSIS, DESCRIPTION, OPTIONS, EXAMPLES, SEE ALSO.
- Use the arrow keys or page up/down to scroll through the manual
- Press / followed by a keyword to search within the manual page
- Press q to quit the manual page

- Syntax: man [command]

pwd

- Shell has a notion of a default location
  - For the root user, home is at /root
  - Regular users, it is /home/username (e.g. /home/chebrolu)
- pwd (present working directory) command tells your current working directory
  - No options needed
  - Displays the full path of the directory you are currently in

- Use Case:
  - Helpful when navigating directories
    - Use pwd to confirm your current directory, especially when working in deep or complex directory structures
  - Helpful with scripting and automation
    - Dynamically get the current directory and perform operations relative to it

Demo

man, clear, pwd

ls

- ls: display contents of the current directory
  - Directories often listed in a different color (e.g., blue)
  - Executable Files may be displayed in green
- Syntax: ls [Options] [Files/Directories]
- Use Case:
  - Quickly see what files and directories exist in your current or specified directory
  - Checking file details like permissions or file size

- Key Options:
  - -l : Shows detailed information
    - File permissions, number of links, owner, group, size, and modification date
  - -lh : Displays file sizes in a human-readable format (e.g., KB, MB)
  - -lt : Sorts the output by the time of last modification, with the newest files first
  - -a : display all files including the hidden files
    - Every directory has at least two entries: “.” and “..” (called dot and dotdot)
      - dot directory is a shortcut for the current directory
      - dotdot is a shortcut to the parent directory
  - -R: list subdirectories recursively
  - -S: sort by file size, largest first
  - -X : sort alphabetically by entry extension

cd

- Changes the current working directory
  - Absolute paths:
    - “/” at the start of your path means “starting from the root directory”
    - (”~”) at the start of your path means “starting from my home directory”
  - Relative Path: Starts from the current directory
    - e.g. ../folder (moves up one directory)
- Syntax: cd [directory]
  - Directory you want to navigate to
  - If omitted, cd defaults to the home directory
- Use Case: efficient file system navigation
  - Enables users to work effectively within different directories

- Key Options:
  - No Options: takes you to your home directory
  - .. : Moves you up one directory level
  - - : Switches to the previous directory
  - ~ : Represents home directory, useful for quickly navigating there
- “Tab” for auto filling
  - Applies to all commands, not just cd!

Demo

ls, cd

mkdir

- Creates new directories
  - Directories can be created using either absolute paths (starting from /) or relative path
  - Directory names can include special characters
    - Best to avoid spaces and stick to alphanumeric characters and underscore
- Use Case: Command helps create directories for organizing files in new projects

- Syntax: mkdir [OPTIONS] [DIRECTORY]
  - Takes one or more directory names as its arguments
  - If the directory already exists and if you don’t use -p, mkdir will return an error
- Key Options:
  - -p : creates the directory only if it doesn’t exist, makes parent directories as needed
  - -v: v stands for verbose, displays a message for each directory that is created

rmdir

- Removes empty directories from the file system
  - If the directory contains any files or subdirectories, it cannot be removed with rmdir
    - Will return an error
- Syntax: rmdir [options] directory_name
  - -p option: removes the specified directory and its parent directories if they are empty
- Use case: clean up empty directories left after moving or deleting files

- Comparison with rm -r:
  - rm -r command can remove directories that contain files or subdirectories
  - Use rmdir when you want to ensure that only empty directories are deleted

Demo

mkdir, rmdir

cp

- Copies files and directories from one location to another
- Syntax: cp [options] source destination
  - source: Files or directories you want to copy
  - destination: Location where you want to copy the file or directory
  - source can be one, or more files or directories, and destination can be a single file or directory
    - When multiple files or directories are given as a source, the destination must be a directory
  - If the destination file already exists, cp will overwrite it without warning

- Usage:
  - Create backups of important files or directories
  - Create a copy of a file before making changes with original as a reference
- Key Options
  - -i : Prompts before overwriting an existing file
  - -r : Copies an entire directory and its contents, including subdirectories
  - -v : (verbose mode) Displays the files being copied, useful for tracking the operation

mv

- Moves or renames files and directories
  - Very similar in spirit to cp, except moves instead of copying
  - mv copies the file to the new location and then deletes the original
- Syntax: mv [options] source destination
- Use Case: Move or rename files and directories to improve organization

- Key Options:
  - -i (Interactive): Prompts for confirmation before overwriting an existing file
  - -f (Force): Forces the move operation without prompting for confirmation, even if it involves overwriting files
  - -u (Update): Moves the source file only if it is newer than the destination file or if the destination file does not exist
  - -v (Verbose): Provides detailed information about the files being moved, including the source and destination paths
  - -n (No Clobber): Prevents overwriting of existing files. If a destination file exists, the move is not performed

rm

- Removes (deletes) files and directories from the file system
- Syntax: rm [options] file_or_directory
  - file_or_directory: file or directory you want to delete
- Use case: Managing disk space and keeping file systems organized
  - Clean up temporary files, remove old backups, or clear out directories

- Key Options
  - -f (Force): Forces the removal of files without prompting for confirmation
    - Useful for removing write-protected files
  - -i (Interactive): Prompts for confirmation before deleting each file
  - -r or -R (Recursive): Deletes directories and their contents recursively
    - This option is required for deleting non-empty directories
  - -v (Verbose): Displays detailed information about the files being deleted
  - -d (Directory): Removes empty directories.

Demo

cp, mv, rm

Outline

- ~~File and Directory Commands~~
- File Viewing and Editing Commands
- Commands for File Analysis
- Process Management
- Security and Permissions

File Viewing and Editing Commands

LINUX TERMINAL FOR BEGINNERS

https://www.reddit.com/r/linuxmemes/comments/oybfil/cat/?rdt=54918

File Viewing and Editing

- echo
- touch
- cat
- less and more
- head and tail
- Editors: vi, nano and gedit

echo

- Displays a line of text or string to the standard output (usually the terminal)
  - Similar to the print function in many programming languages
- Syntax: echo [options] [string]
  - string: the text or variables you want to display
- Use case: Widely used in scripting for providing user feedback, logging, and output formatting

- Key Options
  - -n (No Newline): No newline added at the end of the output
  - -e: Enables interpretation of escape characters (like \n, \t, etc.).
  - -E : Disables interpretation of backslash escapes (default behavior)
- What did echo say to the printf command?
  - "Relax, not everything has to be formatted perfectly!"

touch

- Creates an empty file or updates the timestamp of an existing file
- Syntax: touch [options] file_name
  - file_name: Name of the file to be created or whose timestamp is to be updated
- Common Use case:
  - Creating Empty Files
    - Set up placeholder files for configuration or testing
  - Updating File Timestamps
    - Update the last accessed or modified time of a file
      - Does not alter its content

- Key Options
  - -a : Updates only access time without changing the modification time
  - -m : Updates only the modification time without changing the access time
  - -t : Allows you to specify a particular timestamp instead of using the current time
    - touch -t [[CC]YY]MMDDhhmm[.ss] file.txt
  - -c or --no-create: Prevents touch from creating a file if it does not already exist
    - Only updates timestamps if the file exists

- How to know access and modification times?
  - ls -l --time=atime filename (access time)
  - ls -l filename (modification time)
  - Another alternative: stat
    - stat filename
    - Provides detailed information about a file
- Why did touch go to therapy? :-)
  - To work on its commitment issues — it kept making files but never opened up to them

Demo

echo, touch

cat

- Display the contents of files, combine multiple files into one, and create or append to files
  - Why named cat? can combine (concatenate) outputs also
- Syntax: cat [options] [file1] [file2] …
- Use case: quickly view file content or combine several files into a single output

- Key Options
  - -n : Numbers all lines in the output
  - -b : Numbers only non-empty lines
  - -v : Displays non-printable characters in a visible format
  - -s : Compresses multiple consecutive blank lines into a single blank line.

less

- A file viewer that allows you to view the contents of a file one screen at a time
  - Unlike cat, does not load the entire file at once
  - More efficient for viewing large files
- Syntax: less [options] filename

- Use Case: Efficiently view large files
  - Move up and down through a file using keyboard
    - Space for down, b for up; arrow buttons for scroll
  - Search for specific text within the file using / followed by the search term
    - To go to the next occurrence of the search term, press n
    - To go to the previous occurrence, press N
  - q: quit less and return to the command prompt
- Key Options
  - -N : Displays line numbers alongside the file content
  - -i : Makes search case-insensitive

more

- Used for viewing files one screen at a time, similar to less, but with fewer features
  - Primarily used for paginated output of file content
  - Use space to go forward by one screen, b to move one full screen up.
  - Press q to exit more and return to the command prompt

- Syntax: more [options] filename
- Key Options
  - -n : Defines the number of lines to display at a time
  - +number : Starts viewing the file from a specific line number.
- Less supports search, more doesn’t

Demo

cat, less, more

head

- Used to display the first few lines of a file or a group of files
  - By default, shows first 10 lines
  - But can specify number of lines or bytes to display
- Use case: Quickly view the beginning of a file without opening the entire content

- Syntax: head [options] filename
- Key Options
  - -n: Displays a specific number of lines from the start of the file
  - -c : Displays a specific number of bytes from the start of the file

tail

- Used to display the last part of a file
  - By default, shows the last 10 lines
  - But can customize to display a specific number of lines or bytes
- Use case: Monitoring log files in real-time or seeing the most recent content added to a file

- Syntax: tail [options] filename
- Key Options
  - -n : Displays a specific number of lines from the end of the file
  - -c : Displays a specific number of bytes from the end of the file
  - -f : Continuously outputs new lines as they are added to the file

- less: Best for interactive, full-file viewing with navigation and search capabilities
  - Ideal for exploring and reading large files
- more: Best for viewing files one screen at a time with limited interaction
  - Allows forward scrolling, making it more suitable for sequential reading
- head/tail: Best for quick, non-interactive previews of the start/end of a file
  - Simple, with no navigation or scrolling

Demo

head, tail

vi

- Opens vi editor, a powerful text editor available on most Unix-like systems
  - A modal editor that operates in different modes
  - Some learning curve, onced overcome, allows for very fast text editing
  - Particularly useful in environments where a GUI is not available!
  - An enhanced version of vi is vim (stands for "Vi IMproved")

- Syntax : vi filename
- Key Modes
  - Command Mode: default mode to navigate, delete, copy, and execute commands
    - Press i to enter insert mode
  - Insert Mode: Used for inserting or editing text
    - Any keystrokes in this mode are added directly to the file

Key Commands

- Saving and Exiting
  - Save and Exit: :wq
  - Exit Without Saving: :q!
  - Save Without Exiting: :w
- Navigation
  - Move to Start of Line: 0
  - Move to End of Line: $
  - Move to Start of File: gg
  - Move to End of File: G

- Editing
  - Delete a Line: dd
  - Copy a Line: yy
  - Paste: p
  - Undo: u
  - Redo: Ctrl + r
- Searching
  - Search Forward: /search_term
  - Search Backward: ?search_term
  - Repeat Search: n (for next occurrence)

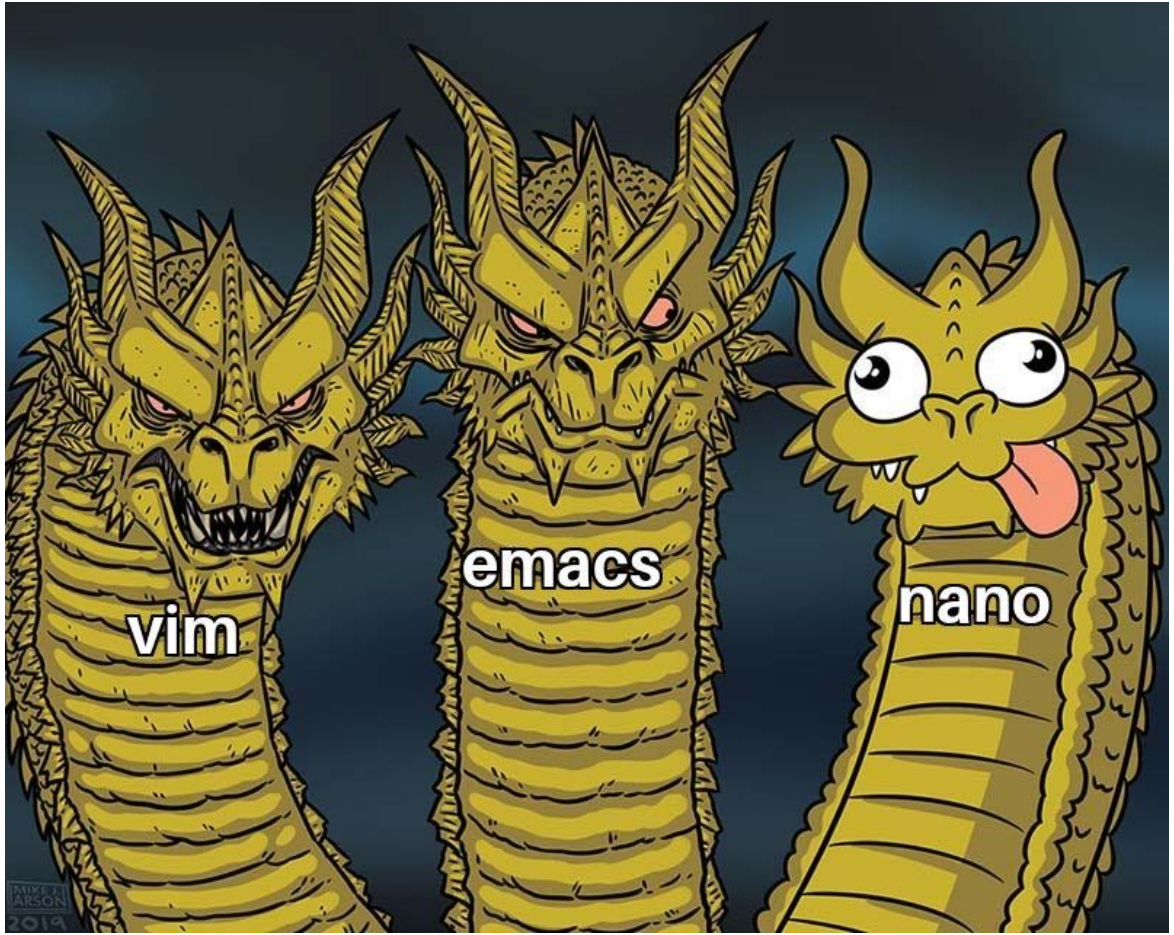
nano

- Opens a simple, easy-to-use text editor available on most Unix-like systems
  - Unlike vi or vim, nano is designed to be user-friendly
  - Keyboard shortcuts displayed at the bottom of the screen, making it more accessible for beginners
- Syntax : nano [options] filename
  - nano -c filename (enables line numbers)

gedit

- Default text editor for the GNOME desktop environment
  - Simple, user-friendly, and accessible GUI based editor
    - Vi and nano are terminal-based editors
  - Menus for saving, searching etc
  - Syntax highlighting for many programming languages
  - Can open multiple files in tab and switch between documents
  - Autosave and backup features to prevent data loss.
- Syntax : gedit [options] [filename]

Demo



https://pbs.twimg.com/media/Eb3V8MGX0AEWbRa.jpg

References

- The Linux Command Line by William Shotts
  - https://linuxcommand.org/tlcl.php
- https://ubuntu.com/tutorials/command-line-for-beginners#1-overview
- https://linuxize.com/ (good resource, use search box for info on different commands!)

Misc.

- File System:

https://www.linuxfoundation.org/blog/blog/classic-sysadmin-the-linux-filesystem-explained

- Figure of Unix variants: https://sosheskaz.github.io/technology/2017/05/12/Adventures-In-Bsd.html