System-Level I/O:
Unix I/O: File Types

Unix I/O Overview

- A Linux *file* is a sequence of *m* bytes:
  - $B_0, B_1, \ldots, B_k, \ldots, B_{m-1}$

- Cool fact: All I/O devices are represented as files:
  - `/dev/sda2` (`/usr` disk partition)
  - `/dev/tty2` (terminal)

- Even the kernel is represented as a file:
  - `/boot/vmlinuz-3.13.0-55-generic` (kernel image)
  - `/proc` (kernel data structures)

File Types

- **Each file has a *type* indicating its role in the system**
  - *Regular file:* Contains arbitrary data
  - *Directory:* Index for a related group of files
  - *Socket:* For communicating with a process on another machine

- **Other file types beyond our scope**
  - *Named pipes (FIFOs)*
  - *Symbolic links*
  - *Character and block devices*

Regular Files

- **A regular file contains arbitrary data**
- **Applications often distinguish between *text files* and *binary files***
  - Text files are regular files with only ASCII or Unicode characters
  - Binary files are everything else
    - e.g., object files, JPEG images
  - Kernel doesn’t know the difference!

Regular Files

- **A regular file contains arbitrary data**
- **Applications often distinguish between *text files* and *binary files***
  - Text files are regular files with only ASCII or Unicode characters
  - Binary files are everything else
    - e.g., object files, JPEG images
  - Kernel doesn't know the difference!
- **Text file is sequence of *text lines***
  - Text line is sequence of chars terminated by *newline char* (‘`\n`’)
    - Newline is `0xa`, same as ASCII line feed character (LF)

Regular Files

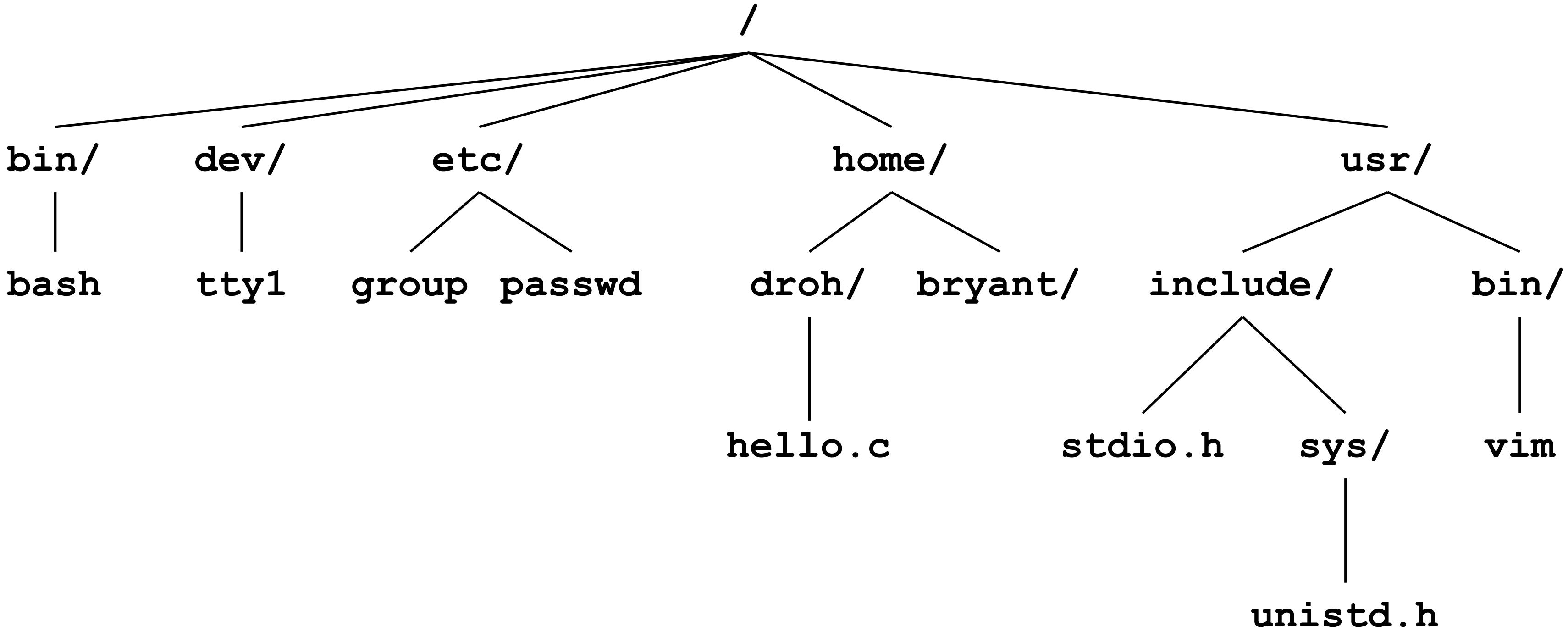
- **A regular file contains arbitrary data**
- **Applications often distinguish between *text files* and *binary files***
  - Text files are regular files with only ASCII or Unicode characters
  - Binary files are everything else
    - e.g., object files, JPEG images
  - Kernel doesn't know the difference!
- **Text file is sequence of *text lines***
  - Text line is sequence of chars terminated by *newline char* (`'\n'`)
    - Newline is `0xa`, same as ASCII line feed character (LF)
- **End of line (EOL) indicators in other systems**
  - Linux and Mac OS: `'\n'` (`0xa`)
    - line feed (LF)
  - Windows and Internet protocols: `'\r\n'` (`0xd 0xa`)
    - Carriage return (CR) followed by line feed (LF)

Directories

- **Directory consists of an array of *links***
  - Each link maps a *filename* to a file
- **Each directory contains at least two entries**
  - `.` (dot) is a link to itself
  - `..` (dot dot) is a link to *the parent directory* in the *directory hierarchy* (next slide)
- **Commands for manipulating directories**
  - `mkdir`: create empty directory
  - `ls`: view directory contents
  - `rmdir`: delete empty directory

Directory Hierarchy

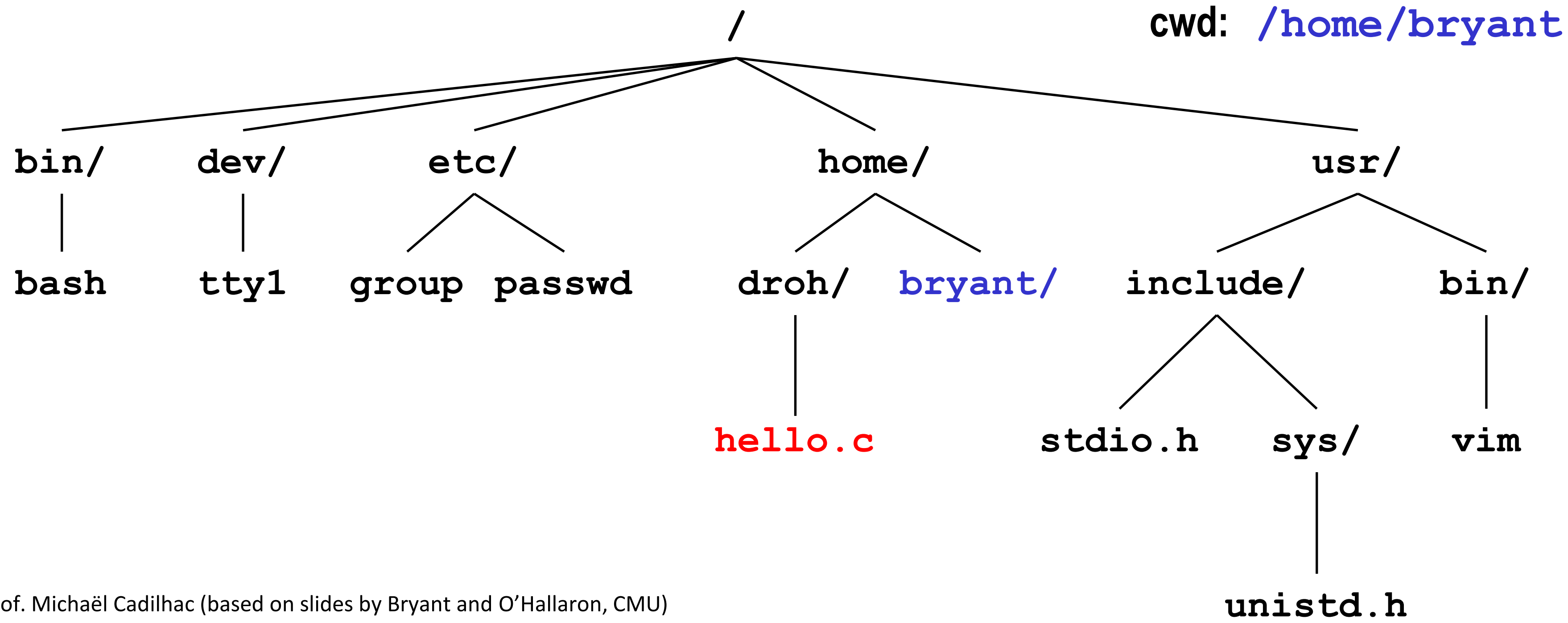
- **All files are organized as a hierarchy anchored by root directory named / (slash)**



- **Kernel maintains *current working directory (cwd)* for each process**
  - Modified using the `cd` command

Pathnames

- **Locations of files in the hierarchy denoted by *pathnames***
  - *Absolute pathname* starts with '/' and denotes path from root
    - `/home/droh/hello.c`
  - *Relative pathname* denotes path from current working directory
    - `../home/droh/hello.c`


/
cwd: /home/bryant
bin/
dev/
etc/
home/
usr/
bash
tty1
group
passwd
droh/
bryant/
include/
bin/
hello.c
stdio.h
sys/
vim
unistd.h