More Linux, make, gcc/g++

Spring 2017 - Aaron S. Crandall, PhD

Today’s Outline

- Announcements
- Thing of the Day
- Linux commands
- g++ and make
- General things like
  - command line conventions
  - STDIN & STDOUT

Announcements

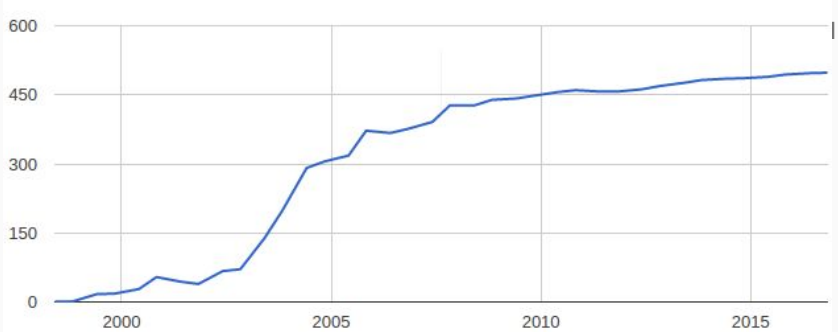
- Linux User's Group is here to help you - tutoring and office hours
  - Once they tell me when… Crandall is getting grouchy with that one
- EECS Undergraduate Student Club Night
  - Tuesday, August 29th @ 4:10pm in Sloan 175 - Come learn about our clubs & orgs!
- Google is visiting campus: Sept 12-14th
  - Hosting talks, internship training, meet their engineers, etc
  - More details to follow as times firm up
- Voiland College 2017 Fall Industry Tour - Nov 20th, 2017
  - Visit Walt Disney company, Dell, and Synapse in seattle
  - Register: wsu.joinhandshake.com/events | vcea.internships@wsu.edu

Thing of the Day:

- South Korea has a company making a giant robot.
- Not joking about it being right out of SciFi http://www.telegraph.co.uk/news/2016/12/27/giant-avatar-style-robot-takes-first-steps-south-korea/

Last class recap

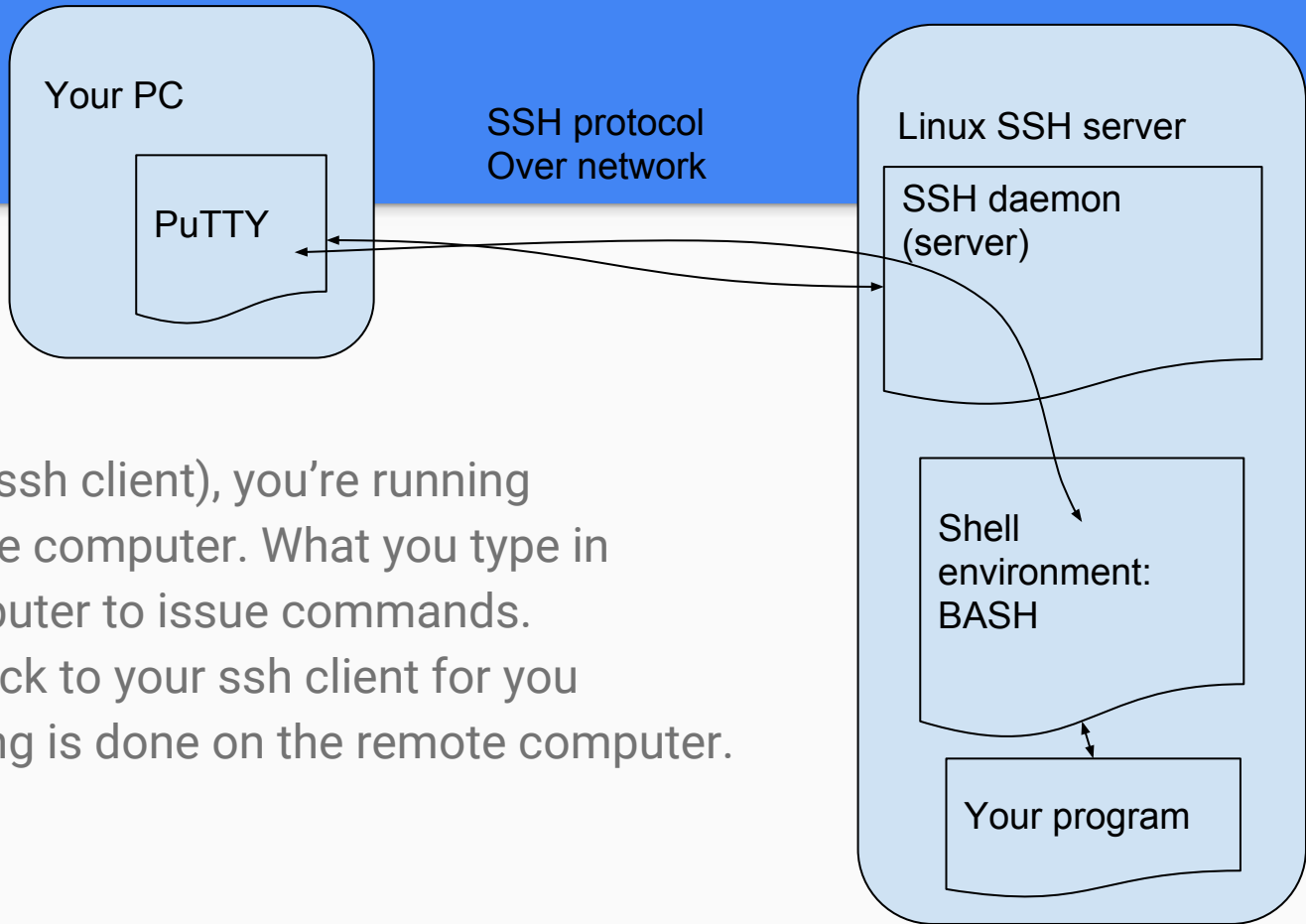
- We talked about what defined an operating system
- What a compiler is
- Why Linux exists and that it has some nice features, especially for developers
- That there's lots of options beyond your normal desktop out there



https://www.top500.org/statistics/list/

Logging into Linux

- Remember to grab an SSH client
  - PuTTY on Windows works great
  - OSX can just fire up the terminal and ssh from there:
    - Applications/Utilities/Terminal
    - ssh username@host -- ssh acrandal@ssh8.eecs.wsu.edu
- SCP to copy files
  - WinSCP for Windows
  - OSX… can just fire up the terminal and scp from there:
    - Applications/Utilities/Terminal
    - scp filename username@host:destinationfilename



When you use PuTTY (ssh client), you're running commands on a remote computer. What you type in goes to the other computer to issue commands. The results are sent back to your ssh client for you to see, but all computing is done on the remote computer.

Some primary command line programs

- ls - List files in directory
- cd - change directory
- rm - remove file
- cp - copy file
- mkdir - make directory
- rmdir - remove directory
- nano / vi / emacs - edit a file
- ssh - use ssh to connect to server
- scp - copy file over ssh to server
- man - manual page for tools

- g++ - use GNU C++ compiler
- make - run make to build a program
- ps - list running programs
- kill - kill a running program
- top - watch running programs

Tons of Linux tutorials out there:
* https://ryanstutorials.net/linuxtutorial/
* http://linuxcommand.org/index.php
*https://www.codecademy.com/learn/learn-the-command-line
* http://www.ee.surrey.ac.uk/Teaching/Unix/

Crandall Advice: Grab a cheat sheet

- The world of UNIX commands is large. As you're starting out, grab a cheat sheet and even keep a notepad of commands you've used until you're more comfortable with the tool set.
- Here's a pretty reasonable one:
  - https://files.fosswire.com/2007/08/fwunixref.pdf

How to run commands on the command line. Elegant, but requires your memory

- On the command line, the first thing you type is the name of a program to run.
- Everything after the name of the program are command line options
  - Unless you chain multiple programs together with pipes
- Command line options tell the program what you want it to do
  - ls (lists the files and directories) … ls -la (lists all including hidden, plus other stuff)
  - The man page (man ls) will tell you more of what's available for a tool
- Eventually, these things start happening very quickly after you practice

Some command line conventions

- STDIN and STDOUT
- Pipes
- Redirection
- Return values
- Signals and special characters
  - Ctrl-C (^C or interrupt) and Ctrl-D (^D or EOF)
  - But ^Z and fg are fun too! Oooo and ^A ^E
- Tab completion is required for pure survival
- History via up & down arrows, or bang bang (!!)

Open Source Summit 2017 in LA
(It's a convention)

A couple more occurred to me

- Your home directory is also called: ~
  - $HOME is the variable holding it too
- Filesystem norms: /home, /etc, /usr, /dev, /var, /tmp, /mnt, /opt, /root
- Can this class be done on a Raspberry Pi computer? Probably
  - Raspbian is a debian fork, just FYI
- Using clear and reset
- Running chained commands: && vs ;
- There are various shells, but most people use bash

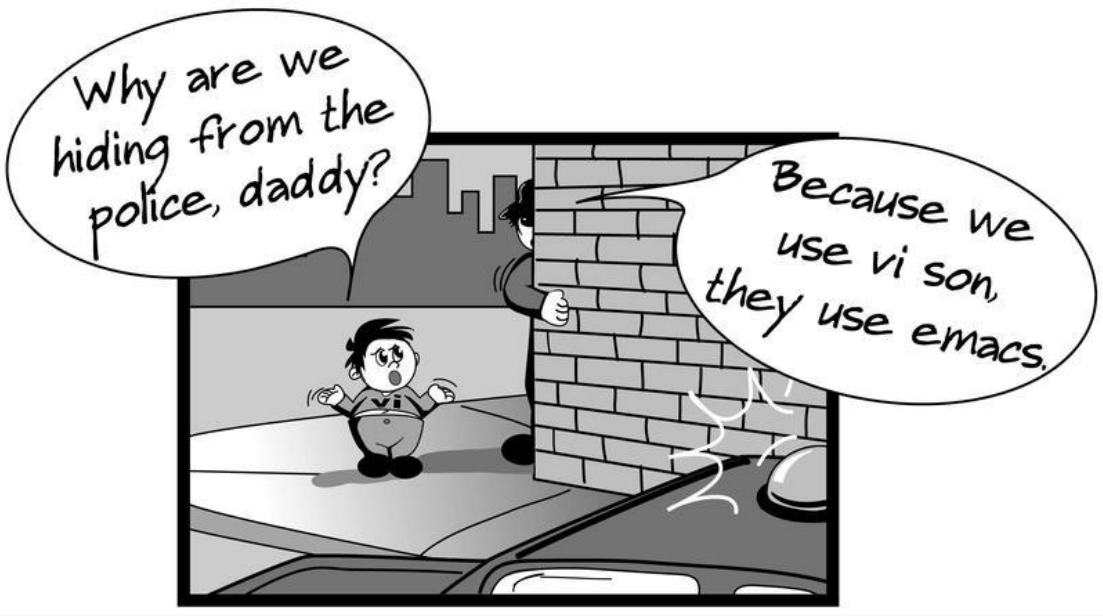
WAIT!!! How do programs use command line options?

- Remember how your programs would sometimes start with:
  int main(int argc, char* argv[])
- Yeah, argc and argv are set by the command line options
- argc is the number of strings (divided by spaces) the program was run with
- argv is an array of char* strings, one with each word
- argc is always at least 1 since the first string is the name of the file used to run the program, including the path
- GUI IDEs (VS) have ways to set the options passed while testing builds

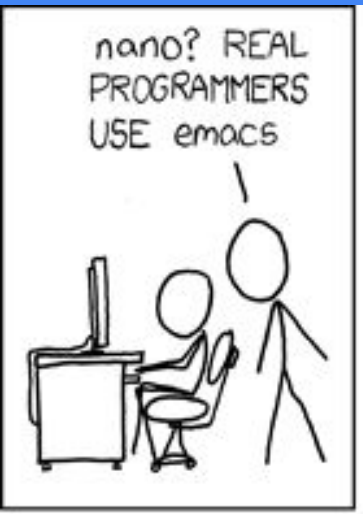
The UNIX filesystem structure

- Where's C:\? --- lolz, we don't need that noise
- Everything lives in a single tree under /
  - This is called "slash" or root (not to be confused with the root user)
- More filesystems (disks, etc) are just mounted under / somewhere
  - Command to add a disk is: mount Removing is: unmount
  - All disks are in the devices directory: /dev
  - Ex: /dev/sda1
- Most of this is taken care of for you in a default Linux install from a distro

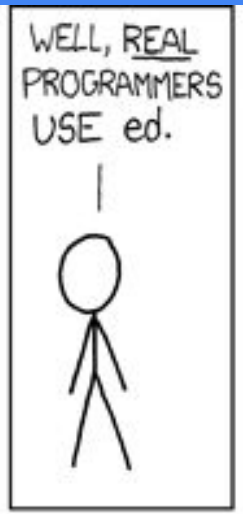
Editing files



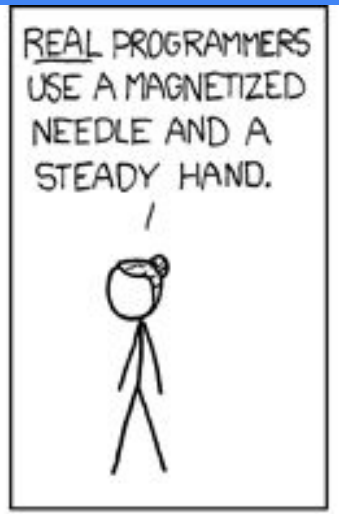
- The big three options:
  - Vi
  - Emacs
  - Nano
- I'm a vi user. (I predate VIM!)
- Emacs users can be wizzards, but I just know how to exit
- Nano is a great starting point, but it's got the limits of a WYSIWYG GUI tool.
  :-(
- There's plenty more options, but these are the big 3

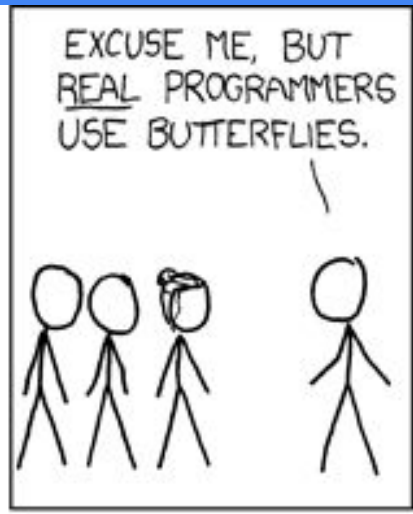

nano? REAL PROGRAMMERS USE emacs


HEY. REAL PROGRAMMERS USE vim.

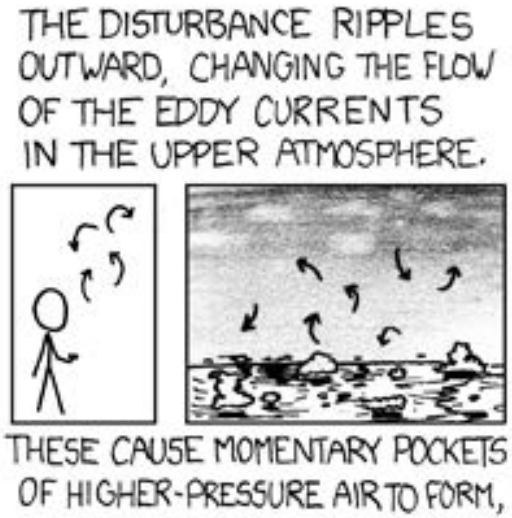

WELL, REAL PROGRAMMERS USE ed.

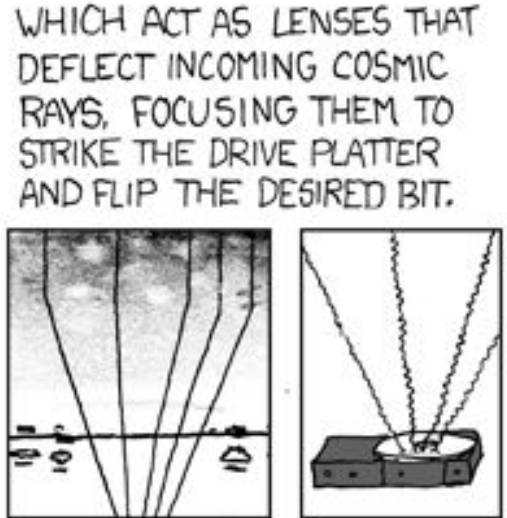

NO, REAL PROGRAMMERS USE cat.

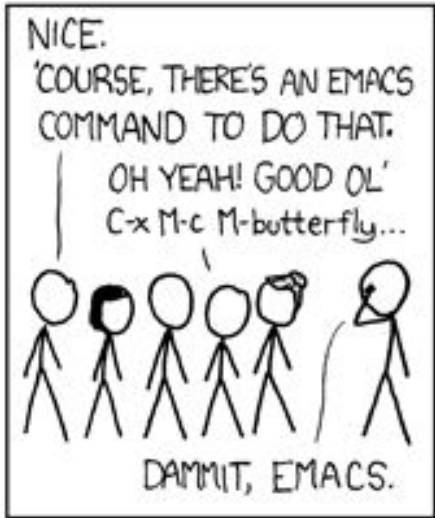

REAL PROGRAMMERS USE A MAGNETIZED NEEDLE AND A STEADY HAND.


EXCUSE ME, BUT REAL PROGRAMMERS USE BUTTERFLIES.


THEY OPEN THEIR HANDS AND LET THE DELICATE WINGS FLAP ONCE.


THE DISTURBANCE RIPPLES OUTWARD, CHANGING THE FLOW OF THE EDDY CURRENTS IN THE UPPER ATMOSPHERE.
THESE CAUSE MOMENTARY POCKETS OF HIGHER-PRESSURE AIR TO FORM,


WHICH ACT AS LENSES THAT DEFLECT INCOMING COSMIC RAYS, FOCUSING THEM TO STRIKE THE DRIVE PLATTER AND FLIP THE DESIRED BIT.


NICE.
'COURSE, THERE'S AN EMACS COMMAND TO DO THAT.
OH YEAH! GOOD OL' C-x M-c M-butterfly...
DAMMIT, EMACS.

Building software via g++

- Run the program g++ and tell it which cpp files you want built
  - In simple programs, it's just that simple
  - Options to include:
    - -g (leaves in debugging symbols)
    - -Wall (enables ALL warnings)
    - -o [filename] (tells g++ what to name the final program)
    - -std=c++0x (tells g++ to use the c++0x language standard)
      - Could use -std=c++11, but it's not the one by default on the EECS servers yet
- Could be more specific and build object files (*.o), then link those
  - Great for larger programs with LONG build times.

What is make?

- A tool to help build software
- Huge supply of documentation: https://www.gnu.org/software/make/manual/make.html
- We're going to use it in an incredibly simple way
  - Most of the assignments will just copy & update the Makefile from MA1
- Comments start with #, wrapping lines with \, and commands start with a hard tab! (\t) - NOT some spaces that look the same
  - ASCII 0d9 vs ASCII 0d32

GUI IDE options

- If you've got a desktop, there's options for GUI tools
  - netbeans
  - Code::Blocks
  - KDevelop
  - Eclipse
  - CodeLite IDE
  - Geany IDE
- I will use ddd to do debugging on a GUI (we'll see more of that later)

Installing software on Linux distros: They use package managers

- The package manager for your distro is the first place for software
  - Debian (and children): apt
    - apt-get install or aptitude or synaptic
  - RedHat (and children): rpm
    - rpm -ivh
  - Arch
    - pacman
  - etc.

Graded assignment: hello world (on linux)

- Will require you to:
  - Login to an EECS SSH server
  - Create a hello world C++ source file
  - Compile the file
  - Run it
  - Take a screenshot
  - Upload the screenshot to Bb

Further questions or clarifications?