Linux Boot Camp

Jack Biggs
Sol Boucher
17 Jan 2016
Connecting

**SSH**
Windows users: PuTTY (or SSH Tectia)
Mac & Linux users: Terminal (Just type `ssh`!)
`andrewid@shark.ics.cs.cmu.edu`

**Files**
Windows, Mac, Linux users: Filezilla
`andrewid@unix.andrew.cmu.edu`
Discouraged for use with text editing - more on that later
Welcome!

$ ls
$ cd private
$ mkdir 15-213
$ cd 15-213
$ mv ~/Downloads/datalab-handout.tar .
$ tar xvf datalab-handout.tar
$ cd datalab
Terminal Shortcuts

- ~ is an alias to your *home directory*.
  - **Ex:** `cp foo.txt ~`
- . is an alias to your *present directory*.
  - **Ex:** `cp ~/foo.txt .`
- .. is an alias to the *parent directory*.
  - **Ex:** `cp ~/foo.txt ..`
- * will match as many characters as it can.
  - **Ex:** `cp ~/*.txt .`
  - **Ex:** `objdump -d *`
  - **Ex:** `rm *.c` *(be very very very very careful!!)*
    - There is no trash with `rm`. It is *gone.*
More Terminal Shortcuts

- Pressing tab will autocomplete filenames.
- Use the up+down arrow keys to scroll through your previous commands.
- Control+R lets you search your command history.
- Control+A jumps to the beginning of the line.
- Control+E jumps to the end of the line.
- Control+U clears everything to the left of the cursor.
- Control+C kills your current program.
- Control+D (on a blank line) exits the terminal.
- Control+L clears your screen.
Fancy Terminal Shortcuts

- Bash automatically splits things up in brackets!
  - Ex: `cp foo{1,2}.txt = cp foo1.txt foo2.txt`
  - Ex: `cp foo.txt{,.bak} = cp foo.txt foo.txt.bak`
- For when typing the same filename gets annoying
- Bash has `for` loops!
  - Ex: Append “15-213” to every file ending in .c
    
```
    for file in *.c; do echo "15-213" >> $file; done
    ```
- Have fun, but don’t break things or lose track of time
ls <dir>

- Lists the files in the present working directory, or, if specified, `dir`.
- `-l` lists ownership and permissions.
- `-a` shows hidden files ("dotfiles").
- `pwd` tells you your present working directory.
cd <directory>

- Try running `cd -` to return to the previous directory.
- Try running `cd ..` to return to the parent directory.
- Changes your present working directory.

```
jbiggs@blueshark ~ $ ls
cover_letter.pdf factorial.py Movies resume.pdf test.wav
demo.py foo2.py Music school timer.py
Desktop foo.txt Pictures solutions.py www
display.py Fravic.pdf private src
Documents Library Public Templates
Downloads Minecraft.jar
jbiggs@blueshark ~ $ cd private/
jbiggs@blueshark ~/$
mkdir <dirname>

- Makes a directory `dirname` in your present working directory.
- Directories and folders are the same thing!
mv <src> <dest>

- cp works in exactly the same way, but copies instead
  - for copying folders, use `cp -r`
- `dest` can be into an existing folder (preserves name), or a file/folder of a different name
- `src` can be either a file or a folder

```
jbiggs@blueshark ~ $ cd private/
jbiggs@blueshark ~/private $ mkdir 15-213
jbiggs@blueshark ~/private $ cd 15-213
jbiggs@blueshark ~/private/15-213 $ mv ~/Downloads/datalab-handout.tar .
```
tar <options> <filename>

- For full list of options, see man tar
- tar stands for tape archive. Was used on tapes!
- x - extract, v - verbose, f - file input
- All of our handouts will be in tar format.
Also, `rm <file1> <file2> ... <filen>`

- To remove an (empty) directory, use `rmdir`
- To remove a folder and its contents, use `rm -rf`
- Please be careful, don’t delete your project.
- There is no “Trash” here. It’s gone.
- Contact `ugradlabs@cs.cmu.edu` to restore.
- Latest restore is up to a day old!
What’s in a file? (using `grep`)

- `grep <pattern> <file>` will output any lines of file that have `pattern` as a substring
- `grep -v` will output lines *without* `pattern` as substring
- `grep -n` prints line numbers
- `grep -R` will search *recursively*

**Try it:**
- `grep 'phase' bomb.c`
- `grep -v -n 'printf' src.c`
- `grep -R 'unsigned' .`
pipes and redirects

- A pipe redirects output from one program as input to another program.
  - Ex: `objdump -d bomb | grep "mov"`
  - Ex: `ls *.c | grep malloc`
  - Ex: `ls -l | grep jbiggs | wc -l`

- Can redirect output to a file.
  - Ex: `cmd arg1 ... argn > file.txt` will write the output of `cmd` over `file.txt`.
  - Ex: `cmd arg1 ... argn >> file.txt` will append the output of `cmd` to `file.txt`. 
Looking for something? `grep -A -B`

- `grep -B <x>`: include x lines **Before** match.
- `grep -A <y>`: include y lines **After** match.
- **Ex:** `objdump -d | grep -A 25 explode_bomb`
- **Ex:** `grep -B 20 return *.c`
What’s in a file? (using `cat`)

- `cat <file1> <file2> ... <filen>` lets you display the contents of a file in the terminal window.
- Use `cat -n` to add line numbers!
- You can *combine* multiple files into one!
  - `cat <file1> ... <filen> >> file.txt`
- Good for seeing what’s in small files.
- Try `cat -n bomb.c`. Too big, right?
What’s in a file? (using **less**)

- **less** `<file>` will give you a scrollable interface for viewing large files **without** editing them.
  - To find something, use `/`
  - To view the next occurrence, press `n`
  - To view previous occurrence, press `N`
  - To quit, use `q`
- Try it: Type “/phase”
man <thing>

- What is that command? What is this C standard library function? What does this library do?
- Pages viewed with less
- Try it!
  - man grep
  - man tar
  - man strlen
  - man 3 printf
  - man stdio.h
  - man man
Editors (a touchy subject)
Editors (a touchy subject)

- **vim** is nice, made for very powerful text editing
  - Try running `vimtutor` to get started learning
- **emacs** is nice, made to be more versatile
  - Definitely do the emacs tutorial in emacs, “ctrl-h t”
- **gedit** has a GUI, but requires X Forwarding setup. Too platform-dependent to show here, sadly.
- I **strongly** recommend editing on the terminal.
- **Gist**: Use an editor with auto-indent and line numbers
**screen**

- Run simultaneous programs in different “tabs”
- `<Control-a>`, then press c: create new tab
- `<Control-a>`, then press k: kill current tab
  - Consider exiting bash rather than killing window (bad)
- `<Control-a>`, then press n: go to next tab
- `<Control-a>`, then press p: go to previous tab
- `<Control-a>`, then press `<number>`: go to tab `<number>`
- `<Control-a>`, then press a: send “Control-a” to window
- `<Control-a>`, then press `?`: help
- All other shortcuts stay, **screen** only binds to `<Control-a>`
Editors (if you really really just want a GUI)

- Simple answer: Go to a Linux cluster on-campus, open a terminal, and run:
  
  ```bash
  ssh -Y andrewid@shark.ics.cs.cmu.edu
  ```

- Now you can run `gedit <filename> &`
- `& forks` your process into the background so you can use the prompt without waiting for `gedit` to finish
Editors (if you really, **really** just want a GUI)

- Not-so-simple answer: Google “How to install X Forwarding on <platform>”
  - Mac: You need XQuartz
  - Windows: You need XMing and PuTTY
- This allows you to execute GUI applications on the shark machines, but have the GUI appear on your computer.
A word about editing locally and using (S)FTP

- We heavily discourage this.
- It is a pain.
- You will waste time.
  - Edit the file
  - Save the file
  - Upload the file
  - FTP: “Do you want to replace this file?”
  - Every single time! (╯°□°)╯︵ （￣︶￣）
- You will likely have to have a console on the shark machines open for gdb and compilation anyway.
  - Use screen!
Commands related to 15-213

- **gdb**, the **GNU Debugger**, will be used for bomb lab.
- **objdump** displays the symbols in an executable.
- **gcc** is the **GNU C Compiler**.
- **make** is a configurable build system often used for compiling programs.
- We will provide other tools in the handouts as well.
INTERESTED IN UPDATING YOUR ANTI-VIRUS SOFTWARE?

OH, I WOULDN'T NEED ANY OF THAT—

I RUN LINUX.

FLIP
Vim Tutorial

- Basics (Quick vimtutor walkthrough)
- Splits & Tabs
  - Splitting the same file
- Specific, useful shortcuts (gd, %, null register, indent)
- Visual mode
- Find
  - Basic Regular Expressions
- Find-and-Replace
- Macros (super awesome!)
- Materials: http://cs.cmu.edu/~213/recitations/bootcamp.zip
Git Tutorial

- **GitLab hosting:** [https://git.ece.cmu.edu/](https://git.ece.cmu.edu/)
- `git clone git@git.ece.cmu.edu:<andrewid>/<repo>`
- `git status`
- `git add`
- `git commit -a`
- `git push`
- `git log`
- `git rm`
- `git diff`
- `git pull`
- Visit a TA if you need help or want to do something advanced.
Resources

Quick references: cs.cmu.edu/~213/resources.html
CMU Computer Club
  - www.contrib.andrew.cmu.edu/~sbaugh/emacs.html
  - club.cc.cmu.edu/talks/fall15/power-vim.html
  - club.cc.cmu.edu/talks/fall15/power-git.html
Great Practical Ideas
  - www.cs.cmu.edu/~15131/f15/topics/bash/
  - www.cs.cmu.edu/~15131/f15/topics/git/
Official manuals
  - info bash / info emacs
  - :help in Vim
  - git help