Linux Boot Camp

Jack, Matthew, Nishad, Stanley
6 Sep 2016
Connecting

**SSH**

Windows users: MobaXterm, PuTTY, SSH Tectia

Mac & Linux users: Terminal (Just type `ssh`)

`andrewid@shark.ics.cs.cmu.edu`
Let’s Bash!

Log in to the Shark machines and run:

```bash
$ echo $0
-bash (make sure this line is correct)
```
Let’s Bash!

Log in to the Shark machines and run:
$ echo $0
-bash (make sure this line is correct)

Not the same? Connect to the Andrew machines (unix.andrew.cmu.edu, NOT Shark) and run:
$ chsh -s /bin/bash
- Log out, then log back into the Shark machines
I Need You To Make A Directory

$ ls
$ cd private
$ mkdir 15-213
$ cd 15-213
FileZilla / File Transfers

- Download datalab-handout.tar from Autolab
- Use scp, or download and install FileZilla
  https://filezilla-project.org/
  - Host: shark.ics.cs.cmu.edu
  - Username: (your Andrew ID)
  - Password: (your Andrew ID Password)
  - Port: 22
- Navigate to 15-213 folder, then drag and drop file
- Same way in reverse to download file to submit

Detailed guide: http://cs.cmu.edu/~213/recitations/using_filezilla.pdf
$ ls
$ cd private
$ mkdir 15-213
$ cd 15-213
$ tar xvpf datalab-handout.tar
$ cd datalab-handout
Terminal Shortcuts

The command line operates on one directory at a time (the “working directory”).

You can use these shortcuts whenever a directory or file path is expected.

<table>
<thead>
<tr>
<th></th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>~</td>
<td>Home directory</td>
<td>cp foo.txt ~</td>
</tr>
<tr>
<td>.</td>
<td>Working (current) directory</td>
<td>cp ~/foo.txt .</td>
</tr>
<tr>
<td>..</td>
<td>Parent directory</td>
<td>cp ~/foo.txt ..</td>
</tr>
<tr>
<td>-</td>
<td>Previous directory</td>
<td>cd -</td>
</tr>
<tr>
<td>*</td>
<td>Match as many characters as possible</td>
<td>cp ~//*.txt</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rm *.c</td>
</tr>
</tbody>
</table>

- Be very very very careful with rm!!
- There is no trash with rm. It is gone.
More Terminal Shortcuts

- Pressing tab will autocomplete file/directory names.
- 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.
ls <dir>

- Lists 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.

```
ls:
[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 Public  test.py
```

```
pwd:
[jbiggs@blueshark ~]$ pwd
/afs/andrew.cmu.edu/usr10/jbiggs
```

```
[jbiggs@blueshark ~]$ 
```
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.

![Directory Tree Example](image-url)
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, p - keep perms
- All of our handouts will be in `tar` format.

```
jbiggs@blueshark ~/private/15-213 $ tar xvf datalab-handout.tar
datalab-handout/
datalab-handout/bits.c
datalab-handout/Makefile
datalab-handout/README
datalab-handout/btest.h
datalab-handout/btest.c
datalab-handout/bits.h
datalab-handout/decl.c
datalab-handout/tests.c
datalab-handout/fshow.c
```
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 -n ‘printf’ src.c`
- `grep -R ‘unsigned’`
pipes and redirects

- A *pipe* redirects output from one program as input to another program.
  - **Ex:** `ls *.c | grep malloc`
  - **Ex:** `ls -l | grep jbiggs | wc -l`
- Can *redirect* output to a file.
  - **Ex:** `echo hello > file.txt` writes “hello” over `file.txt`.
  - **Ex:** `echo hello >> file.txt` *appends* “hello” 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: Open your datalab file, search for strings
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)

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.

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....

DAMNIT, EMACS.
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
  - Emacs tutorial in emacs: “Ctrl-h t”
- **gedit** has a GUI
  - Requires X Forwarding: See Appendix
- **I strongly** recommend editing on the terminal.
- **Gist**: Use an editor with auto-indent and line numbers
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 ANTIVIRUS SOFTWARE?

OH, I WOULDN'T NEED ANY OF THAT -

I RUN LINUX.

FLIP
Vimtutor Walkthrough

■ Chapters 1-3
■ Cheatsheet: http://bit.ly/2c1O1J0
Sublime Text / Atom!

http://cs.cmu.edu/~213/recitation/using_sublime.pdf
Resources

Ask the Course Staff!

http://cs.cmu.edu/~213/help/
Resources

■ Quick references: [cs.cmu.edu/~213/resources.html](cs.cmu.edu/~213/resources.html)
■ CMU Computer Club
  ■ [www.contrib.andrew.cmu.edu/~sbaugh/emacs.html](www.contrib.andrew.cmu.edu/~sbaugh/emacs.html)
  ■ [club.cc.cmu.edu/talks/fall15/power-vim.html](club.cc.cmu.edu/talks/fall15/power-vim.html)
  ■ [club.cc.cmu.edu/talks/fall15/power-git.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/bash/)
  ■ [www.cs.cmu.edu/~15131/f15/topics/git/](www.cs.cmu.edu/~15131/f15/topics/git/)
■ Official manuals
  ■ info bash
  ■ info emacs
  ■ :help in Vim
Appendix
Editors (if you really really just want a GUI)

- Simple answer: Go to a Linux cluster on-campus, open a terminal, and run:
  
  ```
  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.
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
    ```bash
    for file in *.c; do echo “15-213” >> $file; done
    ```
- Have fun, but don’t break things or lose track of time
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>