CS197U: A Hands on Introduction to Unix

Lecture 2: Getting, Editing, and Manipulating Files

J.D. DeVaughn-Brown
University of Massachusetts Amherst – Department of Computer Science
Reminders

• Assignment 1 was due TODAY
  • A few students have not turned it in
  • Remember you need to submit 6 assignments to pass
  • Come to the Edlab for help

• Occasionally the edlab servers break
  • Or they go down for maintenance
  • Try logging on to a different machine
  • If you notice problems email me and I’ll try to get them fixed

• If you need help, email me  jddevaughn@cs.umass.edu
Reminders

• When you ssh from a Mac or between Linux machines you need to include your username if it is different on both machines
  • ssh USERNAME@HOSTNAME
  • ssh USERNAME@elnux1.cs.umass.edu

• Class mailing list
  • If you aren’t getting any emails from me, let me know and I will add you manually.
Reminders:

• Last time we:
  • Learned what Unix and Linux are
  • Used ssh to access a Linux system
  • Moved around through some folders

• Describe to one or two people around you the command(s) to:
  • Move to your home directory and look at what is there
  • Move back one directory and look at the path
  • List the contents of the current directory with extra details
  • Connect to a remote host and change your password
Answers

• Move to your home directory and look at what is there
  • cd; ls

• Move back one directory and look at the path
  • cd ..; pwd

• List the contents of the current directory with extra details
  • ls -l

• Connect to a remote host and change your password
  • ssh user@hostname; passwd

You can use a semi-colon between commands to enter more than one at a time
Using the Command Line

• A situation where I use the command line:
  • My primary machine is a laptop
  • My research lab has a powerful server that is in a server room somewhere on campus and we don’t have direct access to them
  • I can use an ssh client on my machine to log in to the server
    • I can write code, store data and run experiments just like I would on my machine, but everything runs much faster
  • Other people can log in at the same time and do the same thing
Some command line tips

• A reminder in case you forgot…

• Press `<tab>` to “auto complete” a program, file, or folder name

```
$ elnux7> cd /courses/cs100/cs19
$ cs191a.lehnert cs191p/ cs192s/
$ cs197c/ cs197u/
$ elnux7> cd /courses/cs100/cs197u
```

• Displays possible completions if multiple options

• Or completes directory/file name if only one

• Type `history` to show a list of commands you have run recently

• Press `<ctrl-c>` to cancel what you have typed, or quit (some) programs
Reading the **man**ual

- To learn more about a command, type `man [command]`
  - `man` uses `less` to display the help info to you
  - Any shortcut keys you learn for `less` (ie `q`), will apply here as well!

Commands can have a lot of flags!

Use `man` instead of trying to memorize them all!
man is powerful BUT confusing...

- In this class you can ask me for help
  - but some Linux user forums are less friendly

- So check the man page and see if it answers your question first!

- Man pages can be a bit overwhelming
  - but there is a lot of useful info
Today

• This time we will:
  • Manage files and directories
  • Use a text editor
  • Learn about utilities for working with files
  • Learn how to combine multiple commands
  • Some useful tools

• This class is designed to introduce you to tools you might need in the future - if you don’t understand how something works or how/why you would use it - Please Ask!
Managing files

• One of the main things you’ll be doing on a Linux system is:
  • Creating, Editing and Organizing files
• Files in Linux are stored in a directory tree
  • Just like Windows
• Each user has a “home directory”
  • You have permission (from the OS) to read and write files there
Managing files

• Commands for basic file management:
  • `cp <file> <dest>`  # Copy <file> to <dest>
  • `cp -r <folder> <dest>`  # Recursively copy a folder
  • `mv <source> <dest>` # Move a file/folder from <source> to <dest>
    • Note: This is how you rename a file
  • `rm <target>`  # Remove a file (careful -- no recycling bin here!)
Working with directories

• Change between directories using `cd <dirname>`
  • `cd ..` # move up a directory
• The forward slash (/) is used to separate directory names
  • `cd ../../../` # move up two directories
  • `cd /courses/cs100/cs197u`
• To create or delete a directory:
  • `mkdir <folder>` # Creates a new directory
  • `rmdir <folder>` # Delete an empty directory
  • `rm -r <folder>` # Remove a folder and all subfolders/files
Special symbols

• The Linux shell provides some special symbols to make things easier:

• **Asterisk (*)** - acts as a wildcard matching folder or file names
  
  - `cp notes*.txt folder/`  # Copies notes1.txt, notes2.txt, notes-xyz.txt, ...

• **Dot (.)** - represents the current directory

• **Dot Dot (..)** - represents the parent directory

```bash
elinux7>pwd
/home/twood/folder-1
elinux7>ls ..
folder-1/  folder-2/
elinux7>cp * ../../../../twood/
```

*list contents of the parent directory*

*copy all files into folder*
ICE time

• PAIR UP!
  • Navigate to your home directory
  • Create a directory named test
  • Create 3 text files (test1.txt, test2.txt, test3.txt)
  • Copy them all one directory up (1 command)
  • Delete the files you just copied WITHOUT navigating there
  • move the files in your test directory one directory up
  • Delete the now empty directory
Today

• This time we will:
  • Manage files and directories
  • **Use a text editor**
  • Learn about utilities for working with files
  • Learn how to combine multiple commands
  • Some useful tools
Editing files with nano

- Text editors: probably the most common utility you will use
  - In Windows or Mac, you might open TextEdit, Notepad or Microsoft Word
  - On a Linux system there are other text editing programs that will open in your terminal window
    - Lots of options: **emacs** and **vim** are most popular
      - Read the book to learn more about **emacs** or **vim**
    - We will cover these briefly later in the class
    - We will use **nano** because it is simpler
- To open the editor type: **nano**  <filename>
  - Editor program will open in your terminal window
  - The file does not have to already exist
  - If you type the command **nano** with no filename, the editor will ask you to enter a filename when you choose to save
nano

This is the text area.
I can write words here... how neat!

At the bottom of the screen are the commands I can use.
The "^" symbol represents the "ctrl" key

"^" = CTRL key

Text area

Command list
Important $\texttt{nano}$ commands

- $\texttt{<ctrl}-\texttt{X}$ = Exit
- $\texttt{<ctrl}-\texttt{O}$ = WriteOut (save the file)
- $\texttt{<ctrl}-\texttt{R}$ = Read File (open a new file)
- $\texttt{<ctrl}-\texttt{W}$ = Where Is (search within a file)

Don’t use capital letters for key shortcuts
Today

• This time we will:
  • Manage files and directories
  • Use a text editor
  • **Learn about utilities for working with files**
  • Learn how to combine multiple commands
  • Some useful tools
cat - printing out files

• Use to display the full contents of a file to the screen

```bash
> cat days.txt
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
```
**head and tail - for parts of files**

- Print out the top or bottom of a file
- Use `head -n X` to print out the top X lines

```bash
> head -n 3 days.txt
Monday
Tuesday
Wednesday

> tail -n 2 days.txt
Saturday
Sunday
```

with no "-n", default is 10 lines
less is more - reading long documents

- Sometimes you need to look through a very long text file
  - Using cat is impractical (will scroll to bottom and is sometimes slow)
  - Could use a text editor, but then you might accidentally break things
- Solution: use `less` or `more`
  - Use arrow keys to scroll up and down
  - Press `<space>` to jump forward a page
  - Press `<q>` to quit
- Syntax: `less [really-long-file]`
clear – clear the screen

• As you work on the terminal, your screen fills up
  • with commands and output
  • Use clear to clear the screen

```bash
>head -n 3 days.txt
Monday
Tuesday
Wednesday

>tail -n 2 days.txt
Saturday
Sunday

>clear
```
ICE #2

• Same partners:
  • Create a text file with your 5 favorite pokemon (original 150)
  • Print the file using two different commands
  • Print the first two pokemon from the file
  • Print the last two pokemon from the file
  • Reset your screen
sort - sorting files

• Use to sort files alphabetically or numerically

```bash
$ sort days.txt
Friday
Monday
Saturday
Sunday
Tuesday
Thursday
Wednesday
```

• Warning! To sort lists of numbers, use:
  
  ```bash
  $ sort -n numberfile.txt
  ```
grep – filtering a file

• Use to find lines in a file that match a string (or regular expression)

> grep “Friday” days.txt
Friday

> grep “BANANA” days.txt

• Prints only the lines in the file that match the input
• Or no output if no matches
• Count number of matching lines for a string in a file

> grep -c “Friday” days.txt
1
grep – some more flags

• Find lines that don’t contain a string
  ```
  grep -v “Friday” days.txt
  Monday
  Saturday
  Sunday
  Tuesday
  Thursday
  Wednesday
  ```

• Grep for case-insensitive string
  ```
  grep -i “Friday” days.txt
  FRIDAY
  Friday
  FRiDaY
  ```
Saving output to a file

• “Redirect” output of a command to a file
  • Useful for commands that produce many lines of output
  • Save results for later, or to use with another command

• Syntax: `[command] > [filename]`

• Warning! This will REPLACE any file with the same name

• To APPEND to a file, use `>>`
  • `[command2] >> [filename]`

```bash
sort days.txt > sorted.txt
head -n 3 sorted.txt
Friday
Monday
Saturday
```
Today

• This time we will:
  • Manage files and directories
  • Use a text editor
  • Learn about utilities for working with files
  • Learn how to combine multiple commands
  • Some useful tools
Pipes - combining multiple commands

- Pipes allow you to combine multiple commands

- Syntax: `[command1] | [command2]`

- Example:
  
  Sort a file, then print the top 3 entries

  ```
  sort days.txt | head -n 3
  ```

- The output of the first command is the input of the second command

*DIFFERENT THAN USING A SEMICOLON!*
uniq – removing duplicate entries

• Often, you want to remove duplicate entries from a file

• Use uniq together with sort
  
  • uniq - removes identical adjacent lines
  
  • Must sort the file before applying uniq

  • sort <file> | uniq

> sort days.txt
  
  Friday
  Monday
  Saturday
  Saturday
  Sunday
  Tuesday
  Thursday
  Thursday
  Wednesday

> sort days.txt | uniq
  
  Friday
  Monday
  Saturday
  Sunday
  Tuesday
  Thursday
  Wednesday
Today

• This time we will:
  • Manage files and directories
  • Use a text editor
  • Learn about utilities for working with files
  • Learn how to combine multiple commands
• Some useful tools
Password-less SSH login

- Connect to remote hosts without entering your password every time
  - e.g., connect to elnux machines in Edlab
- From a Unix shell only (Linux or Mac OS X)
  - Create SSH keys
    - run ssh-keygen -t rsa
    - Enter nothing for the filename
    - copy ~/.ssh/id_rsa.pub to ~/.ssh/authorized_keys on remote host
      - scp ~/.ssh/id_rsa.pub username@elnux1.cs.umass.edu:~/.ssh/authorized_keys
  - ~ is short for your home directory
- From Windows
  - Different procedure for putty
Search for a command in history

• Use UP and DOWN keys in terminal
  • Takes too long to find a command used a long time ago

• history | less
  • Use /<search_string>, type `n` to go to next
  • copy command
  • Good, but still slow

• history | grep "<search_string>"
  • Use !<number> to execute. Note: <number> is the number listed to the left of the command.

• Use reverse incremental search (in bash shell only)
  • Ctrl + r, start typing command
  • Type Ctrl + r again to find next match
  • Fast!
Assignment 2

• Will use many of the commands covered today, plus a few new ones

• Due **next Thursday at 3:00 pm**
  • Remember to email me if you want to skip this assignment
  • I highly discourage skipping this one

• I’ll be in Edlab before class
# Lecture 2 review

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cp, mv, rm</td>
<td>copy, move, and delete files</td>
</tr>
<tr>
<td>nano</td>
<td>simple text editor</td>
</tr>
<tr>
<td>cat</td>
<td>print files to console</td>
</tr>
<tr>
<td>head, tail</td>
<td>print tops and bottoms of files</td>
</tr>
<tr>
<td>sort</td>
<td>sort files</td>
</tr>
<tr>
<td>uniq</td>
<td>Remove duplicate adjacent lines</td>
</tr>
<tr>
<td>less, more</td>
<td>view long files</td>
</tr>
<tr>
<td>man</td>
<td>provide help about commands</td>
</tr>
<tr>
<td>&gt;, &gt;&gt;</td>
<td>Write command output to a file</td>
</tr>
<tr>
<td></td>
<td>Send command output to another command</td>
</tr>
</tbody>
</table>