

Week 3

Diff and Patching

17 October 2018

CS 35L Lab 4

Jeremy Rotman

Announcements

- Assignment #2 is due TODAY! by 11:55pm
- For Assignment #10
 - ◆ You should begin to choose stories
 - ◆ Email me to tell me what you are choosing
 - ◆ [Here is the link to see what stories people have signed up for already](#)
 - ◆ [Here is the link to sign up to present](#)
 - Please sign up to present by Friday Oct 26

Questions?

Outline

- Assignment #2 Hints
- diff files
- Patches

Assignment 2 hints

- `sed -n 'first_line~step_sizep'`
 - ◆ `p` commands `sed` to print the specified line(s)
 - ◆ `-n` suppresses other output
 - ◆ `first_line` is the line number you wish to begin from
 - ◆ `step_size` is the number of lines to step to reach next printed line
- If you want to create a file with a leading “-” (for testing)
 - ◆ Include the path before the file name
 - E.g. `touch ./-newfile`

How to read diff output

- When you run normal diff
 - ◆ Outputs differences without context
- < indicates the lines that are unique to file 1
- > indicates the lines that are unique to file 2

change-command

```
< from-file-line  
< from-file-line  
---  
> to-file-line  
> to-file-line
```

How to read diff output

→ change-command has one of 3 formats

◆ “*lar*”

- Add lines in range *r* from the 2nd file, starting after line *l* in the 1st file

◆ “*fct*”

- Change lines in range *f* from the 1st file to range *t* from the 2nd file

◆ “*rdl*”

- Delete the lines in range *r* from the 1st file, line *l* is where they would have appeared in the 2nd file if not deleted

change-command

< *from-file-line*

< *from-file-line*

> *to-file-line*

> *to-file-line*

Unified Diff Output

- If you use the -u option, it creates the unified diff output
- Unified diff output gives us more context to see differences
 - ◆ It shows you the lines from both files
 - In the hunks that differ

Unified Diff Output

```
--- from-file from-file-modification-time  
+++ to-file to-file-modification-time
```

```
@@ from-file-line-numbers to-file-line-numbers @@
```

```
Line-from-either-file
```

```
Line-from-either-file
```

```
+Line-added-to-first-file
```

```
-Line-removed-from-second-file
```

```
Line-from-either-file
```

Unified Diff Output

- Note that the first file is “-” and the second file is “+”
- Another way to differentiate is to match the symbol in the header to lines
 - ◆ The file that has +++ in front of it in the header, has the lines with a + in front of them

An Example: Two files

lao

The Way that can be told of is not the eternal Way;
The name that can be named is not the eternal name.
The Nameless is the origin of Heaven and Earth;
The Named is the mother of all things.
Therefore let there always be non-being,
so we may see their subtlety,
And let there always be being,
so we may see their outcome.
The two are the same,
But after they are produced,
they have different names.

tzu

The Nameless is the origin of Heaven and Earth;
The named is the mother of all things.

Therefore let there always be non-being,
so we may see their subtlety,
And let there always be being,
so we may see their outcome.
The two are the same,
But after they are produced,
they have different names.
They both may be called deep and profound.
Deeper and more profound,
The door of all subtleties!

An Example: Unified Diff Output

```
--- lao 2002-02-21 23:30:39.942229878 -0800
+++ tzu 2002-02-21 23:30:50.442260588 -0800
```

```
@@ -1,7 +1,6 @@
```

```
-The Way that can be told of is not the eternal Way;
```

```
-The name that can be named is not the eternal name.
```

```
The Nameless is the origin of Heaven and Earth;
```

```
-The Named is the mother of all things.
```

```
+The named is the mother of all things.
```

```
+
```

```
Therefore let there always be non-being,
```

```
    so we may see their subtlety,
```

```
And let there always be being,
```

An Example: Unified Diff Output

```
@@ -9,3 +8,6 @@
```

```
The two are the same,
```

```
But after they are produced,
```

```
    they have different names.
```

```
+They both may be called deep and profound.
```

```
+Deeper and more profound,
```

```
+The door of all subtleties!
```

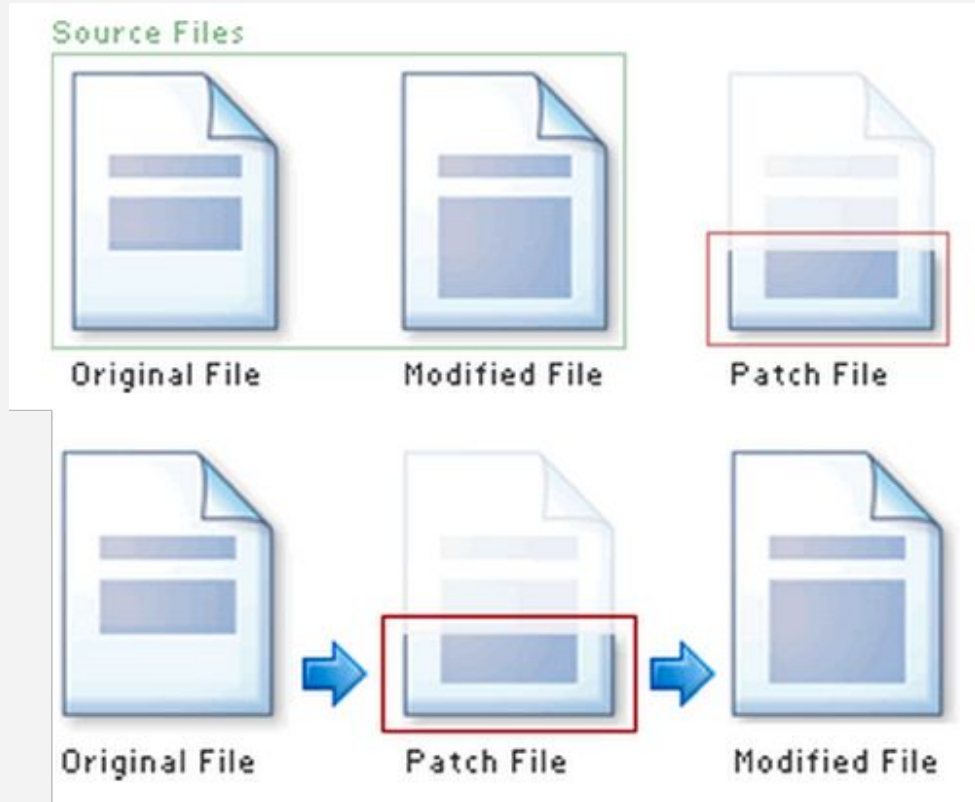
Patching

- You may be very familiar with the term if you play video games
- But what exactly is a patch?

Patching

- You may be very familiar with the term if you play video games
- But what exactly is a patch?
 - ◆ A piece of software designed to fix problems with or update a computer program
- A Patchfile is just a diff file that includes the changes made to a file
- The patch command can be used with the diff file to add changes

Applying a Patch



The patch command

→ **patch** -p num < *patchfile*

- ◆ The -p option allows you to strip the smallest prefix containing *num* leading slashes
- ◆ In particular this relates to filenames within the patchfile
- ◆ If you have a different directory for your filenames, this is important to use
- ◆ E.g. if you had the filename /u/home/j/jeremy/src/coreutils
 - -p0 gives /u/home/j/jeremy/src/coreutils
 - -p1 gives u/home/j/jeremy/src/coreutils
 - -p5 gives src/coreutils

Building coreutils

- A note for when you are working on the lab
 - ◆ Read the INSTALL file on how to configure “make”
 - Hint: Take a close look at the --prefix flag
 - ◆ When you run the configure script, make sure you are using the prefix flag
 - When you are done, coreutils should be installed in ~/coreutilsInstall
 - Or whatever you name our temporary directory

More hints for Lab 3

- To reproduce the bug make sure you are using `./ls` and not just `ls`
 - ◆ Why?
- After patching, you should run `make` to rebuild your patched file
 - ◆ Don't install, just make
- The homework section is about Python, we'll go over that next week, but you can also experiment with Python on your own