CSE427S

# Lab 0: Linux Command Line and SVN

M. Neumann

Due: TUE 3 SEPT 2019 – 1PM

No group work.
The submission for this lab needs to be done via SVN repository commit, which will be explained in the lab instructions below.
The completion of this tutorial counts towards your **lab quiz score** – remember lab quizzes count 10% towards your total course performance.
Thanks to Angelina Lee and Anne Bracy for sharing this tutorial.

## Preparation

Install the **course virtual machine** (VM) following the instructions linked form the course webpage.

## Introduction

The purpose of this lab is to get you up to speed on some basic class tools, namely:

- **The Linux Command Line**: Linux is a free, open source OS that has most of its identity derived from the precursor Unix family of operating systems. Our course virtual machine (VM) will run Linux. Like other operating systems, Linux provides a desktop environment; however, in this class you will learn to love the command line interface. Say goodbye to your mouse ... We're not in Kansas anymore. (We're in Missouri!)
- **SVN**: short for subversion, or svn, (see also `http://subversion.apache.org`). This course relies on SVN to distribute and collect your homework assignments. Rather than using subversion via a GUI-based interface like Tortoise SVN, we will call our SVN commands exclusively from the Linux command line from within the course VM.

This lab is a tutorial introducing the Linux command line. Working on the command line in the Linux course VM will be essential for all labs and homework assignments.

## 1 Explore commands for file system navigation

In the examples below, the > character is the prompt issued by the shell. The shell is the program that reads and interprets the commands provided by the user.
Issue the command
```
> pwd
```
which shows you your *current working directory*. Remember this command for the future, so that you can use it to check which directory you are in once we start moving to different locations or even file systems. For now, your working directory will be your *home directory* (*e.g.*, `/home/cloudera/`) when you first log in. In this case, issue the command
```
> cd ~
```
will bring you to your home directory. You can also simply use `> cd` but keep in mind that ~ can also be used to refer to your home directory in a path. For the rest of this lab we indicate that you are in your home directory as follows:
```
home>
```
The command
```
home> ls
```
provides a listing of the files and subdirectories in the current working directory.
There is another filesystem location we will use in this course which is the *root directory*. You can access it with the command
```
> cd /
```
Issue another `pwd` to verify that you are in the root directory and also another `ls` to see what's in this directory. Among other subfolders you will spot `home`, which is – no surprise – your home directory. Make sure you understand the difference between the following terms, since we will be referring to those in the lab instructions for this course:

- *working directory*
- *home directory*
- *root directory*

## 2 Explore commands for directory creation, etc.

Go back to the home directory. The command
```
home> mkdir tmp_dir
```
makes a new directory (within the current directory) called `tmp_dir`. Observe that this new directory is present with an `ls` command. This is the directory you will use for this course for the rest of the semester.
The command
```
home> rmdir tmp_dir
```
removes the directory if it's empty (which it is right now). If the directory is not empty, you will get an error message, and the directory won't be removed (consider that a safety guard).
Move to the new directory with `cd`, as follows:
```
home> cd tmp_dir home/tmp_dir>
```
An `ls` command should show that the new directory is empty. You can make an empty file in the current directory using `touch`:
```
home/cse427s> touch newfile
```
An `ls` now confirms that the file `newfile` is present.

We can clean up after ourselves by removing the file we have just created.
```
home/cse427s> rm newfile
```
or simply go up one level, and remove the whole thing entirely: `home/tmp_dir> cd ..` `home> rm -rf tmp_dir`
Note that the `..` is a short hand for the parent directory — you can do `cd ..` in any directory, and it basically brings you one directory up. The `rm -rf` removes the directory even though it is not empty.
The `rm -rf` will remove a non-empty directory — use man page to checkout what the `-r` and `-f` mean:
```
home/cse427s> man rm
```
Also checkout commands such as `mv`, `cp`, and `less` — to learn what a particular command do. Remember, `man` is your friend. Any command line supported by linux, or any standard C library routines can be found under the man page.

## 3 Check out lab0 directory

Each student in CSE427s has their own svn repository, which is accessible only by you, the instructor, the TAs, and a few technical support folks in the department. Your repository name will be your `<wustlkey>/cse427s_fl19`. Replace `<wustlkey>` with your wustlkey. Remove the angles `< >` .

### 3.1 Check out the SVN repo on host

If you have not checked out your repository on your *host machine* (this is your computer/laptop running the VM – not the *virtual machine* itself), run the following command in the terminal of your host machine:
```
> svn checkout https://svn.seas.wustl.edu/repositories/<yourwustlkey>/cse427s_fl19 cse427s
```
The first string after `svn checkout` is the URL for your repository, and the second string indicates the directory name you want to use when you check out the repository.

### 3.2 Check out the SVN repo in VM

If you want to checkout your SVN repository inside your VM, you will need to install subversion before executing the above command. To install subversion execute the following command in the terminal:
```
> sudo yum install subversion
```
(if you are asked for a password use *cloudera*)

Now, you can repeat the `svn checkout command` from above in the *home directory* of your VM. If you are prompted for a username and password in order to access you SVN repository, you should provide your wustlkey username and password. **This process will create a directory named cse427s.**

A few comments:

1. If you are asked whether to accept the site certificate for the server, go ahead and hit `p` for permanently.

2. If you are asked whether to store your svn password unencrypted, say **no**.

   *Note: If you cut and paste this from the pdf, there is an* **underscore** *between* `cse427s` *and* `fl19` *that often goes missing.*
   *Also note: there is a space between* `cse427s_fl19` *and* `cse427s`*, where we assume you will check out the repo with the name* `cse427s`*, but really, you can name it anything you like as long as you remember it.*

### 3.3 Updating the SVN repo

If you have checked out your repository, simply step into the directory and do svn update to pull code ans supporting materials for new homework and lab assignments:

```
home> cd cse427s
home/cse427s> svn up
```

This command will pull everything committed to your repository since you last updated – which means, a copy of `lab0` should now appear in your svn repository. Note that I use `home` as the path to the home directory. This path will most likely be longer for you.

## 4 More useful terminal commands

```
home/cse427s> ls
```

you will see some items including one called `lab0`. Is it a regular file, or is it a subdirectory? We can tell with the command

`home/cse427s> ls -l` (this is a lower case L, not a 1)

Which shows us the long form of the directory listing. The first character on the line is a "`d`" if the file is a directory. The next 9 characters show read, write, and execute privileges for the owner, group, and world. You are the owner, you can learn what groups you are in with the `groups` command (go ahead, type "groups" and hit return), and world is everyone. For normal files, execute privileges mean that the file can be executed (*i.e.*, it is a program), just like one would expect. For directories, they indicate permission to navigate the directory (*i.e.*, permission to issue an `ls` command within the directory).
`lab0` contains everything you need to complete the assignment. Move to the directory `lab0` and issue an `ls` command there. You should find two files: `origFile.txt` and `reallyLongFile.txt`.
A quick viewing of a file can be accomplished with the `cat` command

```
home/cse427s/lab0> cat origFile.txt
home/cse427s/lab0> cat reallyLongFile.txt
```

Notice that using `cat` for a really long file is a bit overwhelming since the entire file floods the screen far faster than you can read it. In such cases, you should use `less`.

```
home/cse427s/lab0> less reallyLongFile.txt
```

Hit the spacebar each time you are ready to see the next page of the file. Hit `q` to quit `less` and return to the command line.
You can determine the length of a file (or files) by using the word count command, `wc`.

```
home/cse427s/lab0> wc origFile.txt
home/cse427s/lab0> wc *
```

(The star indicates that you want a word count of *all files* in that directory.)
To understand the output of this command, read the manual entry for `wc`:

```
home/cse427s/lab0> man wc
```

The `man` command opens the `wc` manual in `less`, so once again, hit the spacebar to read more of the manual and type `q` to quit `less` and return to the command line.
*Note*: although the .txt extension is common for normal text files, it is not required.
Make a copy of this file and call it `OrigFile.txt`, noting that on Linux systems, filenames are case sensitive:

```
home/cse427s/lab0> cp origFile.txt OrigFile.txt
```

The `mv` command is short for move. We can change the name of our new file using this command:

```
home/cse427s/lab0> mv OrigFile.txt answer.txt
```

The `ls` command now shows us the files: `origFile.txt`, `reallyLongFile.txt`, and `answer.txt`. The `cp` command did not remove the source file, but the `mv` command did (`OrigFile.txt` should be gone).

## 5 Adding and committing files to your SVN repository

Notice also that although `origFile.txt` is part of your svn repository, the new file `answer.txt` is not. You can observe this directly by issuing the `svn info` command:

```
home/cse427s/lab0> svn info origFile.txt
home/cse427s/lab0> svn info answer.txt
```

### 5.1 SVN add

Fix this by adding the new file to the repository.

```
home/cse427s/lab0> svn add answer.txt
```

Now try the `status` command (or `stat` for short):

```
home/cse427s/lab0> svn status
```

What happened? This command shows the file `answer.txt` with a character `A` in front of it. What this tells you is that, the file `answer.txt` has been earmarked to be added to the repository. Doing a `svn info` on the file will tell you the same thing in more details.

### 5.2 SVN commit

Although it is earmarked to be added, it hasn't actually been added. To make the repository aware of the addition, you need to issue a `commit` to commit the the changes.

```
home/cse427s/lab0> svn commit -m “adding new file answer.txt” answer.txt
```

The `-m` specifies the comment for this commit. It's always a good idea to add a comment about what you're checking in.
Now try the info command again

```
home/cse427s/lab0> svn info answer.txt
```

Notice, you have committed your changes. Also note that, for any of these commands, you can give a command multiple file names.
Another way to see what's been done to a file is to use the `svn log` command.

```
home/cse427s/lab0> svn log origFile.txt
```

This will show you the time stamp and comment for every version of your code. Incidentally, this is how we can see whether your homework was checked in on time. In fact, when we check out your repository, we just ask for the state of the repo as of the due date and time, and we won't even *see* anything that you turn in late.

## 5.3 Verifying a commit

The most recent checked-in version of your repository can always be viewed (by those with permission!) on the web at the same URL that you used to check out your repository, *i.e.*,
`https://svn.seas.wustl.edu/repositories/<yourwustlkey>/cse427s_fl19/`
After you submit your homework via svn, it is a *very* good idea to check out this URL and make sure that everything looks up to date. It is your responsibility to make sure that you submit your homework both punctually and successfully.

## 5.4 SVN help

There are many other commands that `svn` supports, and naturally we won't be able to get into all of them in this document. To see a list of commands available, do:

```
home/cse427s/lab0> svn help
```

One other very useful command once you start coding is `svn revert`. To see how a command, do:

```
home/cse427s/lab0> svn help revert
```

## 6 Linux Playtime

Now that you have played around with Linux in the context of both files and SVN, take a moment to become even more familiar with the command line. Have a look at the following cheat sheet: `https://www.cheatography.com/davechild/cheat-sheets/linux-command-line/pdf/`. That document (and many others you can find online) has a lot of information. But it's a good document. Skim through it and play around with the commands listed. Hopefully what we've covered in this document is enough to get you by, but when in doubt, look though this or other cheat sheets you find online. Likely you will find what you need. The only way to get better at command line is by spending time and trying things out. You will get a hang of it eventually.