

# Assignment: Lab 3 - Navigating the Linux File Structure

New Attempt

---

**Due** Feb 13 by 11pm      **Points** 50      **Submitting** a text entry box or a file upload

---

**Purpose:** This lab will allow the student to get a basic understanding of the file system as presented by the Unix/Linux operating system, the directory tree structure, and the basic navigation commands used by a system administrator. As a system administrator you will need to be resourceful, so feel free to Google the command, use any reference text, and/or collaborate with classmates on executing the commands and seeing what results are expected and also what options can be used with each command.

**Instructions:** Follow the steps below to navigate around your VM. Be sure you understand what each command does and when to use them as they will be used in future lab assignments. Answer any follow-up questions related to the lab and then turn in the document as an attachment to the lab assignment.

## **NAMING THE VM**

One of the duties of a System Administrator is that of making sure all servers are named in some fashion. There are two files (/etc/hostname and /etc/hosts) that will need to be edited in order to name your VM, both need the same data entered into them. Think of a name for your VM and let's name it.

Login to your VM and notice the VM name of box in the prompt. Change the default VM name by entering a new name for your VM...**DO NOT** use special characters and uppercase, keep your name lower case and short. Run the following commands to give your VM a name:

```
sudo nano /etc/hostname
```

You will probably see something similar to:

```
box <--- Change this to a name you chose
```

Save and exit the file.

Next, edit the /etc/hosts file:

```
sudo nano /etc/hosts
```

You will probably see something similar to:

```
127.0.0.1    localhost
```

```
127.0.1.1    box <--- Change this to the same name you chose above
```

# The following lines are desirable for IPv6 capable hosts

```
::1    ip6-localhost ip6-loopback
```

```
fe00::0 ip6-localnet
```

```
ff00::0 ip6-mcastprefix
```

```
ff02::1 ip6-allnodes
```

```
ff02::2 ip6-allrouters
```

Save and exit the file.

Verify you have renamed your VM by doing a VM reboot:

```
sudo reboot
```

The reboot will kill your putty connection, so exit the putty terminal. Wait a few minutes and log back in and notice the VM name in the prompt should now be your new name.

## **UPDATE AND UPGRADE YOUR VM**

One of the duties of a System Administrator is that of making sure all servers are up to date with the latest firmware and software.

`apt-get update` updates the list of available packages and their versions, but it does not install or upgrade any packages.

`apt-get upgrade` actually installs newer versions of the packages you have.

After updating the lists, the package manager knows about available updates for the software you have installed. This is why you first want to update.

Let's start by updating and then upgrade the VM:

```
sudo apt-get update
```

```
sudo apt-get upgrade
```

The upgrade may take up to 15-20 minutes depending on how many packages need to be installed.

Next, do some cleaning up of the VM:

```
sudo apt-get clean
```



```
sudo apt-get autoremove
```


Once this has finished, reboot the VM:

sudo reboot

## CREATING FILES AND DIRECTORIES

Remember to log in to your VM with the user you created in lab 1...your personal user, in which you will use for all remaining labs.

You will need to successfully transfer the [lab3.tar](https://canvas.tamu.edu/courses/135833/files/35189094/download?download_frd=1)  ([https://canvas.tamu.edu/courses/135833/files/35189094/download?download\\_frd=1](https://canvas.tamu.edu/courses/135833/files/35189094/download?download_frd=1)) file to your VM using [WinSCP](http://winscp.net/download/WinSCP-5.9.1-Setup.exe)  (<http://winscp.net/download/WinSCP-5.9.1-Setup.exe>) (or similar app) from the lab computer. You may need to refer back to lab 1 for instructions on using WinSCP.

Next, login to your VM and expand the [lab3.tar](https://canvas.tamu.edu/courses/135833/files/35189094/download?download_frd=1)  ([https://canvas.tamu.edu/courses/135833/files/35189094/download?download\\_frd=1](https://canvas.tamu.edu/courses/135833/files/35189094/download?download_frd=1)) file in your user home directory (make sure you are in your user home directory by running the `cd ~` command:

```
cd ~  
tar -xvf lab3.tar
```

You should see the files and directories expanding on your VM.

Once complete, install the `tree` application. As a system administrator, you will become efficient in doing tasks with different tools. The `tree` tool allows you to display the complete directory tree from the current working directory. You will need to install this package on your VM by doing the following command:

```
sudo apt install tree
```

Verify that the new directory `icky` was created by the previously executed `tar` command, by using the `tree` command:

```
tree icky
```

Now complete the following operations consisting of CLI navigational commands and using both absolute and relative pathing techniques learned previously:

Kale is so not a fruit, so create a new directory under `grocery-store` called `vegetable` and move `Kale` where it belongs using relative pathing.

You disagree that the music in *Guardians of the Galaxy* was bad...so move the `collector` to be under the purple infinity stone.

Who is the idiot that named a file with `shiftone11!!!` Rename the file to **the-nine**.

On second thought, *Lord of the Rings* is for nerds, Blow it away (remove the directory).

They have restrooms at the grocery store so move the **wc** to the grocery store using absolute pathing.

Looks like there are a number of cheese files that should be moved under the cheese directory under grocery store, so do so using relative pathing.

foo, moo, and tmp are not infinity stones. Move them, and all their contents, to be under yicky in a new subdirectory called **other-stuff**.

Create directories of **carrot**, **potato**, **celery**, and **green-bean** under the vegetable directory.

Create a file in each of the directories named the same (example: /home/[username]/icky/yicky/grocery-store/vegetable/carrot/carrot.txt).

Create directories called **hamburger** and **seafood** under the meat directory.

Create directories of **shrimp**, **fish**, **scallops**, **crab**, and **lobster** under the seafood directory.

Create a file in each of the directories named the same.

Rename the other-stuff directory under yicky to in a new subdirectory called **junk**.

## **EXPLORING LOG FILES**

The objective for this task is to learn where log files are in the file system structure and to observe the different types of information logged by the different services running on the VM. As a system administrator, many times you are tasked with helping to debug application programs or analyze log files for specific information.

Navigate to the /var/log:

```
cd /var/log
```

```
ls
```

Observe the contents of a few of the log files in the directory. First the auth.log file. If there is an auth.log file, look at the contents of the file and describe the log file in the lab assignment submission by using the command:

```
sudo more auth.log
```

**NOTE** - If the file is empty, do a listing of the directory and see if there is a archive file typically called auth.log.1 and then view the contents of that file.

View the boot.log. If there is an boot.log file, look at the contents of the file and describe the log file in the lab assignment submission by using the command:

```
sudo more boot.log
```

**NOTE** - If the file is empty, do a listing of the directory and see if there is a archive file typically called boot.log.1 and then view the contents of that file.

View the dpkg.log. If there is an dpkg.log file, look at the contents of the file and describe the log file in the lab assignment submission by using the command:

```
sudo more dpkg.log
```

**NOTE** - If the file is empty, do a listing of the directory and see if there is a archive file typically called dpkg.log.1 and then view the contents of that file.

View the kern.log. If there is an kern.log file, look at the contents of the file and describe the log file in the lab assignment submission by using the command:

```
sudo more kern.log
```

**NOTE** - If the file is empty, do a listing of the directory and see if there is a archive file typically called kern.log.1 and then view the contents of that file.

View the syslog. If there is an syslog file, look at the contents of the file and describe the log file in the lab assignment submission by using the command:

```
sudo more syslog
```

**NOTE** - If the file is empty, do a listing of the directory and see if there is a archive file typically called syslog.1 and then view the contents of that file.

## **CREATE A SYMBOLIC LINK**

Make sure you are in your home directory, then do the following commands:

```
cd ~
```

```
pwd
```

```
nano tcmg303symlink.txt
```

Enter two or three lines of data and then save the file.

Now create the link by running the following command:

```
ln -s tcmg303symlink.txt mytcmg303link
```

Test the link using the more command:

```
more mytcmg303link
```

You should see the data you previously entered into the tcmg303symlink.txt via the symbolic link to the file.

Notice the link using the ls command:

```
ls  
ls mytcmg303link
```

You should see the color is different (typically cyan) indicating the symbolic link.

## **CREATE A SIMPLE SHELL SCRIPT**

Write a simple shell script that asks for a name and favorite color, retrieves the system time and date, and then displays two lines from the input as follows:

First line: "Today's time and date is **date...**"

Second line: "**name** your favorite color is **color...**"

Run the script and validate it works.

## **USING SPECIAL CHARACTERS**

Make sure you are in your home directory, then do the following commands:

```
cd ~
```

Use the single arrow character to create a copy of your shell script file:

```
more yourshellscript.sh > yourshellscriptcpy1.sh
```

View the content of both files to make sure they are the same.

Use the double arrow character to append data to the copy of your shell script file:

```
echo " " >> yourshellscriptcpy1.sh
```

```
echo "# This is a backup copy of my shell script!" >> yourshellscriptcpy1.sh
```

```
echo " " >> yourshellscriptcpy1.sh
```

View the content of the copy of your shell script file and verify that the above data was appended to the bottom of the file.

Use the pipe command to join the more and grep commands and search for the string 'backup' in the copy of your shell script file:

```
more yourshellscriptcpy1.sh | grep backup
```

If all of the above commands were done correctly, one line of data should display.

## **LAB FOLLOW-UP QUESTIONS**

Answer the following questions as part of the lab assignment submission for this lab.

1. What does the tar command allow you to do?
2. What are three commands used to create a file in Unix/Linux?
3. What is important to remember about the rm command in Unix/Linux?
4. Where there any log files in the /var/log directory? If so, what command did you use to view the contents? Describe the log file content.
5. Where you able to edit the file using your symbolic link? Why?
6. What is the name of the shell script you created?
7. Were you able to make a copy of your shell script? Why or Why Not?
8. Did the data append to the copy of your shell script file? Why or Why Not?
9. Did any matching data display when using the pipe character with the more and grep commands? Why or Why Not?
10. Submit a screen shot of running the tree command on the icky directory.
11. Submit a screen shot of the output of running your shell script.

## Lab Follow-Up Questions

1. What does the tar command allow you to do?

The *tar* command creates and extract Archive files.

2. What are three commands used to create a file in Unix/Linux?

*touch [filename]*

*nano [filename]*

*> [filename]*

3. What is important to remember about the rm command in Unix/Linux?

The *rm* command is irreversible, does not usually ask for a prompt, and will not remove non-empty directories if the user does not pass the recursive option.

4. Were there any log files in the /var/log directory? If so, what command did you use to view the contents? Describe the log file content.

Using the *cd /var/log && ls* command reveals that there are log files in the /var/log directory. Entering *sudo more [filename].log* allows you to view the contents of the log file. The *auth.log* file keeps authentication logs for both successful or failed logins, and authentication processes. The *boot.log* file contains start-up messages and boot information. The *dpkg.log* file shows a history of recently installed, upgraded, or removed packages. The *kern.log* file keeps in Kernel logs and warning information (especially useful for fixing problems in custom kernels). The *syslog* file shows general messages and information regarding the system, like a data log of all activity throughout the global system.

5. Were you able to edit the file using your symbolic link?



I was able to edit my symbolically linked file by using the *nano* text editor. After running *nano mytcmg303link* and adding a new line of text, I saved and exited. I then ran *nano tcmg303symlink.txt* to see if editing the data in the symbolic link affected the data in the original file. Sure enough, the data in *tcmg303symlink.txt* matched the data in my newly revised *mytcmg303link*.

6. What is the name of the shell script you created?

The name of the shell script I created is *simple.sh* (located in the msilv204 user's home directory).

7. Were you able to make a copy of your shell script? Why or Why Not?

I was able to make a copy of my shell script using the command *more simple.sh > simple1.sh* because the *more simple.sh* part of the command read out the content of *simple.sh* while the *> simple1.sh* part of the command output the content read from *simple.sh* into the *simple1.sh* (because *simple1.sh* did not originally exist in the directory, the *>* operator created the file).

8. Did the data append to the copy of your shell script file? Why or Why Not?

The data did append to the copy of my shell script file because the *>>* operator appends the existing file, rather than overwriting the existing file like the *>* operator does.

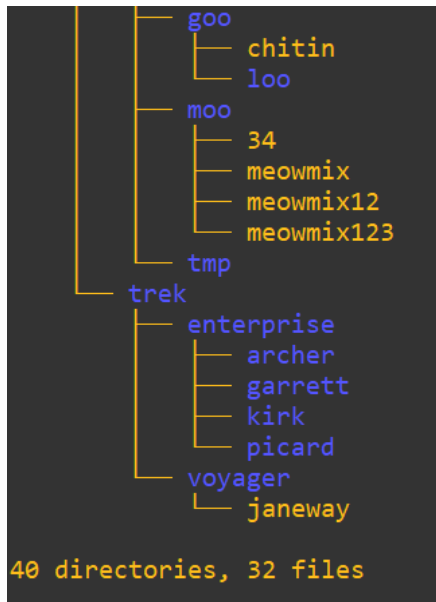
9. Did any matching data display when using the pipe character with the more and grep commands? Why or Why Not?

Matching data displayed when using a pipe character between the *more* and *grep* commands because, in Unix/Linux, the pipe character lets you use two or more commands such that output of one command serves as input to the next. This

means that the *more* command ended up feeding the text within *simple1.sh* into the *grep* utility, allowing me to search for data within *simple.sh* matching the string, “backup.”

10. Submit a screenshot of running the tree command on the icky directory.

```
[Thu Feb 10 18:06:34] [Thu Feb 10 18:06:34 msilv204@ cyber-server-vm-max:~ ] $ tree icky
icky
├── sicky
│   ├── infinity
│   │   ├── blue
│   │   │   └── tesseract
│   │   │       ├── asgard
│   │   │       └── bifrost
│   │   ├── green
│   │   ├── orange
│   │   ├── purple
│   │   │   ├── bad-music
│   │   │   └── collector
│   │   ├── red
│   │   │   └── aether
│   │   └── yellow
│   └── yicky
│       ├── grocery-store
│       │   ├── cheese
│       │   │   ├── american
│       │   │   ├── cheddar
│       │   │   └── provolone
│       │   ├── fruit
│       │   │   ├── apple
│       │   │   ├── banana
│       │   │   ├── kumquat
│       │   │   └── orange
│       │   ├── meats
│       │   │   ├── chicken
│       │   │   ├── ham
│       │   │   ├── hamburger
│       │   │   ├── lamb
│       │   │   ├── seafood
│       │   │   │   ├── crab
│       │   │   │   │   └── crab.txt
│       │   │   │   ├── fish
│       │   │   │   │   └── fish.txt
│       │   │   │   ├── lobster
│       │   │   │   │   └── lobster.txt
│       │   │   │   ├── scallops
│       │   │   │   │   └── scallops.txt
│       │   │   │   └── shrimp
│       │   │   │       └── shrimp.txt
│       │   │   └── steak
│       │   ├── ramen
│       │   ├── vegetable
│       │   │   ├── carrot
│       │   │   │   └── carrot.txt
│       │   │   ├── celery
│       │   │   │   └── celery.txt
│       │   │   ├── green-bean
│       │   │   │   └── green-bean.txt
│       │   │   ├── kale
│       │   │   ├── potato
│       │   │   │   └── potato.txt
│       │   └── wc
│       └── junk
```



11. Submit a screenshot of the output of running your shell script.

```
[Thu Feb 10 18:09:49] [Thu Feb 10 18:09:49 msilv204@ cyber-server-vm-max:~ ] $ ./simple.sh
Howdy! What is your name?
Michael

Nice to meet you, Michael! What is your favorite color?
purple

Splendid!

Today's time and date is Thu 10 Feb 2022 06:10:02 PM UTC
Michael, your favorite color is purple.

[Thu Feb 10 18:10:02] [Thu Feb 10 18:10:02 msilv204@ cyber-server-vm-max:~ ] $ █
```