Skills Practice - Command Line Fundamentals

Goal:

This goal of this practice exercise is to familiarize you with some of the commands you'll use almost every time you log into a Linux system. Additionally, you will learn about the Linux directory structure.

Instructions:

Start a Virtual Machine

For this practice we can use a virtual machine that we created in the previous project. First, start a command line session. Change into your `linuxclass` folder and then change into the `testbox01` directory.

```
cd linuxclass
cd testbox01
```

Next, start the virtual machine using the `vagrant up` command. If the virtual machine is already running, vagrant will let you know that it's ready to use. If it's stopped or paused, vagrant will start the virtual machine.

```
vagrant up
```

Connect to the virtual machine.

```
vagrant ssh
```

Command Line Practice

Let's start out by running the ls command.

```
ls
```

Were you surprised by the lack of output? If there are no normal files or directories in the current directory, then `ls` has nothing to display and simply returns you back to your command prompt.

Let's perform an `ls` on the `/etc` directory.

```
ls /etc
```

Now you will see several files and directories returned by the ls command. By the way, configuration files are typically stored in the `/etc` directory. If you need change the settings for the system, a program, or a service you'll most likely be modifying a file in the `/etc` directory or a directory below `/etc`.

Perform an `ls` on `/home`.

```
ls /home
```

You'll see one item in `/home`, which is `vagrant`. Let's see if it's a file or a directory by using `ls` with the `-l` option.

```
ls -l /home
```

Notice the output for `vagrant`. It has many more details. If the first character of the output starts with a "d", then the item is a directory. The rest of that string (`drwx------`) represent the permissions. We don't need to worry about those just yet, but for now know that if a listing starts with a "d", it's a directory. Otherwise, it's a file. This output tells us that `/home/vagrant` is a directory.

```
[vagrant@testbox01 ~]$ ls -l /home
total 4
drwx------. 3 vagrant vagrant 4096 Apr  4 19:22 vagrant
[vagrant@testbox01 ~]$
```

As a matter of fact, `/home/vagrant` is the home directory for the vagrant user. The `/home` directory is where normal user account home directories are created by default. If there were other normal users on the system, their home directories would live in `/home` as well. For example, if I had a "jason" account on the system, its home directory would be `/home/jason`.

Take a look in the `/var` directory.

```
ls /var
```

You'll see several items in `/var`. Probably the one directory you'll be most interested in is the `/var/log` directory.

```
ls /var/log
```

The /var directory is used to store variable data. The /var/log directory is for log files, which by their very nature are variable. As events happen on the system new entries in various files within the /var/log directory are created.

Let's look at one file in particular. It's /var/log/messages. Perform a long listing on the file and see that it is a file, and not a directory.

```
ls -l /var/log/messages
```

On CentOS and RHEL systems, the /var/log/messages file is the main log file for system messages. This is one of the first places you should look when you are trying to solve a problem on a Linux system.

Paths that start with a forward slash (/) are called full or absolute paths. If a path does not start with a forward slash, then it is a relative path. A relative path is relative to your current location, called the present working directory. See what directory you're in by running the pwd command.

```
pwd
```

It will report that you're in the /home/vagrant directory. Remember that this is the home directory for the user that you're logged into the system as. When you log into a system you are placed into your home directory.

Let's change into the /home directory and confirm you are actually there.

```
cd /home
pwd
```

Since you're in /home, when you perform an ls the contents of /home will be displayed. Let's try it now.

```
ls
```

You'll see vagrant, which represents /home/vagrant. Use a relative path to /home/vagrant to change into that directory.

```
cd vagrant
pwd
```

You'll find that you're in `/home/vagrant`. One handy shortcut to know is the "`cd ..`" command, which takes you to the parent directory of your current directory. Run the command and confirm your present working directory.

```
cd ..
pwd
```

You're now in `/home`. If you run the `cd` command without providing any arguments or paths, you will be placed in your home directory. Try it now.

```
cd
pwd
```

Sure enough, you're back home in `/home/vagrant`.

Viewing and Editing Files

Now you know how to navigate around the file system and how to list the contents of directories. The next step is viewing and modifying the contents of files. First, let's look at the contents of the `/etc/services` file with the `cat` command.

```
cat /etc/services
```

This file lists service names and their associated network port numbers. For example, the SSH service uses port 22, HTTP uses port 80, etc. The contents of the file aren't really important to us at this point. I just happen to know that this file has several thousand lines and will scroll off the top of your screen if you simply `cat` it, like we just did.

Let's use the less command to look at the file. This way we can control the navigation through the file, up and down.

```
less /etc/services
```

Press the spacebar and notice that the file advances one screen. Press enter, and notice that the file advances by one line. Press the "b" key to go back one screen. Also, try using your arrow keys. If you want to learn about advanced navigation and search features, press "h" for help and read the instructions. When you're done reading the help press "q" to quit the help. When you're done looking at the file, press "q" to quit `less`.

If you want to edit or even create a new file, you can use the `nano` command. If you happen to have another text editor that you like such as `vi` or `emacs`, feel free to use those. However, if you're not familiar with those, stick with `nano` as it's easy to use and learn.

Let's create a new file named "file.txt".

```
nano file.txt
```

Now type the following text into the file.

```
This is fun!
```

You'll notice a menu at the bottom of your screen. Some of the common commands include ^O (Control-O), which stands for writeOut the file, or save. If you want help, select ^G (Control-G). When you're done typing in the text, hit ^X (Control-X). This causes nano to exit. If you have unsaved changes it will ask you whether or not you'd like to save them. Type "y" for yes.

You've just created your first file! Let's list the contents of your present working directory and then let's look at the contents of that file.

```
ls
cat file.txt
```

This file only has one line, so we can see its entire contents with the `cat` command. You can use `less` on short files as well. It never hurts to experiment. Try it out.

```
less file.txt
```

Remember to type "q" when you're done looking at the file.

Now let's rename the file from "file.txt" to "file2.txt".

```
mv file.txt file2.txt
```

Confirm your changes by listing the contents of your present working directory.

```
ls
```

Display the contents of "file2.txt" to make sure it has the contents of the original file we created.

```
cat file2.txt
```

Sure enough, it does! Let's edit it and add the following text to it: "Now this is file2!" You'll want to arrow down to a blank like and add that text.

```
nano file2.txt
```

After you've entered the text hit ^X (Control-X) to exit the file and "y" to confirm you want to save your changes.

Look at file again.

```
cat file2.txt
```

Now the file has two lines.

Extra Practice

There are other top-level directories that you can explore. See what's available by performing a listing against the root of the file system.

```
ls /
ls -l /
```

Feel free to change into directories, perform directory listings, and have fun exploring.

Here are several of the top level directories and a short description of what they contain. I've bolded the ones you'll most often use and need to understand.

/ **The root of the file system.**
/bin Binaries and other executable programs.
/boot Files required to boot the Linux operating system, including the Linux kernel.
/dev Device files. Practically everything in Linux is represented as a file. The files in this directory represent devices, such as disks.
/etc **Configuration files.**
/home **Home directories.**
/lib Libraries
/lib64 64-bit Libraries
/media Typically used to mount (attach) media such as CD's and DVD's.
/mnt Typically used to mount (attach) additional file systems.

/opt Optional or third party software.
/proc A virtual file system that provides process and kernel information as files.
/root The root account's home directory.
/run Run-time variable data: Information about the running system since last boot including currently logged-in users and running daemons.
/sbin Essential system binaries and executables.
/srv Site-specific data for services provided by the system.
/sys Used by sysfs which is a virtual file system provided by the Linux kernel that includes information about various kernel subsystems.
/tmp Temporary files, typically cleared at boot time.
/usr This is a secondary hierarchy for read-only user data. It contains the majority of user utilities and applications.
/vagrant If you are using Vagrant to control your virtual machines, the Vagrant project directory on your local computer will be mounted (and shared to the VM) at /vagrant.
/var Variable data, most notably log files.

Finish

When you're done exploring, exit the virtual machine and power it off.

```
exit
vagrant halt
```