Goals

- Linux System basic commands

Notes:

- Commands preceded with *$* imply that you should execute the command as a general user - not as root.
- Commands preceded with # imply that you should be working as root.
- Commands with more specific command lines (e.g. “rtrX>" or “mysql>") imply that you are executing commands on remote equipment, or within another program.
- If a command line ends with "\" this indicates that the command continues on the next line and you should treat this as a single line.

Exercises

Log in as the class users user using ssh to your machine.

```
username:  xxxxx
password: {given in class}
ssh  xxxx@ipaddress
```

Become the root user

At the command prompt type the following command:

```
$ sudo -s
```

Enter your password when prompted

Now that you are root the command prompt will change. We indicate this using the “#” symbol.

You are now the super user - be careful!

Ok, exit the root account:

```
# exit
```

```
$
```

Look at the network configuration of your host

```
$ cat /etc/network/interfaces
```

Notice that configuration of your host is done using DHCP. “cat” is for “concatenate” and is one way to view what is in a file.

List files:

Use ls to list files:

```
$ cd ~          [go to your home directory]
$ ls
```

Do you see anything? Try this instead:

```
$ ls -lah
```

What's inside one of these files?

```
$ cat .profile
$ less .profile
$ clear
```

Press “q” to get out of the less display.

If you don't understand what cat, clear or less do, then type:

```
$ man cat
$ man clear
$ man less
```

Working with the command prompt:

You can recall previous commands by using the up-arrow and down-arrow keys. Give this a try now.

Alternately, try typing this command:

```
$ history
```

If you wish to execute one of the commands in the list you saw type:

```
$ !nn
```

Where “nn” is the number of the command in the history list. This is useful if you want to run a past command that was long and/or complicated.

Command completion:

With the bash shell you can auto-complete commands using the tab key. This means, if you type part of a command, once you have a unique string if you press the TAB key the command will complete. If you press the TAB key twice you'll see all your available options. Your instructor will demonstrate this, but give it a try by doing:

```
$ hist<tab>
$ del<tab><tab>
$ rm <tab><tab>          [Include the space after the “rm”]
```

Working with pipes:

Examples

We saw an example of using pipes when we sorted the contents of our /sbin directory during the presentation. What if you wanted to have this information available in a file and sorted?

```
$ cd
$ ls /sbin | sort > sbin.txt
```

Now view the contents of what is in sbin.txt to verify that this worked.

```
$ less sbin.txt
```

Press the “q” key to quit viewing the contents.

Finding text strings:

Use the command grep to print lines matching a pattern in a data stream (such as a file). For example, view the entry for theec2-useraccount in the system passwd file:

```
$ sudo grep ec2-user/etc/passwd
```

You should see something like:

ec2-user:x:1000:1000:System Administrator,,,:/home/ ec2-user:/bin/bash

The previous items above are:

userid:passwd:uid:gid:Name,extrastuff,,:HomeDir:LoginShell

grep is often used with a pipe to FILTER the output of commands. For instance:

```
$ history | grep ls
```

Will display your previous use of the ls command from exercise 2.