Introduction to Linux

Command Line Practical

If you can answer the test questions on the DLE using the command line, then you have enough to complete the module

Practical I

A shell is a command line interpreter, an interface between the user and the low-level operating system. Mostly it invokes small utility programmes.

Shells are the user interface, usually always running. Takes your commands, passes them to the kernel as procedure calls which then carry out your job. Shells keep you a layer away from physical memory, providing a safer system.

Open a terminal/shell. You will automatically be placed into your home directory.

Commands we could be using include:

- cd (change directory)
- ls (list directory contents)
- mv (move or rename)
- rm (remove)
- cp (copy)
- ln (link)
- cat (view file)
- more (view file one page at a time)
- chmod (change permissions)
- pwd (present working directory)
- mkdir (make new directory)
- rmdir (remove directory)
- man (read the manual page documentation for a command)
- apropos (list commands that have a particular keyword in their description)
- find (search for files or directories)
- locate (faster than find)
- date (display the system date and time)
- who (who is on the system)
- whoami (who are you logged in as)

(not all of the above are in the practicals below)

If you have questions about the syntax of any command, try typing:
'man <command_name>'.

To find out which version of Linux you are using type:
> uname –r (shorter description)
> uname -a

At the Prompt type:

```
> pwd
```

This tells you where you are. Type:

```
> ls
```

This gives you the contents of that directory

Look up *ls* in the manual to find out what options there are. Type:

```
> man ls
```

Move around the manual using page up/down. Exit the manual (type q).

List files using *ls* with different options
e.g.

```
> ls -alg
```

Examine the permissions for various files.

You can move around the file system using *cd* and *pwd* (to find out where you are) & ls (to list contents). Type

```
> cd ..
```

This takes you up one level.

```
> cd /home
```

Takes you to the home directory (not sec204)
Typing cd on its own takes you to sec204

Move from here to the booksrc directory

```
> cd /sec204/Desktop/booksrc
```

List the files in this directory

Let us look at some files using *cat*, *less* and *more*

```
> cat helloworld.c
> cat fmt_vuln.c

> less fmt_vuln.c
> more fmt_vuln.c
> tail fmt_vuln.c
```

Move back to the home directory (see above), then to the sec204 directory.

Type the following command to open the nano editor

```
> nano
```

Type something in here, then hit CtrlX & save it as ‘myfile.txt’. This file will be added to the directory contents.

Use *cat* to show the contents of the file.
List the permissions for the file (e.g. ls –g m*)

Use *chmod* to change permissions for the file for you, group members & others - check the permissions after each command

```
> chmod 700 myfile.txt
> chmod go+rw myfile.txt
> chmod 774 myfile.txt
```

Copy the file in the same directory to another name: myfile1.txt. If you have a file of the same name this will be overwritten.

```
> cp myfile.txt myfile1.txt
```

See if there are any differences between the two files

```
> diff myfile.txt myfile1.txt
```

Create a new directory *docs*

```
> mkdir docs
```

Copy the new file to the *docs* directory. Move to the docs directory and copy the original there.

```
> cp ../myfile.txt .
```

Rename myfile1.txt using the *mv* command to myfile2.txt. Then copy it back to the sec204 directory. Move back to this directory.

List the files in the directory:

```
ls myfile?.txt
ls myfile*.txt
```

Delete myfile2.txt (but not myfile.txt) using the following wildcard:

```
rm my*2.txt
```

Try to delete the docs directory.

```
> rmdir docs
```

Make a new directory *documents*. Move all the contents of the *docs* directory using the following:

```
mv ./docs/* ./documents
```

Now try and delete the docs directory.

Rename the documents directory docs.

In an editor (e.g. nano) create a list of random numbers (each on a new line) save it as numlist and check the result.

```
sort –n numlist > sorted
```

Look at the results in sorted.

then type the following command.

sort –n numlist > sorted ; cat sorted ; rm sorted

Type
> nano &

Notice the PID at the command prompt. This command creates a job running in the background, allowing you to continue working with the command line.

> jobs
To see what jobs are running

> *ps –ef*
This displays all the processes, the user name, PPID (parent process id), start time and full command line.

Stop the process you've just created. The 'kill' command is used to stop a process. This sends the process a 'SIGTERM' signal, meaning terminate. This can be carried out using the job number
> *kill %1* (or whatever job number it is)

or using the PID
> *kill 1502*
meaning terminate pid 1502

Some processes will not respond (if they are in an error condition, or are programmed to ignore SIGTERM) or close down slowly. Use -9 option to kill the process immediately

> *kill -9 1502*

Then use
> *history*
> *!115 (or a recent command line number)*
> *!! (repeats last command)*
> *clear*