T: 0191 222 5000
E: enquiries@netskills.ac.uk
www.netskills.ac.uk

Linux Orientation

Quick start information

Accessing your Virtual Machine

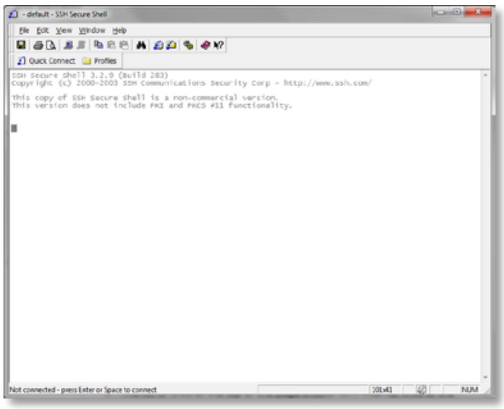
Important

These materials assume you are using the **SSH** application found on the Newcastle University common desktop, however it is only the means by which you access your virtual machine (VM).

Once you are in and at the command prompt, the materials become somewhat generic, so if you prefer to use something else then feel free ☺

Logging In Locate and open an SSH console application.

You should see the following window:



```
SSH Secure Shell 3.2.9 (Build 283)
Copyright (c) 2000-2003 SSH Communications Security Corp - http://www.ssh.com/

This copy of SSH Secure Shell is a non-commercial version.
This version does not include PKI and PKCS #11 functionality.
```

Not connected - press Enter or Space to connect

Figure 1. SSH window (not connected)

Click the `Quick Connect` button near the top left hand side.

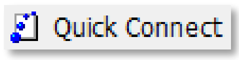
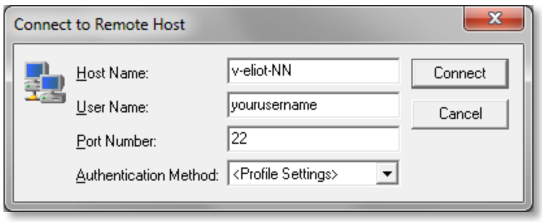


Figure 2. Quick connect

Complete the connection details as shown below – replacing **NN** with the number of your VM and **`yourusername`** with your University login ID.

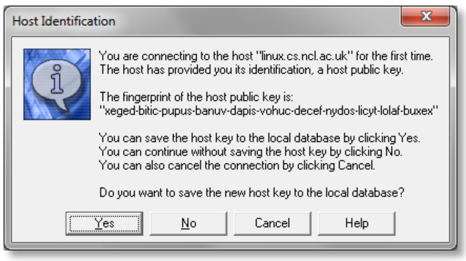


Connect to Remote Host

Host Name:	v-eliot-NN
User Name:	yourusername
Port Number:	22
Authentication Method:	<Profile Settings>

Figure 3. Connection details

Now click `Connect`. If all goes well you should be prompted to accept the host certificate for your machine (this only happens the first time you connect).



You are connecting to the host "linux.cs.ncl.ac.uk" for the first time. The host has provided you its identification, a host public key.

The fingerprint of the host public key is:
"xeged-bitic-pupus-banuv-dapis-vohuc-decef-nydos-licyt-lolaf-buxex"

You can save the host key to the local database by clicking Yes.
You can continue without saving the host key by clicking No.
You can also cancel the connection by clicking Cancel.

Do you want to save the new host key to the local database?

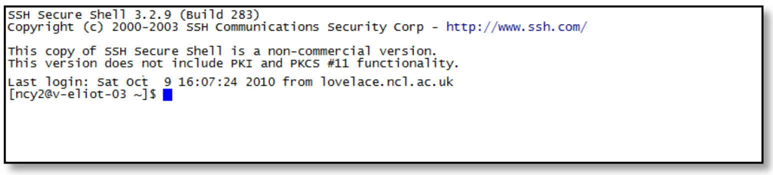
Figure 4. Host certificate prompt (this one is for linux.cs.ncl.ac.uk)

Click `Yes` to continue.

Once you've accepted the certificate you will be prompted for your password.

This is your **normal windows password** that you use for logging on to PCs on campus.

Finally you should be returned to the window you saw in Figure 1. , but this time the window contents will not be greyed out. You should now be logged in and able to type at the command prompt. e.g.



```
SSH Secure Shell 3.2.9 (Build 283)
Copyright (c) 2000-2003 SSH Communications Security Corp - http://www.ssh.com/

This copy of SSH Secure Shell is a non-commercial version.
This version does not include PKI and PKCS #11 functionality.

Last login: Sat Oct  9 16:07:24 2010 from lovelace.ncl.ac.uk
[ncy2@v-eliot-03 ~]$ 
```

Figure 5. Logged in and ready to go

Finally you may be prompted to save this connection by **adding a profile** for it. You can do if you wish – you should be able to work out how to do that though ☺

Your home directory

When you first login in you will be in your **home** directory.

You have full permissions inside this directory, so this is where you will be able to download and build your source files.

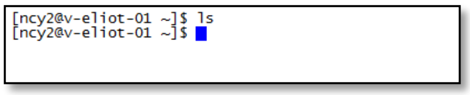
Later on you may be serving web pages from here too!

Looking Around

Directories

At the command prompt type: `ls`

This is the command to list the files and directories at your current location. As your home directory is empty you should see nothing listed.

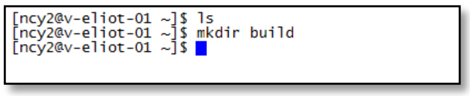


```
[ncy2@v-eliot-01 ~]$ ls
[ncy2@v-eliot-01 ~]$ 
```

Figure 6. Empty home directory

Now type: `mkdir build`

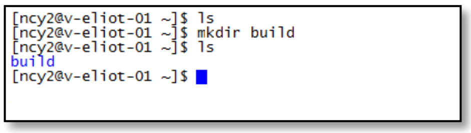
This is the command to make a new directory called `build`. If this is successful you will be returned to your command prompt.



```
[ncy2@v-eliot-01 ~]$ ls
[ncy2@v-eliot-01 ~]$ mkdir build
[ncy2@v-eliot-01 ~]$ 
```

Figure 7. Making a directory

Now type `ls` again. This time your new directory should be listed.

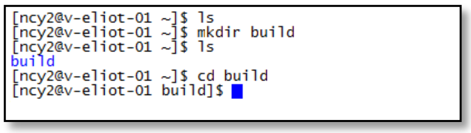


```
[ncy2@v-eliot-01 ~]$ ls
[ncy2@v-eliot-01 ~]$ mkdir build
[ncy2@v-eliot-01 ~]$ ls
build
[ncy2@v-eliot-01 ~]$ 
```

Figure 8. New directory listed

To enter the new directory, type: `cd build`

You should see the command prompt change to show where you are.



```
[ncy2@v-eliot-01 ~]$ ls
[ncy2@v-eliot-01 ~]$ mkdir build
[ncy2@v-eliot-01 ~]$ ls
build
[ncy2@v-eliot-01 ~]$ cd build
[ncy2@v-eliot-01 build]$ 
```

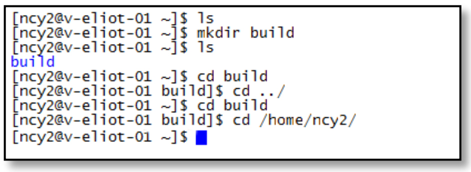
Figure 9. Moving into a new directory

Return to your **home** directory by typing either:

`cd ../` (move up one level from the current directory)

or

`cd /home/yourusername/` (go to my **home** directory from *wherever* I am)



```
[ncy2@v-eliot-01 ~]$ ls
[ncy2@v-eliot-01 ~]$ mkdir build
[ncy2@v-eliot-01 ~]$ ls
build
[ncy2@v-eliot-01 ~]$ cd build
[ncy2@v-eliot-01 build]$ cd ../
[ncy2@v-eliot-01 ~]$ cd build
[ncy2@v-eliot-01 build]$ cd /home/ncy2/
[ncy2@v-eliot-01 ~]$
```

Figure 10. Moving out of a directory (2 versions)

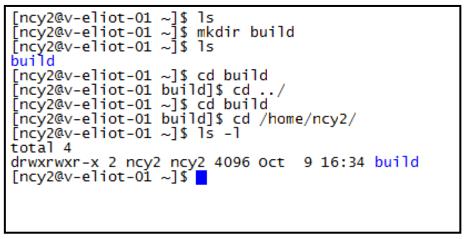
User Groups & Permissions

Long Listings

In your home directory type:

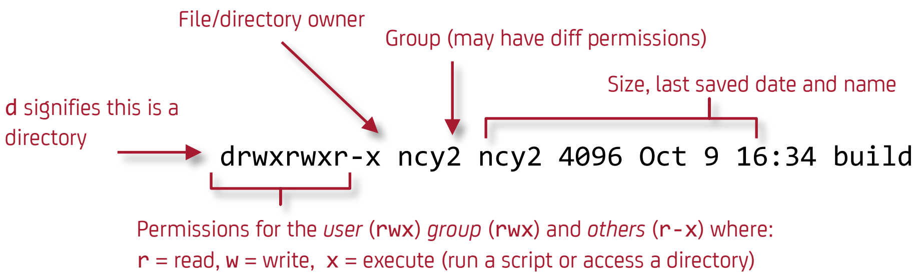
```
ls -l
```

This is the `ls` command, with an option (`-l`) tacked on to the end to list the contents along with more detailed information about them. You should see the following:



```
[ncy2@v-eliot-01 ~]$ ls
[ncy2@v-eliot-01 ~]$ mkdir build
[ncy2@v-eliot-01 ~]$ ls
build
[ncy2@v-eliot-01 ~]$ cd build
[ncy2@v-eliot-01 build]$ cd ../
[ncy2@v-eliot-01 ~]$ cd build
[ncy2@v-eliot-01 build]$ cd /home/ncy2/
[ncy2@v-eliot-01 ~]$ ls -l
total 4
drwxrwxr-x 2 ncy2 ncy2 4096 Oct  9 16:34 build
[ncy2@v-eliot-01 ~]$
```

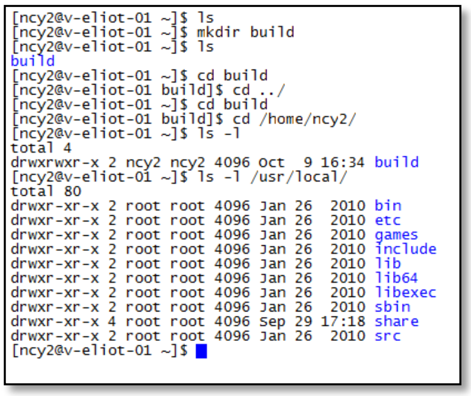
Figure 11. Long listing to show permissions


File/directory owner
Group (may have diff permissions)
Size, last saved date and name
d signifies this is a directory
drwxrwxr-x ncy2 ncy2 4096 Oct 9 16:34 build
Permissions for the *user* (rwx) *group* (rwx) and *others* (r-x) where:
r = read, w = write, x = execute (run a script or access a directory)

Notice how the files are listed as being owned by you and accessible by you and your group (which actually just consists of you – groups are more useful in other contexts we won't be worrying about).

Compare those results with what you get when you type:

```
ls -l /usr/local/
```



```
[ncy2@v-eliot-01 ~]$ ls
[ncy2@v-eliot-01 ~]$ mkdir build
[ncy2@v-eliot-01 ~]$ ls
build
[ncy2@v-eliot-01 ~]$ cd build
[ncy2@v-eliot-01 build]$ cd ../
[ncy2@v-eliot-01 ~]$ cd build
[ncy2@v-eliot-01 build]$ cd /home/ncy2/
[ncy2@v-eliot-01 ~]$ ls -l
total 4
drwxrwxr-x 2 ncy2 ncy2 4096 Oct  9 16:34 build
[ncy2@v-eliot-01 ~]$ ls -l /usr/local/
total 80
drwxr-xr-x 2 root root 4096 Jan 26  2010 bin
drwxr-xr-x 2 root root 4096 Jan 26  2010 etc
drwxr-xr-x 2 root root 4096 Jan 26  2010 games
drwxr-xr-x 2 root root 4096 Jan 26  2010 include
drwxr-xr-x 2 root root 4096 Jan 26  2010 lib
drwxr-xr-x 2 root root 4096 Jan 26  2010 lib64
drwxr-xr-x 2 root root 4096 Jan 26  2010 libexec
drwxr-xr-x 2 root root 4096 Jan 26  2010 sbin
drwxr-xr-x 4 root root 4096 Sep 29 17:18 share
drwxr-xr-x 2 root root 4096 Jan 26  2010 src
[ncy2@v-eliot-01 ~]$
```

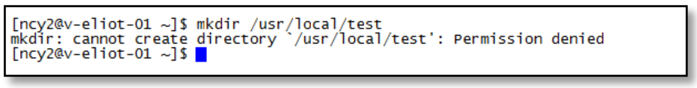
Figure 12. Listing of `/usr/local`

Notice how all of these files and directories are owned and accessible by the user `root`. You have ***read*** access to them, but will need to be able to ***write*** to this area in order to install and configure your web server.

Try making a directory here with:

```
mkdir /usr/local/test
```

You should get a `Permission denied` error as shown below



```
[ncy2@v-eliot-01 ~]$ mkdir /usr/local/test
mkdir: cannot create directory `/usr/local/test': Permission denied
[ncy2@v-eliot-01 ~]$ 
```

Figure 13. Cannot create directory in `/usr/local`

Working as Root

In order to work inside `/usr/local/` you will need to be able to work with `root` (or at least root-like) permissions.

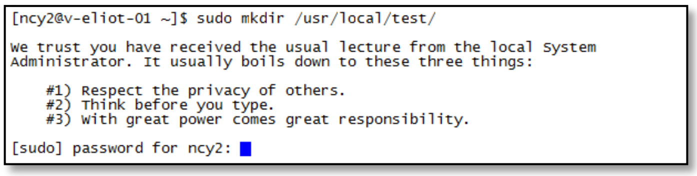
You cannot login to your VM as `root`, however you can temporarily act as a root user by using the `sudo` command. This is similar (although not directly analogous) as the UAC system in Windows which allows you to enter an administrator password to install software etc.)

"sudo make me a sandwich"

Try creating a directory in `/usr/local/` using `sudo`

```
sudo /usr/local/test
```

The first time you do this you'll be met with a friendly – **but important** – warning.



```
[ncy2@v-eliot-01 ~]$ sudo mkdir /usr/local/test/

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

[sudo] password for ncy2: █
```

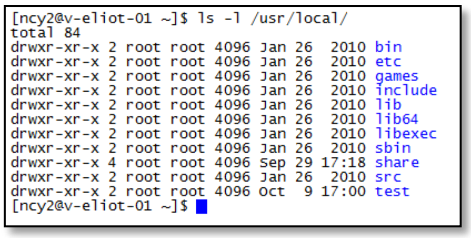
Figure 14. With great power comes great responsibility!

You will be prompted for a password. This is *your* password – same one you logged in with.

If all goes well and the directory is created, type:

```
ls -l /usr/local/
```

Look for your new directory in the listing and notice how it has the same permissions as the others in the directory i.e. it belongs to the `root` user **not you**!



```
[ncy2@v-eliot-01 ~]$ ls -l /usr/local/
total 84
drwxr-xr-x 2 root root 4096 Jan 26  2010 bin
drwxr-xr-x 2 root root 4096 Jan 26  2010 etc
drwxr-xr-x 2 root root 4096 Jan 26  2010 games
drwxr-xr-x 2 root root 4096 Jan 26  2010 include
drwxr-xr-x 2 root root 4096 Jan 26  2010 lib
drwxr-xr-x 2 root root 4096 Jan 26  2010 lib64
drwxr-xr-x 2 root root 4096 Jan 26  2010 libexec
drwxr-xr-x 2 root root 4096 Jan 26  2010 sbin
drwxr-xr-x 4 root root 4096 Sep 29 17:18 share
drwxr-xr-x 2 root root 4096 Jan 26  2010 src
drwxr-xr-x 2 root root 4096 Oct  9 17:00 test
[ncy2@v-eliot-01 ~]$ █
```

Figure 15. New directory – created as `root`

xkcd

If you don't get the sandwich reference, take a look at:

http://xkcd.com/149/

Now you know the full power of `sudo` ☺

Working with Files

You'll need to be able to manipulate files from the command line. These are always useful skills to have and are usually the quickest way to make adjustments to your configuration.

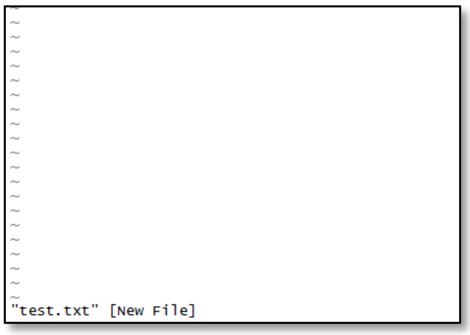
As well as knowing how to move, copy or delete files, you'll also need to be able to edit text files – particularly the configuration files for various parts of your LAMP stack.

There are many (many) different file editors available for Linux, however the one that is always installed on every system is VI (pronounced "vee-eye"). It is marmite for most – you either like it or you don't – however even those who hate it with a passion will admit that it is useful to know how to do the basics!

Create a text file with VI

Change to your **home** directory (if you are not already there) and type the following at the command prompt:

```
vi test.txt
```



```
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"test.txt" [New File]
```

Figure 16. A new file in VI

What you have now is a blank document in VI, ready for you to edit.

Command and Insert modes

Files in VI open in **command** mode which lets you navigate the document and issue commands to quit, save etc.

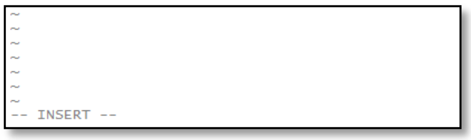
To edit the file you will need to enter **insert** mode.

Insert mode to edit

To enter insert mode type:

```
i
```

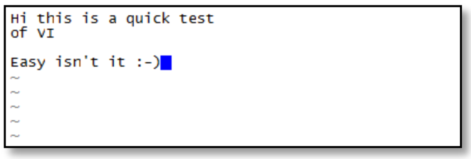
Notice how the status line at the bottom of the console window changes to let you know you are now in insert mode.



```
~
~
~
~
~
~
~
-- INSERT --
```

Figure 17. VI in insert mode

Type some text into the file. In this mode you can use the `<return>` key to add new lines.



```
Hi this is a quick test
of VI

Easy isn't it :-)
~
~
~
~
~
```

Figure 18. Editing a text file in VI

When you have finished, you can leave insert mode and return to command mode by hitting the `<esc>` key.

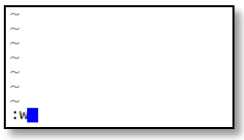
Now you need to save your file...

Command mode to save and quit

Once you have returned to command mode, you can save your file by typing:

```
:w
```

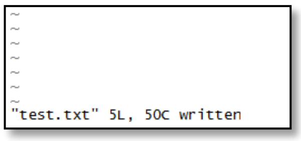
Notice how the commands (and any feedback from them) appear at the bottom of the window when you type them.



```
~
~
~
~
~
~
~
:w
```

Figure 19. Issuing a write command (save) in VI

Hit `<return>` to save. You should see confirmation at the bottom of the window.



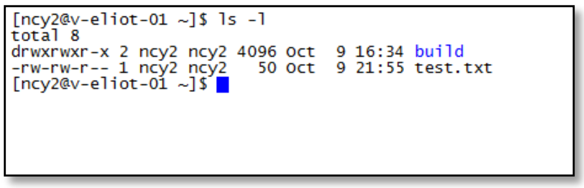
```
~
~
~
~
~
~
~
"test.txt" 5L, 50C written
```

Figure 20. Write confirmation in VI

To quit VI and return to the console type:

```
:q
```

Hit `<return>` and you should be back where you started. If you list the files in your home directory, you should see your new file.



```
[ncy2@v-eliot-01 ~]$ ls -l
total 8
drwxrwxr-x 2 ncy2 ncy2 4096 Oct  9 16:34 build
-rw-rw-r-- 1 ncy2 ncy2   50 Oct  9 21:55 test.txt
[ncy2@v-eliot-01 ~]$
```

Figure 21. A new file - created in VI

Save *and* quit?

You can combine `:w` and `:q` in order to save your file *and* quit in one step using:

```
:wq
```

Quit *without* saving?

You can quit *without* saving anything by using:

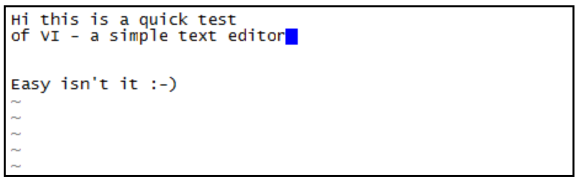
```
:q!
```

Edit an existing file with VI

Re-open your test file in VI with:

```
vi test.txt
```

Use the arrow keys to move the cursor to the point in the file you want to edit and then enter *insert* mode (`i`).



```
Hi this is a quick test
of VI - a simple text editor

Easy isn't it :-)
~
~
~
~
~
```

Figure 22. Editing a file in VI

Make a change and then drop back into *command* mode (`<esc>`), save and quit.

```
:wq
```

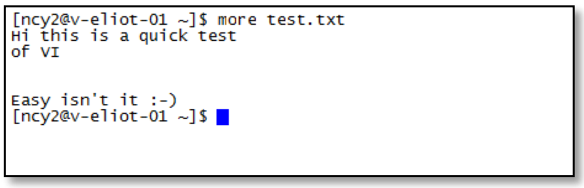
Read a file *without editing* it

Sometimes you will simply want to check the contents of a file, without opening it up in an editor. To do this use the Linux `more` command.

Read your new text file by typing:

```
more test.txt
```

The contents will be displayed in the console window.



```
[ncy2@v-eliot-01 ~]$ more test.txt
Hi this is a quick test
of VI

Easy isn't it :-)
[ncy2@v-eliot-01 ~]$
```

Figure 23. Using `more` to show file contents

For large files, `more` will show you as much of the file as will fit in your console window and you can page through to the end by hitting the `<spacebar>`.

Moving files

You can move files around by using the `mv` command.

The syntax for the command is:

```
mv [file(or directory) to move] [destination]
```

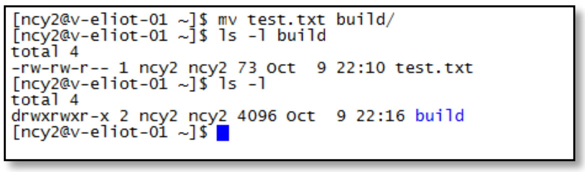
Try moving your file to the `build` directory you created earlier.

```
mv test.txt build/
```

You can check to see if it worked by checking the listing for `build`.

```
ls -l build
```

Check your file is no longer in your home directory by checking there with an `ls` (or `ls -l`)



```
[ncy2@v-eliot-01 ~]$ mv test.txt build/
[ncy2@v-eliot-01 ~]$ ls -l build
total 4
-rw-rw-r-- 1 ncy2 ncy2 73 Oct  9 22:10 test.txt
[ncy2@v-eliot-01 ~]$ ls -l
total 4
drwxrwxr-x 2 ncy2 ncy2 4096 Oct  9 22:16 build
[ncy2@v-eliot-01 ~]$ 
```

Figure 24. Moving a file using `mv`

Renaming files

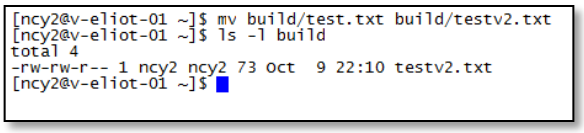
You can also use `mv` to rename files (i.e. "move" it to a new file in the same location).

Rename your test file (remember it is now in your `build` directory!).

```
mv build/test.txt build/testv2.txt
```

List `build` to check the rename worked.

```
ls -l build
```



```
[ncy2@v-eliot-01 ~]$ mv build/test.txt build/testv2.txt
[ncy2@v-eliot-01 ~]$ ls -l build
total 4
-rw-rw-r-- 1 ncy2 ncy2 73 Oct  9 22:10 testv2.txt
[ncy2@v-eliot-01 ~]$ 
```

Figure 25. Renaming a file with `mv`

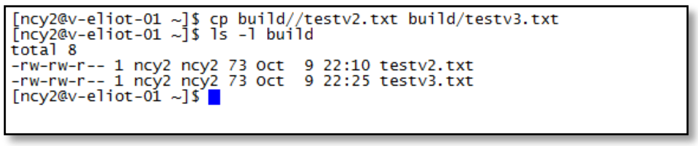
Copying files

The syntax for the copy command `cp` is the same as `mv`. The difference is that you are making er... a copy.

You can copy to a new location or a different filename in the same location. Try.

```
cp build/testv2.txt build/testv3.txt
```

Notice how this time you have created a **new** file (a copy of `testv2.txt`)



```
[ncy2@v-eliot-01 ~]$ cp build//testv2.txt build/testv3.txt
[ncy2@v-eliot-01 ~]$ ls -l build
total 8
-rw-rw-r-- 1 ncy2 ncy2 73 Oct  9 22:10 testv2.txt
-rw-rw-r-- 1 ncy2 ncy2 73 Oct  9 22:25 testv3.txt
[ncy2@v-eliot-01 ~]$ 
```

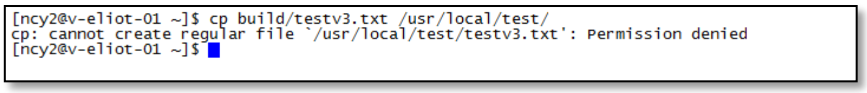
Figure 26. Copying a file using `cp`

"Make me a sandwich!"

Now try and copy one of your files to the `test` directory you created earlier inside `/usr/local/`

```
cp build/testv3.txt /usr/local/test/
```

You should be denied permission to do that ☺



```
[ncy2@v-eliot-01 ~]$ cp build/testv3.txt /usr/local/test/
cp: cannot create regular file `/usr/local/test/testv3.txt': Permission denied
[ncy2@v-eliot-01 ~]$ 
```

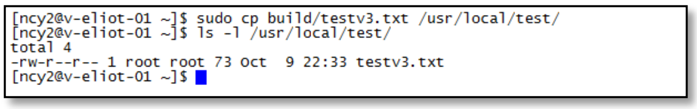
Figure 27. Permission denied to copy to `/usr/local/test/`

You need `root` permission to copy files to `/usr/local/` - which means you need `sudo` again.

Try:

```
sudo cp build/testv3.txt /usr/local/test/
```

That should work!



```
[ncy2@v-eliot-01 ~]$ sudo cp build/testv3.txt /usr/local/test/
[ncy2@v-eliot-01 ~]$ ls -l /usr/local/test/
total 4
-rw-r--r-- 1 root root 73 Oct  9 22:33 testv3.txt
[ncy2@v-eliot-01 ~]$ 
```

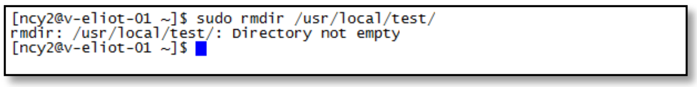
Figure 28. Using sudo to copy to `/usr/local/test/`

Deleting files & directories

There is a specific Linux command for deleting directories called `rmdir`.

The snag with this is it only works if the directory is empty. Try:

```
sudo rmdir /usr/local/test
```

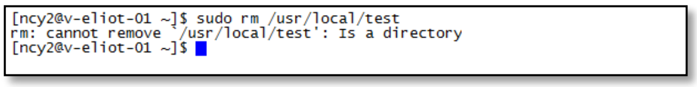


```
[ncy2@v-eliot-01 ~]$ sudo rmdir /usr/local/test/
rmdir: /usr/local/test/: Directory not empty
[ncy2@v-eliot-01 ~]$ 
```

Figure 29. Cannot delete an empty directory

Linux also provides the command `rm` for deleting files and directories so try:

```
sudo rm /usr/local/test
```



```
[ncy2@v-eliot-01 ~]$ sudo rm /usr/local/test
rm: cannot remove `/usr/local/test': Is a directory
[ncy2@v-eliot-01 ~]$ 
```

Figure 30. Figure 1 Cannot use `rm` to delete a directory

This is because `rm` is really for removing files. However you can add the `-R` option and remove files recursively. This will delete a directory *and all of its contents*.

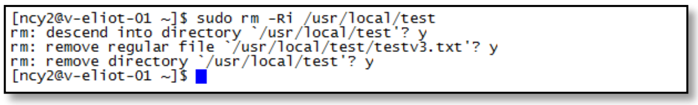
So this will work:

```
sudo rm -R /usr/local/test
```

If you are really worried about the possible dangers of `rm -R`
(and you should be – especially combined with `sudo`) then add in the `i` flag i.e.

```
sudo rm -Ri /usr/local/test
```

This method will ask you to confirm each deletion. Answering `n` (no) at any stage will stop the process.



```
[ncy2@v-eliot-01 ~]$ sudo rm -Ri /usr/local/test
rm: descend into directory `/usr/local/test'? y
rm: remove regular file `/usr/local/test/testv3.txt'? y
rm: remove directory `/usr/local/test'? y
[ncy2@v-eliot-01 ~]$ 
```

Figure 31. Recursive deletion *with* confirmations

Some Other Goodies

Copy & paste

You can copy and paste into your console window, but you will find the keyboard shortcuts are different (important for pasting into the console window).

`<ctrl>+c` *cannot* be used (see below for why).

You'll probably find you need `<shift>+insert` instead – but if your mouse is set up correctly then you may find that *right*-clicking in the console window will paste the clipboard contents to the command line ☺

Command history

Fed up with retyping commands over and over again?...well you don't as your recently used commands are stored.

Access them from the command prompt by using the `<up>` and `<down>` arrows on your keyboard – try it ☺

To run a command press `<return>,` to edit a command, use the `<left>` and `<right>` arrow keys to move the cursor to the bit you want to change.

Auto-complete

Can't remember the name of the file? Order of the directories?

You can use `<tab>` to auto-complete file and directory names. To try it type:

```
cd /usr/lo
```

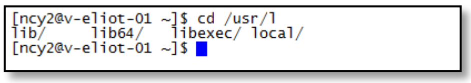
Now press `<tab>` and you should find the line is completed for you to:

```
cd /usr/local
```

If nothing auto-completes, try pressing `<tab>` twice to see all the options. Try:

```
cd /usr/l
```

Pressing `<tab>` will show you the options.



```
[ncy2@v-eliot-01 ~]$ cd /usr/l
lib/      lib64/    libexec/ local/
[ncy2@v-eliot-01 ~]$
```

Figure 32. Auto-complete options

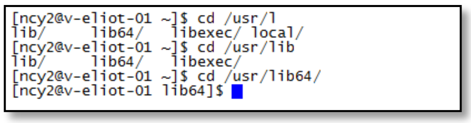
Now try:

```
cd /usr/li
```

Pressing `<tab>` will auto-complete to:

```
cd /usr/lib
```

But there are still 3 possible choices...but you can probably work it out from here ☺



```
[ncy2@v-eliot-01 ~]$ cd /usr/l
lib/      lib64/    libexec/ local/
[ncy2@v-eliot-01 ~]$ cd /usr/lib
lib/      lib64/    libexec/
[ncy2@v-eliot-01 ~]$ cd /usr/lib64/
[ncy2@v-eliot-01 lib64]$
```

Figure 33. Stepwise auto-complete

STOP!! I didn't mean that!

If you need to stop something that has started running (or if you have typed a load of nonsense on the command line and don't want to run it!) hit:

```
<ctrl>+c
```

This will drop you back at the command prompt.

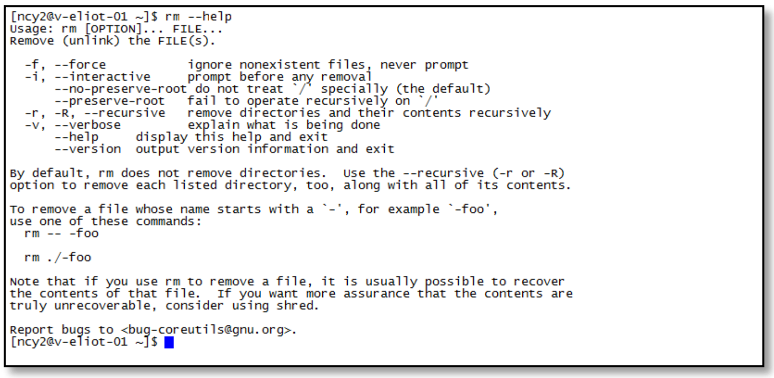
Getting help

If you want to check the available options for a Linux command try:

```
command -help
```

e.g.

```
rm -help
```



```
[ncy2@v-eliot-01 ~]$ rm --help
Usage: rm [OPTION]... FILE...
Remove (unlink) the FILE(s).

  -f, --force           ignore nonexistent files, never prompt
  -i, --interactive     prompt before any removal
      --no-preserve-root do not treat `/' specially (the default)
      --preserve-root   fail to operate recursively on `/'
  -r, -R, --recursive   remove directories and their contents recursively
  -v, --verbose         explain what is being done
      --help     display this help and exit
      --version  output version information and exit

By default, rm does not remove directories.  Use the --recursive (-r or -R)
option to remove each listed directory, too, along with all of its contents.

To remove a file whose name starts with a `-', for example `-foo',
use one of these commands:
  rm -- -foo

  rm ./-foo

Note that if you use rm to remove a file, it is usually possible to recover
the contents of that file.  If you want more assurance that the contents are
truly unrecoverable, consider using shred.

Report bugs to <bug-coreutils@gnu.org>.
[ncy2@v-eliot-01 ~]$
```

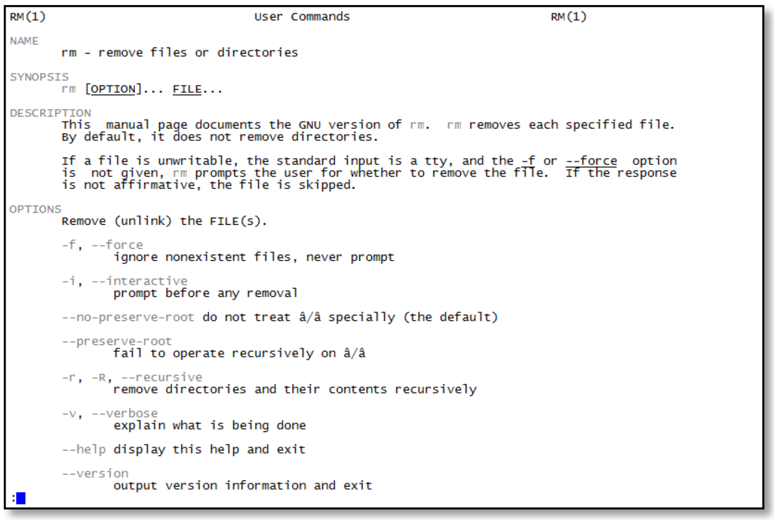
Figure 34. Help options for `rm`

If you want to read the full manual for a command try:

```
man command
```

e.g.

```
man rm
```



```
RM(1)                          User Commands                          RM(1)

NAME
       rm - remove files or directories

SYNOPSIS
       rm [OPTION]... FILE...

DESCRIPTION
       This  manual page documents the GNU version of rm.  rm removes each specified file.
       By default, it does not remove directories.

       If a file is unwritable, the standard input is a tty, and the -f or --force  option
       is  not given, rm prompts the user for whether to remove the file.  If the response
       is not affirmative, the file is skipped.

OPTIONS
       Remove (unlink) the FILE(s).

       -f, --force
              ignore nonexistent files, never prompt

       -i, --interactive
              prompt before any removal

       --no-preserve-root do not treat â/â specially (the default)

       --preserve-root
              fail to operate recursively on â/â

       -r, -R, --recursive
              remove directories and their contents recursively

       -v, --verbose
              explain what is being done

       --help display this help and exit

       --version
              output version information and exit
:
```

Figure 35. Man details for `rm`

How do I log out?

To log out at the command prompt hit:

```
<ctrl>+d
```

You can then close your SSH client (if it doesn't do so automatically).

If you are off campus and have hopped through a timeshare (like `linux.cs`, `aldred` or `finan`) then you will need to `<ctrl>+d` *twice*.

The *first* one will log you out of your VM and return you to the timeshare. The *second* logs you out completely.

Off Campus Access (VM)

You will probably want to be able to work on your VM from off campus.

Unfortunately it is not directly accessible from outside the campus network for security reasons (it is a fully functional server – you are *not* a fully functional sys admin!).

You can however gain access by hopping through one of the Linux timeshare servers which *are* available from off campus.

You will also need an SSH client for your own PC.

Windows

PuTTY is probably the best option for Windows as it is simple and very portable (it is just an EXE file and can be run from a USB stick). You can download a copy from the module resource page at:

https://internal.cs.ncl.ac.uk/modules/2011-12/csc3504/

MacOS/Linux

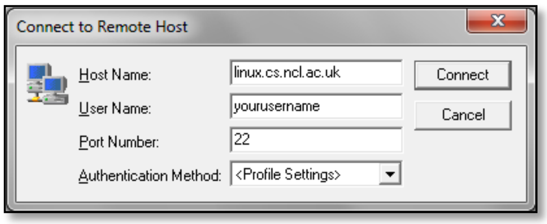
If you have a Mac or Linux PC then SSH is installed by default and should run from the command prompt as `username@remotehost` e.g.

```
> ssh a1234567@linux.cs.ncl.ac.uk
```

Hop through a timeshare

To hop through a timeshare, follow the instructions above for accessing your VM, but instead of accessing it directly log in to the timeshare e.g.

```
linux.cs.ncl.ac.uk
```


Connect to Remote Host

Host Name: linux.cs.ncl.ac.uk
User Name: yourusername
Port Number: 22
Authentication Method: <Profile Settings>

Figure 36. Accessing `linux.cs.ncl.ac.uk`

Once you are logged in type the following at the command prompt:

`ssh vm-eliot-NNN` (where **NNN** is the number for your VM)

The first time you do this you may get a prompt to accept the host certificate for your VM again – type `yes` to accept.

Then you will be asked again for your password (this is to log you in to the VM).

Finally you will be at the command prompt for your VM, as if you were on campus.

Logging out from off-campus

Exactly the same as normal – but you'll need to do it *twice*.

Once to log out from the VM and return to the timeshare, then again to finally log out from the timeshare itself.

What Now?

Now you know how to:

- Access your VM using an SSH client
- Create directories using `mkdir`
- List directory contents their contents using `ls` and `ls -l`
- Move between directories using `cd`
- Run commands as root using `sudo`
- Create and edit text files using VI
- Display file contents using `more`
- Move and copy files using `mv` and `cp`
- Delete files and directories using `rmdir` and `rm`

You will need a few more skills than that, but most of them you should be able to find for yourselves or pick up from subsequent lectures and practical tips.

You are strongly recommended to look at the ISS Linux support material available at:

> http://www.ncl.ac.uk/iss/unix/unixhelp/

In particular make use of the sections on:

Common commands

> http://www.ncl.ac.uk/iss/unix/unixhelp/commandlist.html

Working with files and directories

> http://www.ncl.ac.uk/iss/unix/unixhelp/filedir.html