UWCS Linux 101 Lab

A practical introduction to Linux

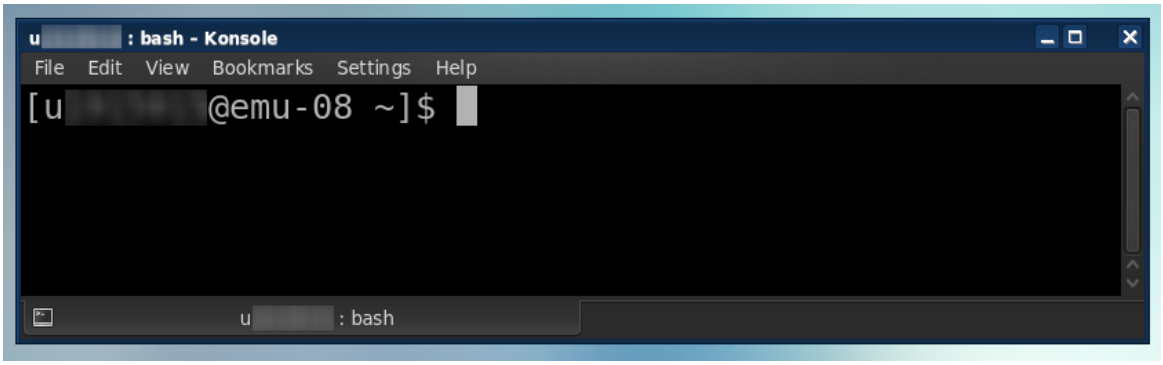
Logging In

Before we start, you'll need to gain access to the department machines - the information needed for this should have been sent to your university email address. If you have any issues logging in, let a lab tutor know! They'll also be around to help if you get stuck.

Now that you've (hopefully) logged in, let's get started with the terminal!

Getting Around

First, click the start button at the bottom-left corner of the screen.
Then, click the "terminal" menu item. You should see something like this:



When opening the terminal, you'll be placed in your home directory - denoted by a ~. But what can you actually access from here? Let's use our first command to find out:

`ls` - Lists the contents of the directory you're currently in.

All you have to do is type `ls` and hit enter to execute the command. What do you see? For the moment, there should just be a few folders, also referred to as sub-directories. If you want to find out what's inside, you need to be able to change directory:

`cd <path>` - Changes directory to the specified path.

Like most commands we will encounter from here on out, this one takes an argument, `path`. This argument specifies exactly where we'd like to change directory to.

Example	Explanation
`cd scripts`	Change to the `scripts` folder in the current directory.
`cd ..`	Change to the parent directory (a level above the current one).
`cd ~`	Change to the home directory (useful if you get lost).

You can also chain multiple directory moves together! For example, moving a directory with `cd ..` then down with `cd scripts` could be combined into just `cd ../scripts`.

Task 1: Before moving on, use the terminal to explore all the folders accessible from your home directory. Make sure to list out all the contents - is there anything inside?

By now, you have sadly realised the folders are empty. So, let's aim to fix that.

Creation and Deletion

In this section, we'll introduce commands to create and delete entities on the file system:

`touch [<path>/]<name>`
Creates a file with the given name in the specified directory (current if no path given).

Note: the square brackets denote that an argument is optional.

`mkdir` is used to create sub-directories in a similar way (replace `touch` with `mkdir`). Files can be removed using the `rm` command (again, replace `touch`). If we want to be able to remove folders, we can override the default behaviour using the `-r` (recursive) flag:

`rm -r [<path>/]<name>`
Removes the specified file or directory. If non-empty, **all contents are deleted.**

Flags are optional arguments declared after the command name that add additional functionality. To view a command's full list of arguments, you can type `man <command>`.

Task 2: With all of this in mind, it's time to make some files and folders! We'll walk you through the process of replicating (then partially deleting) the folder structure below.

While in the home directory, we create the `foo` directory and it's `bar.baz` file:

```
mkdir foo
touch foo/bar.baz
```

To create the `public_html` directory and its contents, a similar pattern is followed. You might think to create both folders using `mkdir public_html/css`, but this will fail, as `mkdir` requires all but the last part of the path to exist already. We can override this behaviour with the `-p` flag, which creates intermediate directories when required. From here, try creating `index.html` and `app.css` for yourself.

Finally, we're going to remove the foo directory and its contents. One of the ways you can do this is `rm -r foo`. For a bonus challenge, see if you can delete the directory using the `rmdir` command instead!

Task 3 (Optional): The new directory is a better place to practice the navigation commands discussed in the previous section. If you forget where you are on the system, you can always type `pwd` to get the full path of your location.

Now that you feel more comfortable, let's make a little website...

Editing Files

When it comes to editing files, there's a variety of programs available to you. And, just like a programmer's preferred distribution of Linux, it's a large point of contention among us! For this lab, we'll be using Visual Studio Code to do this.

Task 4: Using the `code <path>` command on a file/folder, open up the `index.html` file we created and insert some HTML for your site. Here's a very basic example page:

```
<!DOCTYPE html>
<html>
    <head>
        <title>Linux 101 Test Page</title>
    </head>
    <body>
        <h1>Hello world!</h1>
        <p>This is a test page</p>
    </body>
</html>
```

Once you've done this, you can find your shiny new webpage at:

```
https://dcs.warwick.ac.uk/~u[your student id]
```

...or at least you would, if you had the permissions to access it.

Changing Permissions

```
[u1234567 ~]$ ls -l
-rw-rw-r-- 1 u1234567 dcsugrad 92194 Sep 16 19:37 aaaa.gif
drwx-wx--- 2 u1234567 dcsugrad  4096 Sep 16 19:36 cat_pictures
-rw-r----- 1 u1234567 dcsugrad    42 Sep 16 19:37 important.txt
```

The `l` flag for the `ls` command is very useful, as it shows us detailed information on files and folders. For example, we can see who owns it, the group it belongs to, the size in bytes, and when it was last modified. But what we're interested in is the permissions. They determine not only who can access what, but how:

Permission	Symbol	Meaning
Read	`r`	Can view the contents
Write	`w`	Can change the contents
Execute	`x`	Can run as a program.

Each of these permissions can be applied to three different types of users:

Types	Symbol	Meaning
User	`u`	The file owner
Group	`g`	The user group of the file
Other	`o`	Any other user

In short - the first character indicates if something is a directory, the next three represent user permissions, the next three group permissions, and the final three other users.

These permissions can be modified using the `chmod` command.

The syntax for using `chmod` is extensive, with two main ways to achieve the same goal. The first explicitly states the permissions for each permission group, like so:

```
chmod u=rwx,g=rx,o=r <file path>
```

The second way uses an octal digit to represent each permission group:

```
chmod 754 <file path>
```

Each digit represents user, group and other permissions respectively. These digits are calculated by taking 0, and adding values to it depending on wanted permissions.
+4 for read, +2 for write, and +1 for execute.

The `-R` option can be used to perform a recursive permission change. This will modify all files and sub-directories in the directory you specify with your chosen permissions.

Task 5: Fix the permissions for your web page. For `index.html`, give the user read and write permissions, and let the other two permission groups read. Everything else in the `public_html` directory (including the directory itself) should be set to `rwx` for the user, and `rx` for the other two permission groups. Make sure everything is correct afterwards.

Finally, you'll want to add execute permissions in the home directory itself. Type:

```
chmod o+x .
```

Now, time to see our public web page in all of its underwhelming glory!
This concludes the main part of the lab.

Bonus Tasks

B1: Execute these commands in the `public_html` folder. What do you think they do?

1. `cd -`
2. `ls -lat > output.txt`
3. `grep p index.html`

B2: To help organise your work, try creating a `modules` directory. You might also want to add sub-directories within it for the ones you're going to take, such as `cs118` and `cs126`.

B3: Your web page works, but it might be a bit... lacking? Try improving your website. Feel free to add some HTML or CSS if you like. You could make it look pretty... or make it look cursed.

What next?

Thanks for taking part in the lab! Hopefully you found it useful, and sorry if you didn't... We'd be happy to hear your feedback and suggestions. Make sure to check out the department's in-depth systems guide that includes topics such as remote access.

This is available at https://warwick.ac.uk/fac/sci/dcs/intranet/user_guide/

If you're interested in joining the Computing society, or want to ask any other questions, we'd be happy to welcome you in our community - https://go.uwcs.uk/links

See you around!