# 0 The Linux Operating System

In this class we will be doing our work using the Linux operating system. We will be compiling code using `gcc` and we'll be using `gnuplot` to visualize data. Much of your work can be done on the lab computers, but we have set up a Linux server which you can log into remotely, and do work there. Later on in this document we discuss how this can be carried out. To begin, however, let us get acquainted with the Linux operating system and how to carry out simple commands.

## 0.1 Some Basic Linux Commands

We will be interacting with the Linux operating system mostly via the *command line.* If you are used to Windows or even Macs this will likely be quite a change for you. To access the command line, you must first start the `terminal` program. Look on your desktop – there will be an icon labeled `Terminal` with the > _ symbol displayed. Double click on that. You can now type commands at the Linux prompt $ in that window.

The first command is the *list* command, `ls`. Type that at the command prompt. This will give you a list of all the files and directories (folders) in your home directory. You can use a variation of that command `ls -al` which will give you more details such as when the file or directory was created, its size, and the permissions assigned to that entity.

When you first log into Linux, you will be in your home directory or "folder". You can create subdirectories and subdirectories of those subdirectories under your home directory. The command `cd` enables you to move up and down the directory structure.

Linux commands include:

`cd`: Change directory. Let us suppose there is a directory named `comphys`. To change into that directory, issue the command `cd comphys`.
`mkdir`: Make a directory. You should use this command right now to actually make the directory `comphys` if it does not already exist. You should do all your work for this class in that directory. To create that directory, issue the command `mkdir comphys`.
`pwd`: Now that you have created the directory `comphys`, enter that directory (`cd comphys`). To see where you are in the directory structure, use the

command `pwd`. This will display the full *path* of where you currently are in the directory structure. To go back up one directory (that is, to get back into your home directory) use `cd ../`. Alternately, from wherever you are in the directory structure, you can always get back to your home directory by using cd $\sim$.

`rm`: Remove an entity. This can be used to remove (delete) a particular file or directory. Suppose we have a file named `hello.c` and we want to delete it. It can be deleted with the command `rm hello.c`. Warning! once you have deleted a file there is no way to get it back in Linux. In other words, there is no "recycle bin". If we want to remove a directory and all its contents, use `rm -R comphys`, as an example.

`rmdir`: Remove a directory. If a directory is already empty, this command will remove it: `rmdir comphys`.

`cp`: Copy a file. As an example, let us suppose we have a file named `hello.c` and we want to copy it to `hello2.c`, we use the command `cp hello.c hello2.c`. This will preserve the original file and simply make an exact copy of it under the new filename. Copying can occur from one directory to another. Suppose you have the file `hello.c` in your home directory, and want to copy it to the `comphys` directory. Use `cp hello.c comphys/hello.c`. You will then have two copies of `hello.c`, one in the home directory, one in the `comphys` directory.

`mv`: Rename (move) a file. If you want to rename a file, issue a command like `mv hello.c hello2.c`. This essentially copies `hello.c` to `hello2.c` and then deletes `hello.c`. `mv` can also operate from one directory to another, in much the same way `cp` does.

`cat`: Print out a file to the screen. To look at the contents of a file (print it to the screen), you can use, as an example, `cat hello.c`.

`more`: Print out a file, one screen at a time. If the file is long, `cat` will print all of it to the screen, and you will only be able to see the last bit that fits on the screen. To have more control, use `more`. The command `less`, interestingly, does the same thing.

`head`: Print out only the first few lines of a file (default, 10 lines).

`tail`: Print out the last few lines of a file.

## 0.2 Creating Files

A file can be created with an editor. One nice-to-use editor is `emacs`. Let us create the file `hello.c` which will be a real "C" program. Issue the command `emacs hello.c &` and type the following into the `emacs` window which opens:

```
#include <stdio.h>

int main()
{
  printf("Hello World!\n");
  return(0);
}
```

You should now have a file called `hello.c` in your directory. Practice moving this around using `cp`, `mv`, type it out with `cat`, after making a copy, delete it with `rm hello.c`, etc.

Now we will do something fun. This program is not yet in a form that can be executed by the computer. It first needs to be *compiled* into an executable file. This is carried out by a `compiler` which translates our text-based program into machine language that can be understood and executed by the computer. The compiler that we will be using in this class is `gcc`. To compile `hello.c`, issue the following command:

```
gcc -o hello hello.c.
```

If you now look in your directory (`ls`) you will see not only `hello.c` but the file `hello`. It will probably be highlighted in green. This is an executable program. It can be “run” by typing

```
./hello
```

This will print to the screen:

Hello World!

If you have never done programming before, this is your very first computer program.

## 0.3 Connecting to the Comphys Server

You may use the computers in front of you to do your work in class, but you will not have access to those computers between classes, unfortunately. For this reason, we have set up a computational physics server, a computer that you should be able to log into using your laptop or your own desktop computer (more on that later).

Everything that you can do on the computer in front of you can also be done using a remote connection on that server. As a matter of fact, at the end of each class period, you should transfer the files that you have worked on during class to your account on that server. When you submit your homework, you will also submit it from that server. So, lets review how you can connect to that server from the computer in front of you.

In a terminal window, issue the following command:

```
ssh -Y username@comp1.phys.appstate.edu
```

Where you should replace "username" with your ASU username. So, if your name is Albert Einstein, your username will be something like `einsteina`. Once you have issued this command, in a few seconds you may be prompted with a message asking you if this is a secure connection, to which you answer "y". The server will then request a password. All of your accounts have the initial password `comphys`. Once entered, it will let you in, and you will be in your home directory. You have *logged on* to the comphys server. If you use the command `pwd` the full path of your home directory should look like `/home/einsteina`. Now, lets do some file transfer. On the computer in front of you (the local computer), bring up another terminal window, and `cd` into the directory where you have the file `hello.c`. Then, issue the command

```
sftp username@comp1.phys.appstate.edu
```

It will ask for a password and then you will get the `sftp>` prompt. At that prompt, type

```
put hello.c
```

This transfers the file `hello.c` from the local computer to the comphys server using the *secure file transfer protocol* `sftp`. Now, go back to your other terminal window, where you are logged into the server, and use the `ls` command. You should see `hello.c` sitting happily in your home directory.

You can transfer stuff from the server back to your local computer using the command `get`. For instance, at the `sftp` prompt, change to the *shared* directory on the server with `cd ../shared`. Then `get` the files `comphys.c` and `comphys.h`. Those files should now be on your local computer.

Lets change your password on the server. In the terminal where you are logged into the server, issue the command

```
passwd
```

The server will prompt you for your old password, and then will prompt you for a new password and to repeat it.

### 0.3.1 Submitting Homework

Your homework will be submitted electronically from the comphys server. The homework will be in the form of computer programs, such as `ex12.c`. To submit, first transfer the program to the server using `sftp`, then login to the server, and rename the program, so that the name is of the form `username_ex12.c`. Then, issue the command

```
submit username_ex12.c
```

DO NOT submit executable files, only submit the `.c` files which you have written with an editor. This homework file, along with those of all your fellow students will be transferred to a secure directory on the comphys server. This is why the filename must contain your username, to distinguish it from your fellow students. Once you have submitted a file, there is no way to unsubmit it.

Of course, if you have written your programs on the server, there is no need to transfer them.

Once submitted, there should be a file called `username-submissions.log` in the directory from which you submitted the file. That file contains a record of your submissions. Do not edit this file, as that will invalidate it and so it will no longer be a valid proof that you have actually submitted a given file. You can submit a file more than once; we will only grade the latest version.

Homework will be due at 3pm on the due date, immediately before class. For full credit, you must submit before the due date/time. If your submission is late, an automatic 10 points (out of 100) will be taken off. For each additional day another 30 points will be removed. No homework will be accepted more than two days later than the due date/time!!

Finally, to get out of the `sftp` connection, simply type `exit`. Likewise to log out, simply type `exit`.

## 0.4 Connecting to the Server from Home

If you are running Windows, a basic no-frills connection may be made using a free ssh client, called "putty", at:

`http://www.chiark.greenend.org.uk/ sgtatham/putty/download.html`

Putty runs independent of the Windows registry so once you download it, say to your desktop, you just double-click it and it will run (there is no install required). A typical ssh session would involve logging into the server (you provide the server name in the putty screen) to get a command line. You can then do anything you need to do, such as edit, compile using gcc, or copy (cp) or move (mv) for homework submission. You will find that you cannot use `emacs` with putty, as `emacs` is actually a graphical, X-windows program. But you can use a simple non-graphical editor like nano. For instance, if you want to edit the file `hello.c`, type `nano hello.c`. This will bring up a very simple no-frills editor. To exit and save, press `Ctrl-X`. If you want to use `emacs` remotely, you will have to install an X-windows server. More on that below.

That same source supplies a free sftp client, called psftp. Likewise, download it to your desktop. No installation is necessary.

There are other more sophisticated Windows sftp clients that enable a drag and drop functionality. One is available at
http://winscp.net/eng/download.php

On Mac machines, you should be able to bring up a terminal and use `ssh` and `sftp` as from a Linux machine, because the underlying operating system on a Mac is very similar to Linux/UNIX.

You will eventually want to use graphics-based (X-windows) programs, such as `emacs` or `gnuplot` remotely. To utilize these programs remotely from a Windows machine, you will have to download an X-windows server. One possibility is `Xming`. It can be downloaded from
`http://sourceforge.net/projects/xming`
you will need to download two setup files, Xming and Xming-fonts and install them on your Windows computer. During the install, check the option to install icons for both Xming and Xlaunch on your desktop. Also, during the install of Xming, ask the installer to install Putty. Before you attempt to run Xming, reboot your computer.

Once your computer reboots, you should see two icons on your desktop for Xming and XLaunch. Double click on XLaunch, and select "Multiple Windows". Click Next, and then select "start a program". The program "xterm" is default, and this is what you want. Also click under "Run Remote" "Using Putty", and then enter the name of the server, your username on that server and your password. Click next twice, and then Finish, and then an xterm window should appear on your desktop. From this, you can enter any Linux command, as well as run graphics programs such as gnuplot. If you want a second xterm window, simply type `xterm &` in the first, and hit return. More instructions can be found on the computational physics website.

Another option to using putty or Xming is to actually install Linux on your laptop or desktop computer. If you are interested in doing that, speak to me. Real physicists use Linux.

Yet another option is to purchase a Raspberry Pi. These are small, playing-card sized, but full-fledged computers that run on Linux. For $65.00 you can get a Raspberry Pi along with a power supply, a memory card, an instruction booklet, and an HDMI cable that you can use to attach it to your TV set. You will also need a keyboard and mouse. The latest version of Raspberry Pi comes with built-in WiFi, as well as an ethernet connector, so it is easy to connect it to the internet. The programs that we will be using in this class, `gcc, gnuplot, emacs` can all be installed on a Raspberry Pi for free. Plus you can do some cool interfacing projects with it, such as controlling stepper motors. This means it can serve as the brains of a robot! Highly recommended.