Linux Basics

by Valentina Erastova
Durham University

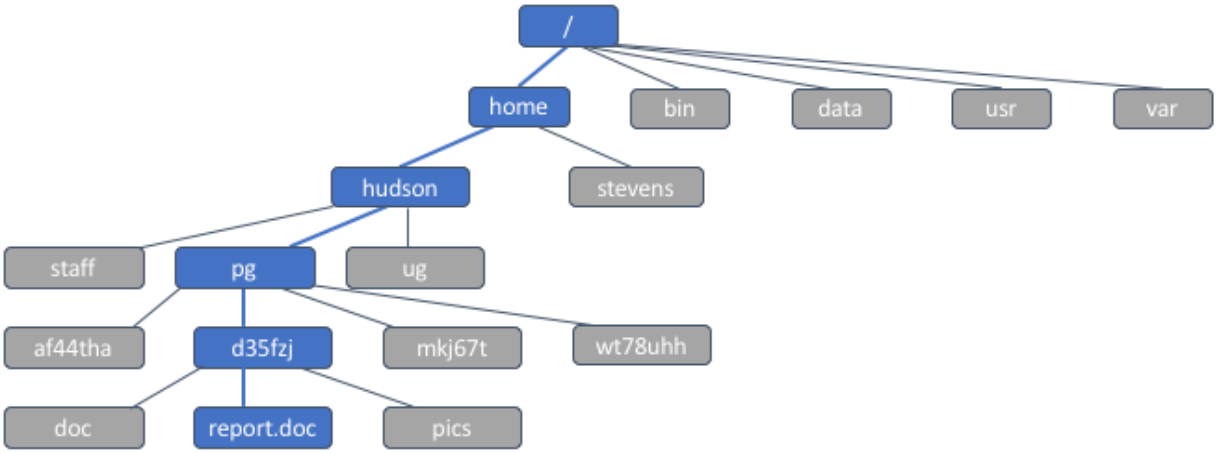
UNIX operating system (OP) was developed in the 80s and is still developing. It is the base of Sun Solaris, GNU/Linux OS and Mac OS.

The OP is made of 3 main parts:

- **kernel**: it allocates time & memory for programs, handles the files & communications.
- **shell**: is a command line interpreter, to interface between the user and the kernel. The shell is started for each user when he/she logs in.
- **programs**: what makes those computers useful for us.

Data organization

Data, such as files and programs, are stored in directories, that are organized in a hierarchical fashion (like a tree, see figure below). The root of the tree is denoted by /. A file is stored in a specific part of the tree. Its position can be defined as the **path** from the root of the tree down until its location. For example, in the picture below, the path to call `report.doc` in the directory `d35fzj` is:
`/home/hudson/pg/d35fzj/report.doc`



Even though UNIX comes with GUI (graphical user interface) we often use a **terminal** to communicate with it. A terminal accepts text commands in a specific syntax (see below), allowing the user to navigate and manipulate the data tree, and call programs in it.

When opening a terminal, the user will be located in a specific directory, called **current working directory (cwd).** This usually corresponds to the **home directory**

of the user, i.e. the directory containing user personal data. In the example above, the home directory of user d35fzj is: `/home/hudson/pg/d35fzj/` Some special symbols defining the directory tree are:

~ : a shortcut for the user home directory
. : indicates the current directory
.. : indicates the current directory's parent directory (upper towards the root)

The user can change the working directory via specific commands (i.e. navigate the directory tree). As we have seen above, the location of files and programs is defined by their path from the root of the tree. This is called the **absolute path**. A second definition of path exists - the **relative path**. This is a path definition with respect of the cwd. For instance, if the cwd is `/home/hudson/pg/d35fzj/pics` the relative path of the file report.doc will be `../report.doc`.

How to name your files?

- **No spaces**
  BAD GROMACS Tutorial
  GOOD GROMACS_tutorial

- **No weird symbols**
  BAD Peter & Mary
  GOOD Peter_Mary

- **Preferably with extensions**
  BAD mydocument
  GOOD mydocument.txt

Using the terminal

This practical requires the usage of a set of basic **Linux commands** to type in the terminal, see the table in the next page. When in doubt about the meaning of a command, for example `grep`, ask the terminal to show you the manual of the command by typing:

```
man grep
```
or
```
whatis grep
```

If you are not familiar with Linux, you can now play around a little bit. Also, you may find this document handy:

https://www.dur.ac.uk/resources/its/info/guides/169linux.pdf

When using command line (terminal), you may need to repeat the command often; you can use arrow key up/down to retrieve the commands you entered previously.

Command	Options	Description
ls		List the contents of the current working directory
	-a	List all files, including hidden ones
	-l	Display long format of listing
	-t	Sort by time, newest 1st
	-r	Display in reverse order
	-h	When in long format, size will be in human readable format (i.e. not bytes)
	-lrth	Multiple options can be combined
	*gro	* is a wild card, will list everything ending with “gro” in the current directory
	../	Will list everything in the above directory
mkdir	[name]	Makes a directory with a desired [name]
cd		Change directory
	../	Up (parent directory)
	Name/.	Down into Name directory
		(i.e. cd without options) take me to my home
	~	also home
pwd		Print the path of the current working directory
cp	file1 file2	Copy file1 to file2
	-R D1 D2	When copying directories use –R (recursive)
mv		Move, options same as copy (also used to rename a file)
rm		Remove. **WARNING** – it will delete whatever you say and there is NO trash bin!
rmdir		Remove directory, same as for above.
more	filename	Displays the contents of a file
head	file	Shows top of the file
tail	file	Shows bottom of the file
	-1000 file	Shows last 1000 lines of the file
grep	text file	Finds and displays the text in a file for every occurrence
	text file |wc	Counts number of text in the file
cat	file1 > file2	Display the content of file1, and send it into file2. **WARNING** - If file2 already exists, it will be overwritten
	file1 >> file2	Appends text of file1 to the end of file2. If file2 does not exist, it gets created.

Working on a supercomputer

A supercomputer is an ensemble of processors performing calculations in parallel. Processors are typically arranged in **nodes**. A node can be seen as a small computer, capable of communicating with others, and sharing the same disk storage space. Calculations can be performed using many nodes simultaneously. These compute nodes are not (normally) shared with other users or jobs (i.e. the supercomputer takes care of allocating nodes to user submitted jobs). Although the supercomputer contains lots of nodes, when connecting to it, the user will connect to a **front end**, i.e. an interface (typically in Linux) to all the nodes of the supercomputer.

Logging into another (super)computer:
`ssh –Y username@hamilton.dur.ac.uk`

Transferring files across to other remote computer, same as `cp` but `scp` (secure copy)
`scp filename username@ham.dur.ac.uk:~/foldername/.`
Use `–r` if transferring directories

You can also transfer from remote computer onto your home one
`scp usrn@ham.dur.ac.uk:~/foldername/filename.doc ./foldername2/.`

From the front end, the user can request the supercomputer to perform calculations. These can be:

- interactive,
- background (denoted with `&` on the end on the command line),
- batch (submitted with a script).

For batch jobs, supercomputers handle resource allocations by means of a **queuing system**. This system helps distributing the computational power in fair and efficient way. The following are useful terminal commands when working on the supercomputer:

Coomand	Options	Description
`sfree`		Gives statistic of the queue
`sbatch`	`jobname`	Submits the job script to the queue
	`-t`	By time, newest 1st
`squeue`		Shows what jobs given user submitted
	`-u \*`	Shows all jobs submitted to the queue
`scancel`	`jobnumber`	Kills the job with the number and all associated ones