A quick intro into a Linux command line.

Andreas Hehn

September 29, 2014

1 Intro

We will discuss how to use a Linux command line. More precisely, we will learn some basic `bash` commands. If you heard about a big security problem with bash (sometimes called "ShellShock") on the news this week, this is exactly the bash we are talking about here. Bash is the most common command line (so called "shell") on Linux, i.e. the standard way to enter commands in Linux. There are also other "shells", e.g. `zsh`. Those are usually very similar to bash and most things will work there in the same way. If you need something which is not mentioned here, usually a simple internet search "How do I ... in bash" will help you. If you get hits going to `stackoverflow.com` or `superuser.com`, click them, they usually have good answers. One advice: please try to understand the solution you find and don't just copy and paste, since you will execute this on your computer or on the ETH cluster and someone might try to put some malware on your system. Otherwise please ask me.

2 Connecting to Euler

Euler is the Computer cluster of ETH. Here's a picture:

Figure 1: Euler: The ETH cluster

It is located at the national computing center [1] in Lugano.

[1] `http://www.cscs.ch`

To connect to Euler you need to be within the ETH network, either by being on-campus or by connecting via VPN to ETH. For VPN please look here: `http://www.vpn.ethz.ch`.

2.1 Windows

WARNING: You may use Windows for this lecture using Visual Studio or any other compiler, but you will have to do the **exam on a Linux computer**. Therefore please familiarize yourself with the Linux installed on the student computers in the computer rooms. You may also consider installing a Linux on your laptop.

To connect to and work on Euler you will need an SSH client. A simple free client for Windows is PuTTY. You can download it from
`http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html`
Just download the `putty.exe`.

To copy files between Windows and Euler you will need a separate program, called WinSCP. You may get it here: `http://winscp.net`. In the next lines I will only discuss PuTTY. WinSCP should be pretty selfexplanatory. If you still run into problems, please contact me.

To connect to Euler you need to go to Category "Session" (on the left) and enter as "Host Name" `euler.ethz.ch`. For Connection type select `SSH`. Then click `open` and you should be asked to enter user name and password. Please enter your ETH nethz account username and password there. Please don't get confused when you enter the password. You won't see any stars while typing or anything. That's normal! Just enter the password and then hit the enter key. When you first connect you have to agree to some terms by writing `Yes`. When you see in the last line something like:

```
 [username@euler01 ~]$
```

you are connected and ready to go.

2.2 Linux

On Linux or MacOS you can easily connect to Euler by opening a terminal and login to Euler with your nethz account typing

```
ssh username@euler.ethz.ch
```

and enter your password (it won't print any stars or anything). When you first connect you have to agree to some terms by writing `Yes`. When you see in the last line something like:

```
 [username@euler01 ~]$
```

you are connected and ready to go.

To copy files between your computer and Euler you use `scp`, which behaves very similar to the bash command `cp`. It is always

```
scp <source> <destination>
```

If you want to copy a file named simulation.cpp to Euler, execute

```
scp simulation.cpp username@euler.ethz.ch:simulation.cpp
```

This will copy the file into your home directory. Important is the `:` which separates the name of the computer that you want to copy to/from from the file name on that computer. If you don't start the file name with `/`, the file name/path you give after the `:` will begin in your home directory. The other way round works the same way, you just have to put the host name to the source instead of the destination:

```
scp username@euler.ethz.ch:simulation_results.txt simulation_results.txt
```

Some graphical file browsers in Linux also provide a way to connect to a remote server via ssh. For example in Ubuntu you may open your home folder and click on File-¿Connect to Server, select Type "SSH" and enter your user name/password. This is equivalent to using `scp`.

3 Help

If you enter

```
help
```

you will get an overview over the bash commands. Appending a command behind `help` will give you more detailed information about the command. E.g.

```
help cd
```

will give you all information about the change directory command.

Beside bash commands there are also programs you can call. You can basically call any program that is installed on the Linux machine from bash. A very simple but useful program is `ls`. It lists the files within the directory you are in. To get help for programs, like `ls` in this case, you can enter:

```
man ls
```

It will show you the man-page which usually explains what the program does and lists possible options to the program. You can scroll the man page with the arrow-keys and you quit the man-page by pressing `q` on the keyboard. If this doesn't help, you may find a lot of stuff on the internet.

4 Directories

Like on Windows you have files and directories (also called folders). As soon as you connected, you "are" in some directory. This is called your (current) working directory. Meaning all the commands you execute will take this directory as a reference, i.e. if you create a new file it will be created in this directory by default. Or if you delete a file it will look for the file name in this directory, not in any other. Usually you start in your home-directory. Which is where you can create your files and store your data. You can always print the working directory by executing

```
pwd
```

which stands for "print working directory". Subdirectories are separated by a slash `/` in Linux.

Also note that Linux is case-sensitive (in contrast to Windows)! You can have two different folders one called `pictures` and one called `Pictures` which will *not* be the same!

5 Useful commands

Very useful: you can auto-complete names by hitting the `<TAB>` key on your keyboard!

`pwd`	Print the current working directory, i.e. where am I?
`help <cmd>`	Print help about command `<cmd>`.
`man <program>`	Show the program manual (manpage) for `<program>`. (Press `q` to exit.)
`cd <directory>`	Change working directory to `<directory>`. You can also go to subdirectories directly by separating them with a `/`.
`cd`	Will change your working directory to your home directory.
`cd ..`	Go one directory up.
`ls`	List all files (and directories) in the working directory.
`ls -l`	same as above, but displays properties of the files too.
`mkdir <directory>`	Create a new directory named `<directory>`.
`mv <from> <to>`	move or rename a file `<from>` to `<to>`. Works with directories, too. If you want to move a file/directory in a directory, but keep the name just add a `/` behind the name of the directory you wrote as `<to>`, e.g. `Pictures/`.
`cp <from> <to>`	copy a file `<from>` to `<to>`. If you want to copy a whole directory with all its contents do `cp -r` for recursive instead.
`rm <filename>`	Remove/delete file. Careful, the file is really gone then. If you want to remove a whole directory and all its contents just add a `-r` like for `cp` again, and give the name of the directory instead of a file.
`exit`	Exit the session.
`nano <filename>`	open `<filename>` with nano, a simple text editor. Keyboard shortcuts are written at the bottom, you can use them by pressing “Ctrl” and the letter written there.
`less <filename>`	display a (text) file, you can use the cursers to scroll. (press `q` to exit.)

5.1 Wildcards

If you need to do something with multiple files, e.g. remove all files that start with `old_file_`, you can use the `*` wildcard:

```
rm old_file_*
```

If you use it with `rm` be careful that you don’t accidentally delete files you wanted to keep!

You can use this with many commands, one more example: copy all `.txt` files from the current working directory to a directory called `notes`:

```
cp *.txt notes/
```

The trailing `/` means that you don’t what to copy the file(s) to a file called `notes`, but that you want to copy it *into* a directory called `notes`.

6 Useful Keyboard shortcuts

Arrow up/down	scan through commands you executed before.
TAB	Auto-complete filename/command.
Ctrl+C	immediately abort running command/program. E.g. useful if you wrote an endless loop and you want to stop it. It usually let's you get out of most things.
Ctrl+R	Will open a search form, which searches in the commands you executed before for whatever you enter. Press `ESC` to exit the search.

7 Compiling and running your own programs

7.1 Compiling

To compile your C++ programs on Linux you can use the GNU-Compiler-Collection 'GCC'. Here are some basic use cases.

```
g++ -o myprogram main.cpp blob.cpp
```

will compile the C++ code `main.cpp` and `blob.cpp` and save the compiled program in `myprogram` (`-o` stands for output). If you don't specify `-o <programname>` your program will be named `a.out`. A little more advanced example

```
g++ -O2 -std=c++11 -o my_picture_program main.cpp -ljpeg
```

Compile `my_picture_program` from the C++ source `main.cpp`, optimize it good (`-O2`) (i.e. "optimization level 2" - you usually want the compiler to optimize your code), use C++11 (`-std=c++11`) and link against the jpeg library (`-ljpeg`), because I want to use the library to load and save jpeg images. There are plenty of options. Have a look at `man g++`.

The reference solutions usually hide all these compiling steps using an (automated) build system, to compile the reference solutions it's usually enough to execute

```
make
```

in the solution folder, which will then do execute all the compilation commands automatically.

7.2 Running

If you try to execute a program (like `ls`) by typing its name in the command line, Linux will only search its officially installed programs. This means if you compile a program in your home folder bash will not find it just by its name. If you want to execute such a program you have to give bash the full path to where to find the program - not just the name. If you are in the same directory as the program this can be done by adding a `./` before the name of the program:

```
./myprogram
```

The `.` means "this directory", and the `/` is the separator between the program name and the directory name - just like for subdirectories.

8 Text editors

I showed you `nano` which is a very simple text editor for the terminal. In principle you may use any editor that generates plain text files for your C++ codes. In the exam you will use a Linux system. There you may use any of the installed graphical text editors (not sure which ones are installed, but I guess gedit, geany, kate,...) or a terminal editor like nano.

If you are interested, there are more advanced terminal editors, which provide for example syntax highlighting, and many many more features. Two very popular ones are `vim`[2] and `emacs`[3]. Both are installed on most Linux systems, are approximately equally powerful and it's approximately equally hard to get used to them in the beginning. In fact one of the big disagreements in the Linux community is if Vim or Emacs is the better editor. Usually you pick one and then stick with it. I use vim...

9 Euler commands/programs

For commands that are specific to Euler, please look at the Wiki of its predecessor, the old ETH cluster Brutus. It used essentially the same software:
`http://www.brutus.ethz.ch`[4]
Especially important are the pages on "Compiling an application" and "Using the batch system", which tell you how to compile and submit a program for execution,respectively. Further information may also be found in the exercise sheets and slides.

10 Further Information and Questions

In principle you should manage to do the lecture with the commands I listed here and the commands shown in the lecture slides and exercise sheets. However, there are some more basic things that are quite useful, so you may have a look at one of the following extended tutorials:
`http://www.software-carpentry.org/v5/novice/shell/index.html`
`http://linuxcommand.org/lc3_learning_the_shell.php`
Especially the chapter on "Redirecting I/O" of the second tutorial is very powerful. It shows how you can e.g. redirect what is usually printed on the screen into a file.

If you have further questions or run into problems just send me a mail `hehn@phys.ethz.ch` or drop by my office HIT G 31.4. Especially if you see something during the lecture or exercises, which you do not understand. Please don't accept it silently, but ask me!

[2]`http://www.vim.org`
[3]`https://www.gnu.org/software/emacs/`
[4]Only accessible from within the ETH network.