PH502 Practical 1

ICHEC

January 12, 2018

1 Overview

This practical provides an overview of UNIX/Linux. It is intended as a "getting started" document for new users or for those who want to know "in a nutshell" what UNIX is all about from a practical user's perspective. It is also intended as the first presentation in a hands-on workshop that introduces programming on the ICHEC systems.

Level/Prerequisites: None.

Learning Outcomes: After the practical session users should be able to:

- Login to the ICHEC Fionn Supercomputer.
- Edit files.
- Use modules.
- Compile simple C or Fortran programs on Fionn.

2 Logging in to the ICHEC systems

You will be using the ICHEC heterogenous cluster - Fionn during this practical. More details of the Fionn supercomputer can be found on the ICHEC website, `www.ichec.ie`, or at the end of this document (Page 11).

Once you have successfully logged in on the local system you can ssh to the Fionn cluster. Temporary ICHEC accounts have been set up for the purpose of this practical, ph5xx## where ## is a number, which you will be assigned. From the command line you can log in as follows:

```
ssh -X username@fionn.ichec.ie
```

Note '-X' or '-Y' can be used in the above command to enable X11 forwarding. This allows graphics to be displayed on your local machine and some of the programs that you will use later require this argument to be given to ssh. (Windows users please check Networking in Page 11).

Please change the password when you log in first using *passwd* commad. Each user home folder is located in /ichec/home/users/.

For today's practical we will carry out all tasks in a folder called practical01. The '!$' argument for cd below takes the argument from the previous command, mkdir and in this case is the name of the folder.

```
mkdir practical01
cd !$ ; pwd
```

3 UNIX Shell

A shell is the interface between you and the UNIX system. The default shell is called "bash" (Bourne Again SHell). There are other shell environments, which one you use is just personal preference. A set of useable shells are listed in the file "/etc/shells". To view or list the contents of a file use 'less' *e.g.*

```
less /etc/shells
```

Each shell can be customizable to suit your preferences. Each time a shell is started, system and user files are examined to setup the shell environment. For the bash shell the user file is ".bashrc" (in your home directory). To go to the home directory: *cd* $\sim$

If a file name starts with a dot (like ".bashrc"), these files are normally invisible. If you type 'ls' in the home directory the file will not be present. The command 'ls' lists the files stored within a directory.

4 UNIX Help pages

The UNIX help pages are a set of single pages per command or utility. Here a page refers to a file rather than a physical page. These pages are grouped into sections: section 1 contains the most relevant commands and section 3 contains help on C functions.

For instance 'ls' lists the files within a directory. To view the UNIX help on 'ls' type

```
man ls
```

You can view a command's man page if you know the command. Type 'q' to quit these man pages. However if you do not know exactly what you are looking for, you are stuck. The man pages can be searched using a keyword. To do this type

```
man -k <keyword>
man -k "list directory"
```

You can see that more than one command matches this keyword. The keyword is enclosed in double quotes because spaces are treated as delimiters. If using more than one word keywords they must be enclosed in double quotes.

What argument to "ls" is required to list a file name starts with a dot (use "man")? Try in your home directory.

5 Editing Files

In this section we will ensure everyone can create, save, open, edit, rename, and remove files. It will be left up to you to chose an editor. Emacs, vi, nano and gedit are available on Fionn. Basic Emacs and vi commands are available on page 12.

Please carry out the following steps:

1. Edit your .bashrc file. If not exist, create a new one.
   Bring up an editor window, using vi (*vi $HOME/.bashrc*), emacs (*emacs $HOME/.bashrc*) or gedit (*gedit $HOME/.bashrc*). "gedit" is recommended but will need to have Xwindows on your machine with Xwindow forwarding enabled.
   Features include:

   (a) Either start a new file or open an existing one. You should be able to type in the edit window, delete and cut and paste with the mouse.

(b) When saving the file it is important to give it the correct extension. If it is a C program then save with the extension ".c". If Fortran then save with extension ".f90".

(c) Some editors detect what type of document you are working with and colour the text accordingly.

2. Add the following line:

```
alias rm='rm -i'
```

3. Quit and save the .bashrc file.
4. Create the file xxx by typing 'touch xxx'. Verify it is there by typing 'ls'.
5. Delete the file using 'rm'.
6. Log out with 'exit' and back in again.
7. Do the same steps again create and delete xxx (Steps 4-6). Is there any difference?

6 Modules

The module system manages software tools/packages (including compilers, libraries) on ICHEC clusters. Some module commands are outlined on page 11 and more info can be found in our tutorial `https://www.ichec.ie/academic/national-hpc/documentation/modules`

1. Describe what you see when you issue the command: *module avail*

2. In order to see and load a particular module, you will need to load the relevant base module first, or in the same command. Load the development module: *module load dev* Describe what you see when you issue the command: *module avail*

3. What does the following command do *module show* module_name?

4. How many C compilers are available?

5. Which C compiler is active after you issue the following command: *module load intel/2017-u3 gcc/6.3.0*?

6. What is the result of *module list*?

7. How do you remove one module? How do you remove all loaded modules?

7 Compilation

In this section you will work through the helloworld example, beginning with the very simplest compile and link. As you have seen there are several compilers available but we recommend using Intel compilers. ("icc" for C and "ifort" for Fortran) To make these available in your shell environment type "module load dev intel/2017-u3". You can edit your ".bashrc" file to load them automatically at login.

1. Both C and FORTRAN are compiled programming languages.

2. This means that the code is written in a human readable form and is converted into a machine readable form, by a compiler.

3. The machine readable form is saved to a different file which can then be executed. Running the program means to load the executable file into memory so that the CPU will execute the program commands.

4. After the program has finished the data stored in memory is not usually accessible. However information can be sent to the screen or to disk as the program is executing.

5. Below is a schematic of the compilation process. Code from different source files are transformed into object files. These object files along with external libraries are linked together to form an executable program.

6. External libraries contain sets of useful functions, usually written by third parties. The C maths library is an example.

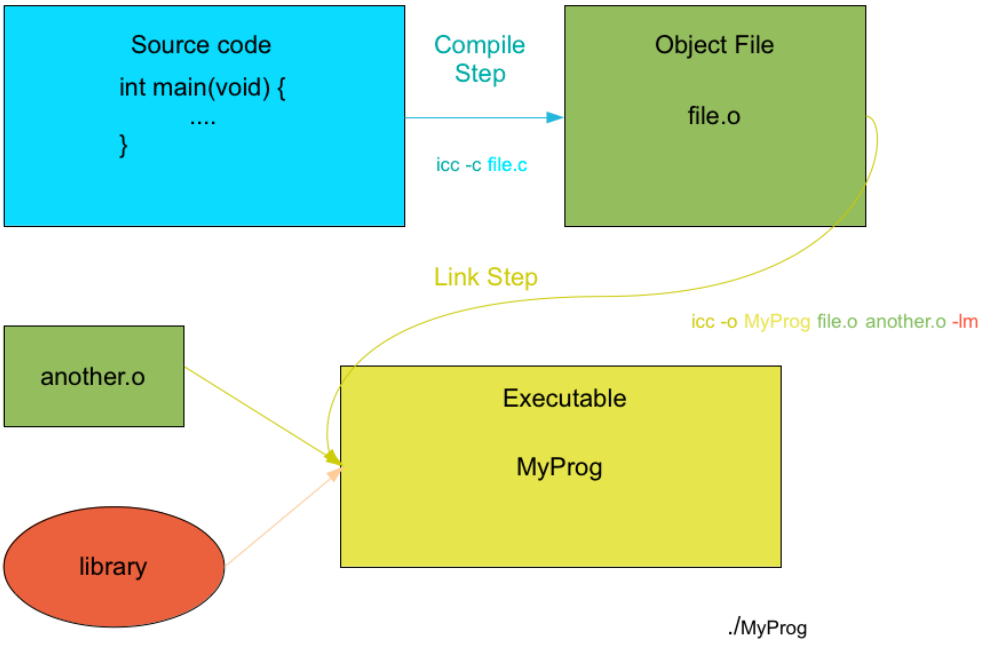

Figure 1. Schematic of the Compilation Process

7.1 Hello World

1. Create the file helloworld.c or helloworld.f90.

   The C program, helloworld.c:

```c
#include <stdio.h>
int main(void) {

    printf("Hello World\n");
    return(0);
}
```

   The FORTRAN program, helloworld.f:

```
!  THIS IS THE MAIN PROGRAM
   PROGRAM helloworld

   WRITE(*,*) 'Hello World'
   END
```

2. Before we run the program it must be compiled first. Here is a example for C and FORTRAN. (Remember to install the correct modules first 'module load dev intel/2017-u3').

```
icc -o helloworld helloworld.c

# or

ifort -o helloworld helloworld.f90
```

3. To run the program type (the ./ tells the loader to look for helloworld in your current working directory; otherwise you may see "file not found" or a "Command not found" message)

```
./helloworld
```

8 Printing Syntax

8.1 Printf Syntax

- printf is used to display information on the screen. It will print variable $r-values$ as they are in memory at the moment when the statement is executed. The basic syntax for printf is:

```
printf(format,variable1,variable2,...);
```

- The number and type of each variable must match those in the format. Each %# corresponds to a variable, with # determining the variable type. Spaces and punctuation appear as written as they appear.

- Below is a table of some of the format symbols.

format	variable type
%d	integer
%.nf	floating point, n decimals
%.ne	scientific notation, n decimals
%c	character
%s	string
\n	new line
\t	tab

- Here is an example below.

```
// default format
printf("Two ints %d %d and two floats %f %f\n", int1, int2,
                                                float1, float2);
// floats with 4 decimal places
printf("Two ints %d %d and two floats %.4f %.4f\n", int1, int2,
                                                    float1, float2);
```

8.2 FORMAT in Fortran

- The format statement in FORTRAN formats the output. Here is an example below.

```
! default format
write(6,*) ' Two ints ',int1,int2,' and two floats ',x1,x2
!
write(6,'(a,2i6,/,a,f8.2,f10.3)') ' Two ints ',int1,int2,&
                                  ' and two floats ',x1,x2
```

- Below is a table of format symbols for FORTRAN:

Symbol	Description
a	character string
iN	integer with N spaces
fN.M	real with N spaces and M decimnals
Nx	N spaces
/	new line

8.3 Exercise

- Copy print.c or print.f90 from /ichec/home/users/ph5xx00/practical01 to your currect directory:

```
cp /ichec/home/users/ph5xx00/practical01/print.c .
```

Print formatted data to stdout.

9 Input with scanf

- scanf is used to read character, string and/or numeric data from keybord. This value is then assigned to a variable.

```
scanf(format, variable);
```

- Example:

```
scanf("%d", &n)
```

- And for Fortran: input stream is unit 5,

```
read(5,*) n
```

10 Exercises

1. Write a simple program to reproduce the following output:
   "Account: 1 Subtotal: 1234.56 Total: 7890.12"
   The account number, subtotal, and total should all be variables.

2. Write a program in C or FORTRAN to sum the first 1000 terms in the sequence $1, 1/2, 1/3, 1/4, \cdots, 1/1000$. Perform the calculation forward *i.e.* from 1, and backwards from $1/1000$. Compare the results.

   To start, copy 2.c or 2.f90 from /ichec/home/users/ph5xx00/practical01 to your currect directory.

3. Calculate the values of "tan(x)" where x is in radians in the range $[0, 60]$ every five degrees. Place the results in an array. Print the array.

   - One radian is equal to $\frac{180}{\pi}$ degrees. Write a function to convert from degree to radian.
   - When compiling with C use $\#include < math.h >$ and compile with "-lm". This will ensure the tan function is available.

   To start, copy 3.c or 3.f90 from /ichec/home/users/ph5xx00/practical01 to your currect directory.

4. One possible way to compute the area under the curve $f(x)$ with $x \in [a, b]$ is to use the Trapezodial rule:

$$\int_a^b f(x)dx \sim \frac{b-a}{2N}(f(x_0) + 2f(x_1) + 2f(x_2) + \cdots + 2f(x_{N-1}) + f(x_N))$$

   where $x_0 = a$ and $x_N = b$ with $N-1$ equidistant points between a and b.

   Use the formula above and the array in Exercise 3 to calculate the area under the curve $y = tan(x)$ from $0 \rightarrow \pi/3$. Compare with the actual result: $\int_0^{\pi/3} tan(x)dx = log(2)$.

   Hint: $x_i = i$ and $f(x_i) = Tan[i]$ for $i = 0, 1, ..., 11, N = 12$.

   To start, copy 4.c or 4.f90 from /ichec/home/users/ph5xx00/practical01 to your currect directory.

11 Further ICHEC Information and Support

The website `www.ichec.ie` provides:

- Information on Training Courses
  `https://www.ichec.ie/academic/training-education/graduate-modules`
- User support information
  `https://www.ichec.ie/academic/national-hpc/user-support`
- Hardware and software details
  `https://www.ichec.ie/about/infrastructure`
- General information about ICHEC
  `https://www.ichec.ie/academic`

Quick Reference Guide

Getting Help

- **man** *program* (manual pages for *program*)
- **module load** *application* (initialize the environment of an application)
- **module avail** (list applications on current server)
- **module list** (list of loaded applications)
- **module purge** *application* (remove application environment)
- **FAQ:** `http://www.ichec.ie/support/faq`

Unix Commands

- **ls** (list directory)
- **less** *file* (print a file to the screen)
- **cp** *file1 file2* (copy a file)
- **rm** *file* (delete a file)
- **mv** *file1 file2* (move or rename a file)
- **cd** *dirname* (change the current directory)
- **cat** (sends the file to standard output)
- **pwd** (print name of the current directory)
- **mkdir** *dirname* (create a directory)
- **rmdir** *dirname* (delete a directory)
- **exit** (quit the session)
- **passwd** (change password)
- **history** (list all commands given previously)
- **!stuff** (executes the last command that started with "stuff")
- **head** *file* (list ten first lines of the file)
- **tail -100** *file* (list the last hundred lines of the file)
- **tail -f** *file* (keeps listing the end of file. Handy for following an output file when lines are appended to it.)
- **grep** *stuff file* (print lines containing the word stuff from the file)
- **diff** *file1 file2* (lists file differences)
- **ls -la** > *file* (output of a command to a file)
- **ls -la** | **grep** "word" (chaining (piping) multiple commands)
- **tar cvf** t.tar t.* (make a tar-file t.tar from all files whose names begin with t. You can also tar a directory.)
- **tar xvf** t.tar (extract all files from the tar-archive t.tar)
- **gzip** t.tar (compress file t.tar to save space)
- **gunzip** t.tar.gz (uncompress file t.tar.gz)

Computers

- **Fionn:** *fionn.ichec.ie*
  - The Thin component is an SGI ICE X system of 320 nodes or 7680 cores made of 2.4GHz Intel Ivy Bridge cores. Each node has 2x12 core processors, 64GB of RAM and is connected using FDR InfiniBand.
  - The Hybrid partition contains 32 nodes. Each of these nodes has 2x10 core 2.2GHz Intel Ivy Bridge with 64GB of RAM. 16 nodes have 32 Intel Xeon Phi 5110P's while the other 16 have 32 NVIDIA K20X's.
  - The Fat section is an SGI UV2000 where 1.7TB of RAM is accessible to 112 cores of Intel Sandy Bridge (14x8 cores processors) and 2 Intel Xeon Phi 5110P's.

File Transfer

- **scp** *computer1:file1 computer2:file2* (copy files from computer1 to computer2)
- An example of scp usage:
  scp *temp.txt username@fionn.ichec.ie:*. (copies the file temp.txt (from current directory to fionn) Because the directory in the target machine was not specified the file goes to the home directory of *username*.)
- Most ssh-clients also have a graphical file transfer program available. Ex: WinSCP in Windows.

Networking

- **ssh** *computer* (open a new secure session)
- In Windows you will need an ssh-program. Ex: Putty
- In Linux use **ssh -X** *computer* or **ssh -Y** *computer* to enable X-connection
- In Windows to enable graphical X-windowing choose "Forward X11" from your ssh-program settings. You will also need a separate X-emulator program, e.g. XMing

Paging With less

- **less** file (print a file to the screen)
  - **[return]** (next line)
  - **[space]** (next screen)
  - **b** (previous screen)
  - **/stuff [enter]** looks for the next occurrence of "stuff"
  - **h** (list the commands of less)
  - **q** (quit the less program)

Unix Files Permissions

- **ls -l** (long list of files present. First column details the file permissions e.g. drwxr-xr- -)
  - Missing privileges are represented with a '-'.
  - **First** character: (file type: d = directory, - = file, l = link)
  - **2 - 4th** character: read/write/execute for **user**
  - **5 - 7th** character: read/write/execute for **group**
  - **8 - 10th** character: read/write/execute for **others**
- File permissions are altered by giving the chmod command and the appropriate octal code for each user type.
  Read = 4
  Write = 2
  Execute = 1
- **chmod 755** *file* (Full permission for the owner, read and execute access for the group and others.)
- **chmod +x** *file* (Make the file called filename executable to all users.)

Emacs Editor

- **emacs** *file* (start the emacs editor)
- **emacs -nw** *file* (emacs without X-windows)
- Notation **[Ctrl]-c** means: "hold down the Control key and press the c key"
- Moving: cursor keys and page up/down keys
- **[Ctrl]-x [Ctrl]-c** (quit and save)
- **[Ctrl]-x [Ctrl]-s** (save)
- **[Ctrl]-g** (interrupt an emacs command if you get stuck in the minibuffer)
- **[Ctrl]-h [Ctrl]-h** (Emacs help system)
- Other text editors are e.g. **nano**, and **vi**

VIM Editor

- **vi** *t.txt* (open file named t.txt)
- Two modes: insertion mode and command mode.
- **[ESC]** returns the editor to command mode
- **Inserting Text**
  - **i, I** (insert before cursor, before line)
  - **a, A** (append after cursor, after line)
  - **o, O** (open new line after, line before)
  - **r, R** (replace one char, many chars)
- **Deleting Text**
  - **x, X** (character to right, left)
  - **D** (to end of line)
  - **dd** (line)
- **Searching and Replacing**
  - **/w, ?w** (search forward/backward for w)
  - **/w/+n** (search forward for w and move down n lines)
  - **n, N** (repeat search (forward/backward))
  - **:s/old/new** (replace next occurrence of old with new)
  - **:s/old/new/g** (replace all occurrences on the line)
  - **:%s/old/new/g** (replace all occurrences in file)
  - **:%s/old/new/gc** (same as above, with confirmation)
- **Undoing**
  - **U** (undo changes on current line)
  - **u** (undo last command)
- **Quitting**
  - **:x or :wq** (exit, saving changes)
  - **:q** (quit (unless changes))
  - **:q!** (quit (force, even if unsaved))

System Status

- **ps** (process status)
- **top** (continuous process status)
- **uptime** (show the load of the computer)
- **who** (list logged-in users)
- **whoami** (display current user)
- **date** (print data & time)
- **finger** *user* (gives information about user)
- **df -kh** (disk status in human readable units)
- **du -kh** (disk space used by a directory)
- **qsub**, **qstat**, **qdel** (submit, get status of, and cancel batch jobs in torque)
- **lfs quota -u your_username /ichec/home** *project* (List your user quota)

Finding files and text within files

- **find** / *-name fname* (Starting with the root directory, look for the file called fname)
- **find** / *-name "*fname*"* (Starting with the root directory, look for the file containing the string fname)
- **grep** *textstringtofind /dir* (Starting with the directory called dir , look for and list all files containing textstringtofind)

Program Development

- Compilers on ICHEC machines (Fortran, C): GNU, Intel
- use the module command to check the version and to load the environment. It will also put libraries in the path.
- An example of compiling a program with gcc:

  module load dev gcc/6.3.0
  cc -o prog -fast prog.c
  Run the program: ./prog

How to contact ICHEC

- WWW homepage: `http://www.ichec.ie`
- Staff: `www.ichec.ie/about_us/people`
- Helpdesk: `http://helpdesk.ichec.ie/`
- Dublin Office Address:

  ICHEC
  7th Floor Tower Building
  Trinity Technology & Enterprise Campus
  Grand Canal Quay
  Dublin 2, Ireland
  T: +353 1 5241608 F: +353 1 7645845
- Galway Office Address:

  ICHEC
  IT302, IT Building
  NUI Galway
  Galway City, Ireland
  T: +353 91 495305 F: +353 91 495573