CS240 Operating Systems, Communications and Concurrency

Dermot Kelly

Introduction to Unix

The operating systems course practicals involve **learning aspects of Ubuntu operating system. Ubuntu** is a computer operating system based on the Debian GNU/Linux(Unix) distribution and distributed as free and open source software. It is named after the Southern African philosophy of *Ubuntu* ("humanity towards others") - wikipedia.

Over the course of the labwork you will become familiar with Linux commands and utilities, writing command scripts for the shell, using some C and/or Java programming to test various features of the operating system such as process management and interprocess communication and working on some concurrency problems. The practicals focus on the Unix process model and interprocess communication mechanisms. Access to these mechanisms both at the shell level(command prompt) and from within a running program(API library calls) are examined. Using the command line interface allows us to extend the command interface using scripts unlike the fixed commands offered by a GUI.

You should keep the practical handouts for reference as each practical builds on previous work. Always read the handout and carry out the instructions contained therein to accumulate credits for each practical. The University Library contains many texts on Unix operating system versions which provide useful supplementary reading and some of these are identified on the course' Moodle page.

The laboratory workstations are capable of booting under Ubuntu or Windows. **You must ensure that the workstation is running the Ubuntu operating system before you can proceed.** You may need to restart the machine and select Ubuntu at startup.

Access to the system requires the entry of a username and password. The same password that you use to access the system using Windows is used for access to Ubuntu. However, you may not be able to change your password under Ubuntu. To change your password you may need to be running Windows and press CTRL-ALT-DELETE.

IMPORTANT: Before you leave, you must disconnect your current session as other users may damage your files if you remain logged in and they should use the computer after you. To disconnect your session enter **exit** or **logout** at the shell prompt. Alternatively access the logout function through the graphical user interface.

Linux uses a style of disk partitioning that differs from Windows. The names given to Linux partitions are derived from the type of the drive, the drive letter and the partition number (e.g. hda1 instead of, for example, C: in windows). The relationship between the partition which stores the associated data and the directory path name used to access it is set up through the mounting process where various accessible partitions are attached at various points in the file tree hierarchical name space when the system is booted. As a result, although the files are stored on a number of separate physical disks, they are unified and accessible transparently to the user under a common file name space.

Linux uses the forward / character where Windows uses the backward \ character in path names. For example, “dir c:\” in Windows is equivalent to “ls /” in Linux.

Typical Linux Partition Organisation

/	root partition stores Linux installation and program files
/boot	stores Linux kernel
/home	used to store personal files. This maps to your network X Drive.
/usr	used to store program files
/tmp	used to store temporary files
/var	used for variable sized data including unprocessed email
/swap	used as extra memory by working applications

The Window Manager
The version of Ubuntu/Linux used in the laboratories uses the GNOME window manager/desktop environment (www.gnome.org). The GNOME project provides two things: The GNOME desktop environment, an intuitive and attractive desktop for end-users and the GNOME development platform, an extensive framework for building applications that integrate into the rest of the desktop.

Unix Commands - (Note: Commands are case sensitive)
Here are a sample of some unix shell commands that may be entered at a command shell prompt "$". A shell process is a command parser that interprets the commands of a shell scripting language. Under the **Applications/Accessories** menu you can open a new terminal window which is attached to a shell command parser.

The commands listed below represent binary executable utility programmes located somewhere in the shell’s search path. The shell locates the executable binary image and creates a child process to execute the command. These commands can be included as part of shell script programs which allow the use of variables, specification of selection and iteration statements as well as redirecting the source and destination of data. More on this later. **TRY OUT EACH OF THE COMMANDS LISTED BELOW**. Open a terminal window and enter the commands after the $ prompt.

General Utility Commands

pwd	Print the current path and directory
cd dirname	Change directory to the directory dirname in the current directory
date	Print date and time
man	Print entries from the on-line manual (ASCII type terminal browsing tool) e.g man pwd (exit manual by pressing q)
who	List the current users of the system
xterm	A standard terminal emulator for X Windows enabling access to a Unix command shell
gnome-terminal	Alternative terminal emulator enabling access to a Unix shell on the graphical desktop
vi filename1	A text based editor (No GUI) Edit or create a file called filename1 (Exit by pressing ESC followed by :q)
emacs filename2	A graphical desktop coding editor - Edit or create a file called filename2
gedit filename3	A graphical desktop text editor - Edit or create a file called filename3
more filename	Browse or page through a file - press space bar to advance through file
cat filename	Read a text file and write it to standard output
lp filename	Print file on the laboratory printer

File Management

cp filename1 filename4	Copy filename1 to filename4
ls	List the files in the current directory
mkdir dirname	Create a new directory called dirname within the current directory
mv filename4 filename5	Move or rename filename4 to filename5, different name or location
rm filename1	Delete filename1
rmdir dirname	Delete directory dirname

Process Management

kill processid	Send a signal to a process to indicate it should terminate
ps	Display list of processes and their status
time	Time a process execution, e.g. time ls

To find out more about the option settings and parameters to these commands, make use of the online manual facility. For example, try to find out what **ls -l** does by entering the command **man ls** and reading the explanation of parameters allowed to ls.

Try **ps -e** and then read the manual to explain what the '-e' flag did. Most of the utilities have optional flags which are indicated by various letters following the '-' character. More than one option may be specified, e.g. **ps -el**

From the process listing of the last command, pick a process id which has 'gnome-terminal' opposite it. Using the **<processid>** value given opposite ‘gnome-terminal’ in the first column, type **kill <processid>** What happened?

Editing

There are a number of text editors available for creating and editing files. **gedit** is the official text editor of the GNOME desktop environment. It has a graphical user interface and is probably easier to use for those new to Linux. It is accessible under the applications/accessories menu.

Another editor you can use is the versatile character oriented editor known as vi. To use the vi editor, type **vi** at the shell prompt. The vi editor operates in one of two modes. In *command* mode, it accepts commands, for example to save the file or delete lines or to move the cursor to a particular part of the file. In *text* mode, it accepts character data to be entered into the file at the cursor position. The editor begins in command mode. To enter text mode type **i** or **a**. **i** causes characters to be inserted at the cursor position, while **a** causes characters to be inserted after the cursor position. To get out of text mode and back to command mode press the ESC key. You will have to do this to save your file and exit the editor.

Note: The arrow keys can be used to position the cursor when in text mode. Most commands can be preceeded by a number which causes the command to be carried out that many times.

vi commands

i	Enter text mode, at cursor
a	Enter text mode, after cursor
l	Move cursor right one character
h	Move cursor left one character
j	Move cursor down one line
99j	Move the cursor down 99 lines
k	Move cursor up one line
w	Move cursor right one word
b	Move cursor left one word
$	Move cursor to end of line
^	Move cursor to start of line
x	Delete character at cursor
dd	Delete current line at cursor
7dd	Delete 7 lines from cursor position
:r filename	Insert contents of filename at cursor position
:q!	Exit from editor without saving changes
:wq	**Exit** from editor **saving changes**

As the best choice, you should consider using Emacs. **Emacs** is a powerful, customizable code editor with a graphical user interface. To use emacs, type emacs at the shell prompt.

For the duration of this practical you should try to become familiar with the Linux environment and learn the usage of some of the more common Unix commands listed on this handout. During the course of the practicals, you will be writing some code and so some familiarity with the character based editors **vi** or GUI editor **emacs** or **gedit** will be necessary. You should practice creating and manipulating some simple text files using each of the editors listed and decide which one you prefer.

Unix Fundamentals

You should read and carry out the instructions of the section entitled 'Introduction to Unix' before continuing. Attempt the exercises below and document the result of your attempts, where possible, on paper. You may need to hand this up to the demonstrator at the end of the session.

Remember, when using the Unix system, an online help facility is available to you that explains the meaning and usage of Unix commands and gives related command information. At the command prompts type:-

```
man <command>
```

To begin with, **create a directory called CS240** for this course and make it your current directory.

1. Create four files called **file1**, **file2**, **file3**, **file4** in the directory **CS240**. Enter a line into each of the n files of the form 'This is file n' where n is the appropriate number.

2. Copy the four files to four files of the same name but with the extension **.bak**

3. Edit file1.bak and change it to 'This is a copy of file1'

4. Using the cmp command, compare file2 with file2.bak and file1 with file1.bak

   i.e.
   ```
   cmp file1 file1.bak
   cmp file2 file2.bak
   ```

5. Get a list of files in the root directory '/'

6. What does the command **ls /d*** do?

7. Remove all files with **.bak** extension from directory **CS240**.

8. Make a directory called **text**.

9. Move all the files, **file*** into **text**.

10. List files in directory text.