Basics of Unix Commands

Module 01 Stage 01

CS110

Department of Computer Science and Engineering
Indian Institute of Technology Guwahati
Semester II, Academic Year 2017 - 2018

1 Introduction

Welcome to the first lab training session. Primary objective of this session is to help everyone to get started with the Unix environment. This session makes only one assumption that each student can understand basic instructions given in simple English.

In this session, we will go through the following assignments -

- Starting a Terminal session
- Basic commands for file and directory operations
- Working with an Editor
- Compiling a given C program file

At the end of the session, we expect each student should be able to do the following tasks without taking any external help -

- able to login
- able to open a terminal session
- able to move across directory structure
- create a new directory
- create and edit a new file
- edit an existing file
- able to read Man pages of Unix commands
- compile a given C program
- execute the compiled program

Let us get over any fear and enjoy learning. Please do not hesitate to ask the assigned tutor if you face any difficulty during the practice sessions.

2 Starting a Terminal Session

The first practical task to perform during this session is to open a terminal session. It is assumed here that you are already able to successfully login into the system and are probably looking at something like the Fig 1. The launcher (leftmost vertical panel) contains shortcut to multiple

Figure 1: Blank Desktop - Your screen immediately after your login. Background image and launcher position may differ but it will be something very similar.

applications. At the top of this panel, you see a Ubuntu logo. Clicking on that brings the **search bar**.

In case you face difficulty in login, please cross check whether you have entered correct *login* and *password* (Both *guest* are in lower case letters). The correct details are as follows:

login: guest
password: guest

Now, let us focus on our first task. Remember, there may be multiple ways to do this task and for that matter most of the tasks can be done in more than one ways. As an illustration, I will start a terminal session in two different ways. You have to find the one which is the easiest for you.

Method 1: To starting a terminal window, steps are as follows:

I. Click on Ubuntu logo present at the top left corner of the launcher. It opens a search bar.

II. Type `terminal`

III. Choose the appropriate icon to launch a terminal session.

Method 2: Another way to launch a terminal session is as follows:

I. Find *Alt* and *F2* keys on your keyboard

II. Press both the keys together

III. Type `gnome-terminal`

IV. Click on the appropriate icon to launch a terminal session.

Hopefully, you are able to open a terminal using one or both the methods. A terminal window will appear with a command prompt $ near the top-left corner of the terminal window. There might be something more written before $, but lets not worry about that at this moment.

You may find the 2nd method little difficult. Again, do not worry. Take help of the assigned tutor to make sure you practice both these methods. We will see next, how your life can be made easier.

Can I make my life little easier? We always look for making our life easier and comfortable. Let us try to make an easier way to launch a terminal session. Can you guess what would be a solution, if not how exactly to do that? We already see that launcher contains shortcuts for few applications. We should be able to create a shortcut for *launching a terminal* as well. Lets do that.

You will see a terminal icon appears on the launcher panel when you start a terminal session. **Right click** on that icon, and click on **Lock to Launcher** option. It then permanently locks the *terminal* icon to the launcher. Next time, you login to the system, you can start the terminal session by directly clicking on the *terminal* icon on the launcher.

If you already see the *terminal* icon on the launcher, then try to remove it first and then bring back the same icon again.

3 Basic Commands

Opening a terminal session was not at all difficult task, but it was interesting to know at least three different ways to do the same task. What to do with this terminal session now? In this section, we will explore some basic commands. Commands are themselves a program, designed to execute specific task.

Listing file and directories

Every user who logs into a linux system, works from a *directory* (also referred to as *folder*). This directory is called **home directory** to the user.

Now, we will explore how to list the contents of this directory. This task will be accomplished by command **ls**. For your easy reference, you can remember **ls** as "list contents". As the name suggests, **ls** command lists the contents of your current working directory.

To run this command, you have to type `ls` and press `Enter` on your terminal. For example

```
$ ls
```

You do not have to type $. As discussed earlier $ is a command prompt appearing by default on the terminal screen. This is how computer tells you that it is ready to take next command. You have to type only **ls**.

After pressing `Enter`, you will see a list of file and/or directory names on the terminal. It may be possible that, there are no files or directories present in the current directory and in that case, the command prompt $ will appear without listing anything.

Creating directories

We will now make a directory to hold all files you will be creating during the today's practice session. For this, we will have to use the command **mkdir**. Lets create a directory with the name following YYYYMMDDX convention. YYYY is for *Year* in four digits, MM is for Month in two digits, DD is for Date in two digits, and X will be *M* if your lab session is in the morning otherwise it will be *A*. You guessed it right, you have to type

```
$ mkdir 20180111M
```

if you are attending lab in the *morning* session on 11th January. Your directory name will change as per the date and the session of your lab.

Now to check whether the directory has been created, type

```
$ ls
```

Do you see your new directory name appearing on the screen? We created a new directory for our personal use without cluttering other users.

Changing directories and knowing your current working directory

You have created a new directory but you may be wondering how to work from that directory, i.e., how to see what files/directories are there or how to create another file or directory inside the new directory. There must be a command to move across directory structure. And that command is **cd** (remember as acronym of change directory). Lets learn to use this command by typing

```
$ cd 20180111M
```

This command is expected to move you to the directory named **20180111M**, but how do we know whether this has been really done? **Again, remember** that the directory name will differ based on your lab session. This document is written by taking one example. **You need to appropriately modify the directory name**.

To find out your location in the directory structure, you can take help of another command **pwd** (remember as "print working directory"). **pwd** prints the current working directory.

Lets type

```
$ pwd
```

You will see something like `/home/guest/20180111M` appears on the terminal screen. The command **pwd** prints the directory name in relation to the whole file-system with / being the root directory, *home* being the top directory, *guest* being a sub-directory of *home* and *20180111M* being the current working directory, which is also a sub-directory of *guest*. At this time, you only need to focus on output part beginning at directory *guest*.

Command Summary

Command	**What it does**
ls	list files and directories
mkdir *newdir*	create a directory named "newdir"
cd *destination_directory*	change to directory named "destination_directory"
pwd	print the full path of the current working directory

4 Getting started with Text Editors

In this subject, we are going to write programs. The programs need to be present as a computer file for use on a computer. C programs are text files readable to humans and created by programmers.

These files will be translated to executable versions that can run on a computer. You will see some of these backgrounds covered in the lecture as well as in next practice session onwards.

Our interest at the moment is in creating a text files containing C programs. We will create such file with the help of a **text editor**. A **text editor** is used for editing plain text files. All C programs are written as plain text files. Although there are multiple options available for such editors, we will use a text editor, called **gedit**, as it is easy to use.

Starting gedit

Lets first observe how to open **gedit**. If you have understood the basic steps followed in opening "a terminal session", you can follow the exactly the same steps to start **gedit**. For making it simple for you, I reiterate the Method 1 again here.

Method 1: To open **gedit**, you have to follow steps described below:

I. Click on Ubuntu logo present on the launcher. It opens a search bar.

II. Type `gedit`

III. Choose the appropriate icon to launch gedit.

An alternate method using terminal window

Alternative Method: To open **gedit** using a command on a **terminal window**, follow the steps given below:

I. Type `gedit &` at the command prompt, and press enter i.e.,
$ `gedit &`

Creating a new file

Let us suppose we want to create a C program file **welcome2cs110.c**. In case, **gedit** is already opened, then we can start typing directly in it. Let us type the following c code:

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

```
   printf("Welcome to CS110 course\n");
   printf("We hope you enjoy exploring and learning this course\n");
}
```

After typing the above C code, now you save this file by either pressing `ctrl` and `S` keys together or by clicking on "**Save**" option at the top right corner of the **gedit** window. You appropriately choose a directory and save with the file name **welcome2cs110.c**.

Note 1. It is a requirement that all c program file names end with .c

We could have performed the same task by typing `gedit welcome2cs110.c` & at the command prompt of the terminal session. It will look like this

$ `gedit` *welcome2cs110.c* &

This will open a **gedit** window. If there is no file named **welcome2cs110.c** present in the current working directory, the opened **gedit** window will be empty, i.e., without any content; otherwise the opened **gedit** window will show the content of the existing file **welcome2cs110.c**.

5 Compiling and Executing a C program file

Once your c program file is ready, you can compile and execute it to see whether it is giving correct output. We will follow the two steps for that.

Step 1. Type `gcc` **welcome2cs110.c** at the command prompt and press `Enter`, i.e.,
$ `gcc` *welcome2cs110.c*

Step 2. Type `./a.out` at the command prompt and press `Enter`, i.e.,
$ `./a.out`

The first step compiles the c program and creates an executable file with default name **a.out**. To run the program, we need to run the executable file **a.out** by following the 2nd step.

Compilers are very unforgiving of errors in the code. If your program does not compile, please open your C program file (in **gedit** editor) and make sure that your text is exactly as in the our example above. Some common mistakes are

- forgetting putting semicolon at the end of a statement.
- mismatch of number of opening and closing brackets.

6 Some more useful commands

Copying and Moving a file or a directory

You have by mistake created the new file with incorrect name. For illustration purpose, let us consider the incorrect name is **welcomecs110.c**. Fortunately, we have multiple ways to correct the error.

First we will copy the incorrect file to a file with a correct name using a command **cp**. And then delete the incorrect file using a command **rm**. So let us do that

```
    $ cp welcomecs110.c welcome2cs110.c
Now lets remove the file with incorrect name.
    $ rm welcomecs110.c
```

The above two steps can be combined in one step using a command **mv** for moving a file.

```
    $ mv welcomecs110.c welcome2cs110.c
```

The commands **cp** and **mv** can be used to copy or move file from one directory to other directory as well.

Removing a file or directory

Lets come to the last task of the session by removing the directory created by you. For this we can use either **rmdir** or **rm** command. But first make sure you are at one level up the directory hierarchy from the directory you want to delete. For example, if you have created the directory **20180111M** inside **/home/guest** and want to delete **20180111M**, you have to make **/home/guest** as current working directory. You can check your current location with the command **pwd** and move across directory path using the command **cd**.

Assuming that your current working directory is **/home/guest**, you can use either **rmdir** or **rm** command as follows:

```
$ rmdir 20180111M
```

If the directory **20180111M** is not empty, then you will see an error message. To avoid this error message, you have to first make the directory empty, i.e., you have to delete all files and directories within this directory. Use **cd** command to go inside the directory. We will delete all files using **rm** command. Assuming that **welcome2cs110.c** file has been created in this directory, lets delete this file as follows:

```
$ rm 20180111M
```

Now go back to one directory above the current directory. You can use **cd** command in the following manner:

```
$ cd ..
```

Run the **rmdir** command again to delete the directory **20180111M**. This time, there should be any error.

The command **rm** can be used more efficiently to delete a directory. Please check it carefully:

```
$ rm -r 20180111M
```

Here command **rm** uses an option **-r**, which recursively deletes all contents of a directory. In other words, you don't have to go inside the directory to make it empty. It will remove all of its contents.

Note 2. Unlike other operating systems, Unix does not support recovery of removed files or directories in generic setting. Please double check the file/directory name before pressing **enter** on **rm** command.

Knowing a command in more details

You may wonder how to remember these commands and their various options. Do not worry. You do not have to remember the full details of it. You only have to remember the name of the command. Other details can be obtained from **Manual** page of a command. You can access **manual** page of a command using another command **man** as follows:

```
$ man ls
$ man cd
$ man pwd
$ man rm
```

7 Advice for the next week lab session

Hurray! you have done with the first major hurdle. We are sure now your all fear working with command line are vanished. You feel very confident. Good.

The tasks included in next drill will require a lot of work. It is strongly advised that you read the drill document much before coming to the lab session. In fact, you must do some practice before coming to the lab session. Next lab session will involve assessment problems as well.

8 Comments and Suggestions

Please send your comments and suggestions for this document to anand.ashish@iitg.ernet.in