Lab01:Getting familiar with Linux

Objectives

1. **Executing some of the most frequently used Linux commands**
2. **Familiar with text editor like vim and nano**
3. **Understanding directories in linux**
4. **Understanding and changing file permission and ownership**
5. **Compile and run C programs**

Executing a Linux Command

From the command prompt simply type the name of the command:
`$ command`
Where $ is the prompt character for the Bourne shell.
Or if the command is not on your path type the complete path and name of the command such as:
`$ /usr/bin/command.`

1. pwd

Description: Displays the name of the current directory.pwd stands for present working directory. By typing this command you are informed of which directory you are currently in.
Syntax: pwd

2. cd

Description: Changes the current directory to any accessible directory on the system.
Syntax: For instance to change from /home/user1 to a subdirectory of user1 wordfiles use the following:

```
$ cd wordfiles
```

3. ls

Description: Displays the listing of files and directories. If no file or directory is specified, then the current directory's contents are displayed. By default the contents are sorted alphabetically.
Syntax: To view the contents of user1 home directory use this:

```
$ ls
```

4. USING THE man PAGES

The man pages are manual pages provided in a standard format with most Linux software. Almost all the commands that ship with Red Hat Linux distribution include man pages. Using the man command in its most basic form, any existing man page can be read:
$ man *command-name*

5. echo

Description: This command is used to print the arguments on the screen .
Syntax: $echo <text>

6. who

Description: It is used to display who are the users connected to our computer currently
Syntax: $who – option‟s

7. clear

Description: It is used to clear the screen.
Syntax: $clear

8. mkdir

Description: To create or make a new directory in a current directory .
Syntax: $mkdir <directory name>

9. cd

Description: To change or move the directory to the mentioned directory .
Syntax: $cd <directory name>

10. rmdir

Description: To remove a directory in the current directory & not the current directory itself
Syntax: $rmdir <directory name>

11. Cat

a. With > operator

Description: To create a new file in the current directory
Syntax: $cat > <filename>

b. Without > operator

Description: To display the content of file mentioned
Syntax: $cat <filename>

12.sort

Description: To sort the contents in alphabetical order in reverse order.
Syntax: $sort <filename >

13.PIPE

Description: It is a mechanism by which the output of one command can be channeled into the input of another command.
Syntax: $cat file.txt | sort

14.cp

Description: To copy the contents from source to destination file . so that both contents are same
Syntax: $cp <source filename> <destination filename>
$cp <source filename path > <destination filename path>

15.mv

Description: To completely move the contents from source file to destination file and to remove the source file.
Syntax: $ mv <source filename> <destination filename>

16.rm

Description: To permanently remove the file we use this command
Syntax: $rm <filename>

17. sudo apt-get update

Description: update the packages database

18.sudo apt-get install vim

Description: get and install vim editor

19.vim or nano editor

Description: text editor
Syntax: $vim <filename>
$nano <filename>

Hand on vim and nano

20.script

Description: write treminal's data intofile.
Syntax: $script <filename>
Note: Write exit to end writing.

Important Directories in the Linux File System
Most of the directories that hold Linux system files are "standard." Other UNIX systems will have identical directories with similar contents. This section summarizes some of the more important directories on a Linux system.
/
This is the root directory. It holds the actual Linux program, as well as subdirectories. Do not clutter this directory with your files!
/home
This directory holds users' home directories. In other UNIX systems, this can be the /usr or /u directory.
/bin
This directory holds many of the basic Linux programs. bin stands for binaries, files that are executable and that hold text only computers could understand.
/usr
This directory holds many other user-oriented directories

File Permissions and Ownership
All Linux files and directories have ownership and permissions. You can change permissions, and sometimes ownership, to provide greater or lesser access to your files and directories. File permissions also determine whether a file can be executed as a command.
If you type ls -l or dir, you see entries that look like this:

-rw-r—r— 1 fido users 163 Dec 7 14:31 myfile

The -rw-r—r— represents the permissions for the file myfile. The file's ownership includes fido as the owner and users as the group.

File and Directory Ownership
When you create a file, you are that file's owner. Being the file's owner gives you the privilege of changing the file's permissions or ownership. Of course, once you change the ownership to another user, you can't change the ownership or permissions anymore!

The chown command
Use the chown (change ownership) command to change ownership of a file. The syntax is chown <owner> <filename>. In the following example, you change the ownership of the file myfile to root:

```
$ ls -l myfile
-rw-r—r— 1 fido users 114 Dec 7 14:31 myfile
$ chown root myfile
$ ls -l myfile
-rw-r—r— 1 root users 114 Dec 7 14:31 myfile
```

To make any further changes to the file myfile, or to chown it back to fido, you must use su or log in as root.

File Permissions

Linux lets you specify read, write, and execute permissions for each of the following: the owner, the group, and "others" (everyone else).

read permission enables you to look at the file. In the case of a directory, it lets you list the directory's contents using ls.

write permission enables you to modify (or delete!) the file. In the case of a directory, you must have write permission in order to create, move, or delete files in that directory.

execute permission enables you to execute the file by typing its name. With directories, execute permission enables you to cd into them.

For a concrete example, let's look at myfile again:

```
-rw-r—r— 1 fido users 163 Dec 7 14:31 myfile
```

The first character of the permissions is -, which indicates that it's an ordinary file. If this were a directory, the first character would be d.
The next nine characters are broken into three groups of three, giving permissions for owner, group, and other. Each triplet gives read, write, and execute permissions, always in that order. Permission to read is signified by an r in the first position, permission to write is shown by a w in the second position, and permission to execute is shown by an x in the third position. If the particular permission is absent, its space is filled by -.
In the case of myfile, the owner has rw-, which means read and write permissions. This file can't be executed by typing myfile at the Linux prompt.
The group permissions are r—, which means that members of the group "users" (by default, all ordinary users on the system) can read the file but not change it or execute it.
Likewise, the permissions for all others are r—: read-only.
File permissions are often given as a three-digit number—for instance, 751. It's important to understand how the numbering system works, because these numbers are used to change a file's permissions. Also, error messages that involve permissions use these numbers.

The first digit codes permissions for the owner, the second digit codes permissions for the group, and the third digit codes permissions for other (everyone else).
The individual digits are encoded by summing up all the "allowed" permissions for that particular user as follows:

Read permission	4
write permission	2
execute permission	1

Therefore, a file permission of 751 means that the owner has read, write, and execute permission (4+2+1=7), the group has read and execute permission (4+1=5), and others have execute permission (1).
If you play with the numbers, you quickly see that the permission digits can range between 0 and 7, and that for each digit in that range there's only one possible combination of read, write, and execute permissions.

Changing File Permissions
To change file permissions, use the chmod (change [file] mode) command.
The syntax is chmod <specification> file.
There are two ways to write the permission specification. One is by using the numeric coding system for permissions:
Suppose that you are in the home directory.

```
$ ls -l myfile
-rw-r—r— 1 fido users 114 Dec 7 14:31 myfile
$ chmod 345 myfile
$ ls -l myfile
—wxr—r-x 1 fido users 114 Dec 7 14:31 myfile
$ chmod 701 myfile
$ ls -l myfile
-rwx——x 1 root users 114 Dec 7 14:31 myfile
```

This method has the advantage of specifying the permissions in an absolute, rather than relative, fashion. Also, it's easier to tell someone "Change permissions on the file to seven-five-five" than to say "Change permissions on the file to read-write-execute, read-execute, read-execute."

You can also use letter codes to change the existing permissions. To specify which of the permissions to change, type u (user), g (group), o (other), or a (all). This is followed by a + to add permissions or a - to remove them. This in turn is followed by the permissions to be added or removed. For example, to add execute permissions for the group and others, you would type:

```
$ chmod go+r myfile
```

Other ways of using the symbolic file permissions are described in the chmod man page.

How to Compile and Run a C Program on Ubuntu

Step 1: Use a text editor to create the C source code

```
#include <stdio.h>
main() {
printf("Hello World \n");
}
```

Close the editor window.

Step 2: Compile the program.
Type the command
gcc -o first first.c
This command will invoke the GNU C compiler to compile the file first.c and output (- o) the result to an executable called hello.

Step 3: Execute the program
Type the command
./first
This should result in the output Hello World

Write a program which assign two different numbers to variables, then sum two variable and print it.