**Linux Administration Command Guide**

**1)** **Creating a user account using useradd command on Ubuntu**

# sudo useradd -s /bin/bash -d /home/ktpl -m ktpl

-s /bin/bash – Set /bin/bash as login shell of the new account

-d /home/ktpl/ – Set /home/ktpl/ as home directory of the new Ubuntu account

-m – Create the user’s home directory

**2) Set a Password for newly added User**

# passwd ktpl

**3) File and Directory Permissions**

**\* Ownership of Linux files \***

Every file and directory on your Unix/Linux system is assigned 3 types of owner, given below.

**\* User \***

A user is the owner of the file. By default, the person who created a file becomes its owner. Hence, a user is also sometimes called an owner.

**\* Group \***

A user- group can contain multiple users. All users belonging to a group will have the same access permissions to the file. Suppose you have a project where a number of people require access to a file. Instead of manually assigning permissions to each user, you could add all users to a group, and assign group permission to file such that only this group members and no one else can read or modify the files.

**\* Other \***

Any other user who has access to a file. This person has neither created the file, nor does he belong to a user group who could own the file. Practically, it means everybody else. Hence, when you set the permission for others, it is also referred as set permissions for the world.

Let us understand the Permission system on Linux.

**\* Permissions \***

Every file and directory in your UNIX/Linux system has following 3 permissions defined for all the 3 owners discussed above.

* Read: This permission give you the authority to open and read a file. Read permission on a directory gives you the ability to lists it’s content.
* Write: The write permission gives you the authority to modify the contents of a file. The write permission on a directory gives you the authority to add, remove and rename files stored in the directory. Consider a scenario where you have to write permission on file but do not have write permission on the directory where the file is stored. You will be able to modify the file contents. But you will not be able to rename, move or remove the file from the directory.
* Execute: In Windows, an executable program usually has an extension ".exe" and which you can easily run. In UNIX/ Linux, you cannot run a program unless the execute permission is set. If the execute permission is not set, you might still be able to see/modify the program code (provided read & write permissions are set), but not run it.

**ls -l** command on terminal gives File type and Access Permissions like below

ravi@ravic-pc ~/Desktop $ ls -l file.txt

-rw-rw-r-- 1 ravi ravi 0 Nov 29 16:49 file.txt

Here, the first '-' implies that we have selected a file. Else, if it were a directory, d would have been shown like below.

ravi@ravic-pc ~ $ ls -ld Desktop/

drwxr-xr-x 14 ravi ravi 4096 Nov 29 16:49 Desktop/

The characters are pretty easy to remember.

r = read permission (4)

w = write permission (2)

x = execute permission (1)

- = no permission (0)

**\* Changing file/directory permissions with 'chmod' command**

We can use the 'chmod' command which stands for 'change mode'. Using the command, we can set permissions (read, write, execute) on a file/directory for the owner, group and the world.

Syntax: chmod permissions filename/directoryname

There are 2 ways to use the command

1) Absolute (Numeric) Mode

2) Symbolic mode

Number Permission Type Symbol

0 No Permission ---

1 Execute --x

2 Write -w-

3 Execute + Write -wx

4 Read r--

5 Read + Execute r-x

6 Read + Write rw-

7 Read + Write + Execute rwx