**Study of Linux File Permissions**

horizontal line

Linux, like other Unix-like operating systems, allows multiple users to work on the same server simultaneously without disrupting each other.

Individuals sharing access to files pose a risk exposing classified information or even data loss if other users access their files or directories. To address this, Unix added the file permission feature to specify how much power each user has over a given file or directory.

**● Check Permissions in Command-Line with Ls Command**

If you prefer using the command line, you can easily find a file’s permission settings with the ls command, used to list information about files/directories. You can also add the –l option to the command to see the information in the long list format.

To check the permission configuration of a file, use the command:

**ls –l [file\_name]**

For instance, the command for the previously mentioned file would be:

**ls –l test.txt**

It shows the permission settings, grouped in a string of characters (-, r, w, x) classified into four sections:

1. File type. There are three possibilities for the type. It can either be a regular file (–), a directory (d) or a link (i).
2. File permission of the user (owner)
3. File permission of the owner’s group
4. File permission of other users

● **Using Chmod Command to Change File Permissions**

As all Linux users, you will at some point need to modify the permission settings of a file/directory. The command that executes such tasks is the chmod command.

The basic syntax is:

**chmod [permission] [file\_name]**

There are two ways to define permission:

1. using symbols (alphanumeric characters)
2. using the octal notation method

* **Define File Permission with Symbolic Mode**

To specify permission settings using alphanumeric characters, you’ll need to define accessibility for the user/owner (u), group (g), and others (o).

Type the initial letter for each class, followed by the equal sign (=) and the first letter of the read (r), write (w) and/or execute (x) privileges.

To set a file, so it is public for reading, writing, and executing, the command is:

**chmod u=rwx,g=rwx,o=rwx [file\_name]**

To set permission as in the previously mentioned test.txt to be:

• read and write for the user

• read for the members of the group

• read for other users

Use the following command:

chmod u=rw,g=r,o=r test.txt

* **Define File Permission in Octal/Numeric Mode**

Using the octal notations table instead of ‘r’, ‘w’, and ‘x’. Each digit octal notation can be used for either of the group ‘u’, ‘g’, or ’o’.

So, the following work is the same.

**chmod ugo+rwx [file\_name]**

chmod 777 [file\_name]

Both of them provide full read, write and execute permission (code=7) to all the group.

**● Changing User File and Group Ownership**

Aside from changing file permissions, you may come across a situation that requires changing the user file ownership or even group ownership.

Performing either of these tasks requires you first need to switch to superuser privileges. Use one of the options outlined in the previous passage.

To change the file ownership use the chown command:

**chown [user\_name] [file\_name]**

Instead of [user\_name] type in the name of the user who will be the new owner of the file.

To change the group ownership type in the following command:

**chgrp [group\_name] [file\_name]**

Instead of [group\_name] type in the name of the group that will be the new owner of the file.