File Access Permissions

There are three types of files in LINUX. They are **directory** files, **ordinary** files and **special** files. We will be dealing with directory and ordinary files only. The output of the ls -l command shows the details clearly. Ordinary files start with **"-"** and the directory files start with **"d"**. Hence in the following example **girl** and **air.c** are ordinary files while **ashadir** is a directory file.

Example 2.10

```
$ ls -l
total  3
-rw-r--r--   1   anu   student    10   Jan    1     10:39   girl
drwxrwxr--   2   anu   student    80   Jan   10     15:30   ashadir
-rwxrwxrwx   1   anu   student    40   Jan   13     20:40   air.c
$ _
```

When any user creates a file, the creator is said to be the owner of that file. We can perform any operation like delete, edit or copy on that file. If a user wants other people to access his/her files, then permission has to be granted by the owner of the file. This way LINUX helps in the security of files.

There are various types of permissions available. They are read (r), write (w) and execute (x).

Read permission is used to display, copy or to compile a file. Write permission is used to write, edit or to delete a file. Execute permissions are used to execute a file.

Associated with any LINUX file is the owner of the file, the group which consists of users who need to share that file and others who do not belong to that group.

The ls -l option gives the list of permissions granted to each file. The first column of the ls -l command gives a list of the permissions granted to all those associated with any LINUX file.

The permissions are given from the second position onwards. The first three characters indicate the permissions of the owner of the file. The next three positions indicate the permissions of the group and the last three the permissions for others.

Example 2.11

```
$ ls -l
total  3
-rw-r--r--  1  anu  student   10  Jan   1    10:39  girl
drwxrwxr--  2  anu  student   80  Jan   10   15:30  ashadir
-rwxrwxrwx  1  anu  student   40  Jan   13   20:40  air.c
$ _
```

No read permission will mean that we cannot list the contents of the directory and we cannot remove the directory using rm -r option.

No write permission will mean that we cannot make a subdirectory, remove a subdirectory or move files to other directories.

No execute permission will mean that we cannot display the contents of the directory, change to the directory, display a file in that directory and copy a file in that directory.

Changing the FAP of a File

We can change the mode of any file or directory using the **chmod** command.

Continuing with the above example of the output of ls -l option, let us take the air.c file. Suppose the user wants to revoke the execute permission , then the command is,

```
$ chmod -x air.c
$ _
```


If the user wants to grant the execute permission, then

```
$ chmod +x air.c
$ _
```

The granting and revoking of permissions can be done together also like +wx for write & execute permission and -wx for revoking write and execute permissions.

FAP can be changed for one particular category or for all users. This is done by specifying the name of the user before the +/- sign.

'u' - granting or revoking of permissions for the owner of the file only.
'g' - granting or revoking of permissions for the group who needs to share that file only.
'o' - granting or revoking of permissions for others only.

Octadecimal Representation of FAP

File Access Permissions can be represented Octadecimally. Octadecimal representation means representing the values with eight as base. Given below are the octadecimal values

Numbers	Values	Numbers	Values
0	000	4	100
1	001	5	101
2	010	6	110
3	011	7	111

Table 2.1

If a file has the permission set, then the value allotted is 1 otherwise the value is 0. Suppose a file has the following permissions **rwx------**, then this means that the owner has values 111, the group has 000 and others have 000. The value corresponding to 111 is 7and this represents the **user**, the value corresponding to 000 is 0, which represents the **group** and the other value corresponding to 000 is 0 which represents **others**. Hence the Octadecimal value for this file is 700.

```
$ chmod 777 text (granting all permissions for all for the file text)
```