File Permissions

All the three owners (user owner, group, others) in the Linux system have three types of permissions defined. Nine characters denotes the three types of permissions.

1. **Read (r) :** The read permission allows you to open and read the content of a file. But you can't do any editing or modification in the file.
2. **Write (w) :** The write permission allows you to edit, remove or rename a file. For instance, if a file is present in a directory, and write permission is set on the file but not on the directory, then you can edit the content of the file but can't remove, or rename it.
3. **Execute (x):** In Unix type system, you can't run or execute a program unless execute permission is set.But in Windows, there is no such permission available.

**Permissions are listed below:**

|  |  |  |
| --- | --- | --- |
| **permission** | **on a file** | **on a directory** |
| r (read) | read file content (cat) | read directory content (ls) |
| w (write) | change file content (vi) | create file in directory (touch) |
| x (execute) | execute the file | enter the directory (cd) |

Permission Set

**File permissions for (-rw-rw-r--)**

|  |  |  |
| --- | --- | --- |
| **position** | **characters** | **ownership** |
| 1 | - | denotes file type |
| 2-4 | rw- | permission for user |
| 5-7 | rw- | permission for group |
| 8-10 | r-- | permission for other |

When you are the **User owner**, then the user owner permission applies to you. Other permissions are not relevant to you.

When you are the **Group** then the group permission applies to you. Other permissions are not relevant to you.

When you are the**Other,** then the other permission applies to you. User and group permissions are not relevant to you.

Setting Permissions With chmod

You can change the permissions with chmod command accordingly to your need. Below are some examples to change the permissions for different groups.

To add permissions to a group.

**Syntax:**

1. chmod **<groupName>**+**<permissionName>** **<fileName>**

.

To remove permissions from a group

**Syntax:**

1. chmod **<groupName>**-**<permissionName>** **<fileName>**

**Example:**

1. chmod g-x file
2. chmod u-w file

To add permission to all the groups together

**Syntax:**

1. chmod a+**<permissionName>** **<fileName>**

**Example:**

1. chmod a+w file

To add permission to all the groups without typing a

**Syntax:**

1. chmod +**<permissionName>** **<fileName>**

Look at the above snapshot, this example is same as the earlier one only difference is that we haven't typed **a** in this.

To set explicit permission

**Syntax:**

1. chmod **<groupName>**=**<permissions>** **<fileName>**

**Example:**

1. chmod o=rw file

To set explicit permissions for different groups

**Syntax:**

1. chmod **<groupName>**=**<permissions>** **<fileName>**

**Example:**

1. chmod u=rwx,g=rw,o=r file

Setting Octal Permissions

Octal permissions can also be set for the groups.

For example, to set**r** octal will be**4,** to set**w** octal will be**2,** to set**x** octal will be**1.**

**Octal Table:**

|  |  |  |
| --- | --- | --- |
| **binary** | **octal** | **permissions** |
| 000 | 0 | --- |
| 001 | 1 | --x |
| 010 | 2 | -w- |
| 011 | 3 | -wx |
| 100 | 4 | r-- |
| 101 | 5 | r-x |
| 110 | 6 | rw- |
| 111 | 7 | rwx |

From this we can conclude that,

1. 777 = rwxrwxrwx
2. 765 = rwxrw-r-x
3. 654 = rw-r-xr--

and so on.

umask

While creating a file or directory, by default a set of permissions are applied. These default permissions are viewed by **umask** command.

For safety reasons all Unix systems doesn't provide execution permission to newly created files.

Adding execution permission is upto you.

**mkdir -m**

The 'mkdir -m' command can be used to set the mode.

**Syntax:**

1. mkdir -m **<mode>** **<fileName>**

**cp -p**

The 'cp -p' command preserves the permissions and time stamps from source files.

**Syntax:**

1. cp -p **<sourceFile>** **<destinationFile>**