File and Directory Permissions

Understanding File Permissions

In Linux, every file has three basic permissions: read, write, and execute. These permissions are represented by the letters r, w, and x, respectively.

Let's understand this with an example. Last time, we used the ls command to list all the files in our current working directory. Let's do the same, one more time.

Now, the first line in the row, represents some alphabets. The same alphabets we just talked about. So, these alphabets are categorized in three categories. The first category shows permissions for the user or owner of the file, the second category shows permissions for the group to and the last one is for all other users.

As we can see, some of the permissions are starting off with alphabet d but some are starting off with a -. Those which are starting with the "d" are the directories, which we can also identify by the blue colour and the ones starting off with a - sign are the files.

Changing File Permissions with chmod

Now we will see how we can change file permissions. We can do this using the chmod command which allows us to add or remove permissions for the owner, group, or other users.

There are two ways of doing this - one is the symbolic way and the other one is the binary or numerical way.

Let's see the symbolic way first

For that, let's create a new file called test.sh with some command in it. Now those who don't know, .sh represent a shell file which is kind of an executable script in linux. But when we try to to execute it, it gives us a permission denied error.

But Why's that ?

Let's find out.

Looking at the permissions of our concerned file, we can see that it do not have any executable permission set to it. Let's change that.

So, we will use the chmod command.

we will specify on which we want to perform the changes - user, group or others. and then we will add the x permission to it.

```
chmod u+x test.sh

./test.sh
```

Nice! We have successfully executed our script.

To remove this permissions, all we have to do it is to replace the + sign with a - one.

```
chmod u-x test.sh
```

Now if we want to give execute permissions to all the three categories - the users, group and others. We can do that too.

```
chmod ugo+x test.sh
```

Same can be done for read and write permissions also. Just replace them with x or just add all of them at once. Like:

```
chmod ugo+rwx test.sh
```

Now let's see how we can do the same using the numerical way.

- 4: read permission
- 2: write permission
- 1: execute permission

For example if we want to set read, write and execute permissions for user, read and write permission for group and only read permission for others. We can do this by

```
chmod 764 test.sh
```

So, whenever we have to add some permissions using the numerical way. We just have to add the permissions numerical way to make up a digit for user, group and others. Here the first digit was 7 which was made by adding numerical number of read, write and execute. Same was done for group and others column.

Understanding Directory Permissions

Directory permissions work slightly differently than file permissions. The read permission (r) allows users to list the contents of the directory, while the write permission (w) allows users to create, delete, or rename files within the directory. The execute permission (x) is required to navigate into the directory using the `cd` command.

Changing Directory Permissions with chmod

Changing directory permissions follows the same principles as changing file permissions, but with a different interpretation of the permissions. Let's see with an example

- First we will make a directory called scripts. Now let's give read, write and execute permission for user and only read and execute permission for groups and others.

```
mkdir scripts


chmod 755 scripts
```

- We can also give recursive permission to a directory by specifying the -R flag. That means, all the subdirectories and files within that directory will also have a same permission. For example - let's give read and write permissions for user and only read permission for groups and others.

```
chmod -R 644 scripts
```
