User Groups and Permissions in Linux

short line

# **User Groups and Permissions in Linux**

From smartphones to cars, supercomputers and home appliances, home desktops to enterprise servers, the Linux operating system are everywhere.

### Introduction

To create a secure environment in Linux, you need to learn about user groups and permissions. For example, if you work in a company and you want the finance department to read a file but not make any modification to it, then you need to use permissions in Linux. It is a must for every programmer working with Linux nowadays.

### Prerequisites

To follow along with this tutorial, you should have:

* Familiarity with the Linux operating system.

### 

### Table of contents

* File permissions.
* User accounts.
  + Create a user.
  + Delete a user.
* User groups.
  + Create a group.
  + Add user to a group.
  + Delete user from a group.
  + Delete a group.

### 

### 

### 

### 

### 

### 

### File permissions

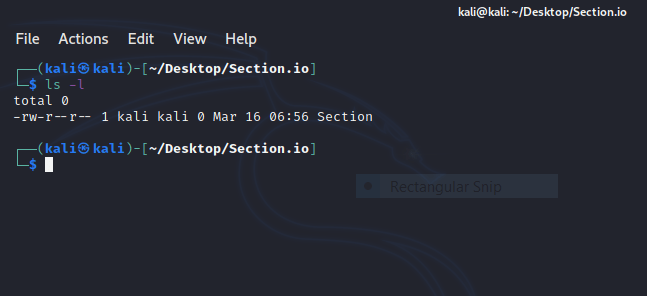
Let’s start by talking about the ownership of Linux files.

1. User: the owner of the file (person who created the file).
2. Group: the group can contain multiple users. Therefore, all users in that group will have the same permissions. It makes things easier than assign permission for every user you want.
3. Other: any person has access to that file, that person has neither created the file, nor are they in any group which has access to that file.

When you perform the following command:

ls -l

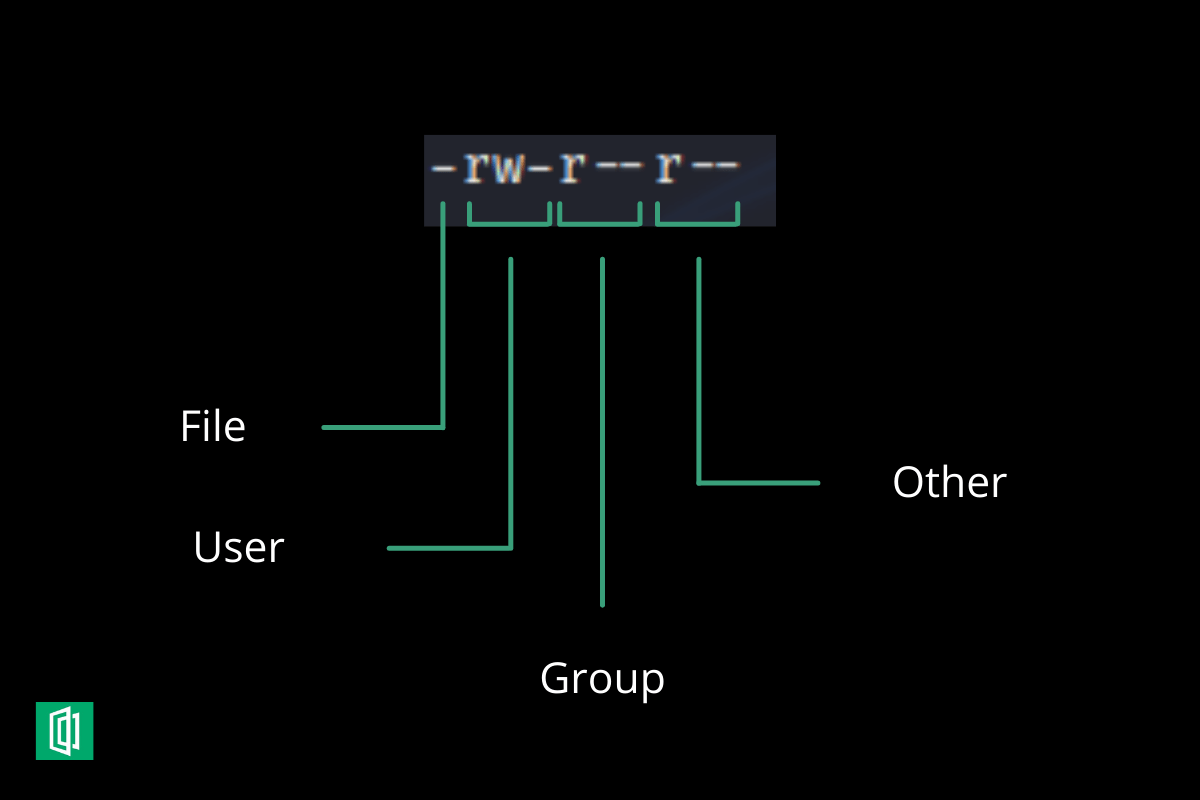
Then you will see the file’s permissions, like the following:



We will work with this part “-rw-r–r–".

The characters mean:

* ‘r’ = read.
* ‘w’ = write.
* ‘x’ = execute.
* ‘-’ = no permission.



As we see above, the empty first part means that it is a file. If it were a directory then it will be the letter “d” instead. The second part means that the user “Home” has read and write permissions but he does not have the execute one. The group and others have only the read permission.

Let’s change the permissions using the chmod command.

chmod o+w section.txt

This command will add the write permission for other users to my text file **“section.txt”.**

Now if you try to execute ls -l then you will see **-rw-r--rw-.**

**“o”** refers to others, **“g”** for the group, **“u”** for the user, and **“a”** for all.

Now let’s add the execute permission to the user with:

chmod u+x section.txt

The permissions will be **-rwxr--rw-**.

If you want to remove the permission, you can use the same method but with **“-”** instead of **“+”**. For example, let’s remove the execute permission from the user by:

chmod u-x section.txt

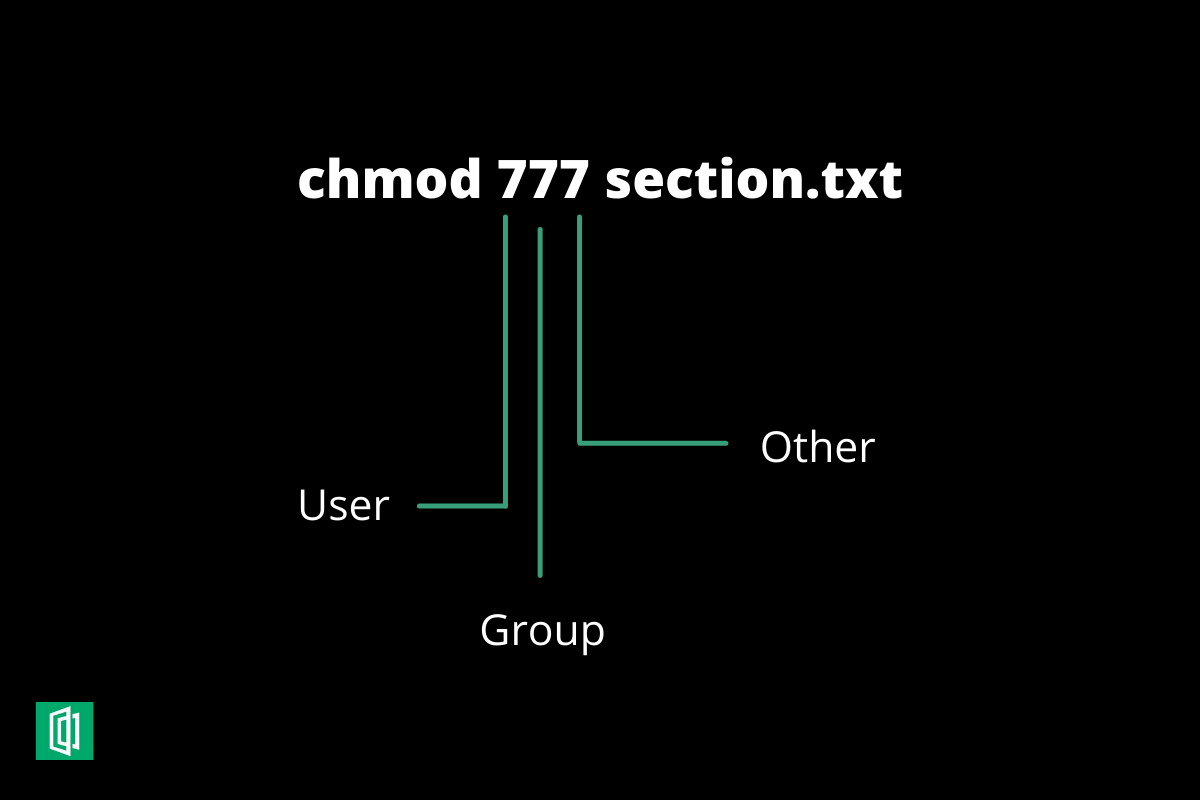
And the permissions now are **-rw-r--rw-**.

Also, you can use Symbolic Mode to modify permissions like the following:

| **Number** | **Permission** |
| --- | --- |
| 0 | No permission |
| 1 | Execute |
| 2 | Write |
| 3 | Execute and Write |
| 4 | Read |
| 5 | Read and Execute |
| 6 | Read and Write |
| 7 | Read, Write and Execute |

For example, let’s give every permission for all with:

chmod 777 section.txt



Then the permissions will be **-rwxrwxrwx**.

Let’s remove the execute from the group and the write from other by:

chmod 765 section.txt

The permissions will be -rwxrw-r-x.

### 

### User accounts

#### 

#### **Create a user**

We can create a new user account by issuing the following command:

sudo useradd testuser

We can make sure that the user has been created in two ways:

1. id testuser .

And the output will be something like this:

uid=1007(testuser) gid=1009(testuser) groups=1009(testuser)

This will show the user id and the groups that the user is currently in, usually, a new group with the same username is assigned to the user.

1. By opening the following file: /etc/passwd.

So we can issue cat /etc/passwd and we will see the new user that has been created.

After creating the user using the command above, you notice that no user directories have been created inside /home directory, which is not good since the user cannot log in to his account.

To create a new user with its directories, we can issue:

sudo useradd -m -s /bin/bash testuser

If you navigate to the /home directory, you notice that a new directory with the name *testuser* is created.

Afterward, you need to set a new password to the testuser by:

sudo passwd testuser

We noticed that creating a new user takes a lot of commands to accomplish, so there is a command that automates everything:

sudo adduser testuser

After creating a new user and setting a password to it, you can log in in two ways:

1. Through GUI.
2. By the terminal: su - testuser.

#### 

#### 

#### 

#### 

#### 

#### 

#### **Delete a user**

As the process of adding users, there are two commands that delete a user.

sudo userdel testuser

If you try that command, you will notice that the user directory has not been deleted and you need to delete it by yourself.

You can use this automated command to do everything for you:

sudo deluser --remove-home testuser

### 

### 

### 

### 

### User groups

A group is a collection of users. The [primary purpose](http://learningwithpradeep.blogspot.com/2020/05) of the groups is to define a set of privileges like read, write, or execute permission for a given resource that can be shared among the users within the group.

##### Create a group

You can see all of the groups you have by opening the following file:

cat /etc/group

Let’s create a group with the name of a *section* by:

sudo groupadd section

#### Add user to a group

We will add the *testuser* user to the section group by:

sudo usermod -aG section testuser

#### 

#### Delete user from a group

You can delete the testuser from the group with:

sudo gpasswd -d testuser section

#### Delete a group

Let’s delete the previous group by:

sudo groupdel section