Linux Commands

How to Change File Permissions in Linux

6 months ago • by Prateek Jangid

Linux works well as a multiuser operating system. Many users can access a single OS simultaneously without interpreting each other. However, if others can access your directories or files, the risk may increase.

Hence, from a security perspective, securing the data from others is essential. Linux has features to control access from permissions and ownership. The ownership of files, folders, or directories is categorized into three parts, which are:

- **User (u):** This is the default owner, also called the file's creator.
- **Group (g):** It is the collection of multiple users with the same permissions to access folders or files.
- **Other (o):** Those users not in the above two categories belong to it.

That's why Linux offers simple ways to change file permissions without hassles. So in this quick blog, we have included all the possible methods to change file permissions in Linux.

How to Change File Permissions in Linux

In Linux, mainly Linux file permissions are divided into three parts, and these are:

- **Read (r):** In this category, users can only open and read the file and can't make any changes to it.
- **Write (w):** Users can edit, delete, and modify the file content with written permission.
- **Execute (x):** When the user has this permission, they can execute the executable script and access the file details.

Owner Representation	Modify permission	Permission symbols for symbolic mode	Permission symbols for absolute mode

	using the operator		
User → u	To add use '+'	Read → r	To add or subtract read use ± 4
Group → g	To subtract use '-'	Write → w	To add or subtract read use ± 2
Other → o	To set use '='	Execute → x	To add or subtract read use ± 1

As you can see from the above table, there are two types of symbol representation of permission. You can use both of these modes (symbolic and absolute) to change file permissions using the chmod command. The chmod refers to the change mode that allows users to modify the access permission of files or folders.

Using chmod Symbolic Mode

In this method, we use the symbol (for owner- u, g, o; for permission- r, w, x) to add, subtract, or set the permissions using the following syntax:

```
chmod <owner_symbol> mode <permission_symbol> <filename>
```

Before changing the file permission, first, we need to find the current one. For this, we use the 'ls' command.

```
ls -l
```

Here the permission symbols belong to the following owner:

- **'-' :** shows the file type.
- **'rw-' :** shows the permission of the user (read and write)
- **'rw-' :** shows the permission of the group(read and write)
- **'r- -' :** shows the permission of others (read)

In the above image, we highlighted one file in which the user has read and write permission, the group has read and write permission, and the other has only read permission. So here, we are going to add executable permission to others. For this, use the following command:

```
chmod o+x os.txt
```

As you can see, the execute permission has been added to the other category. Simultaneously, you can also change the multiple permissions of different owners. Following the above example, again, we change the permissions in it. So, here, we add executable permission from the user, remove write permission from the group, and add write permission to others. For this, we can run the below command:

```
chmod -v u+x ,g-w,o+w os.txt
```

Note: Use commas while separating owners, but do not leave space between them.

Using chmod Absolute Mode

Similarly, you can change the permission through absolute mode. In this method, mathematical operators (+, -, =) and numbers represent the permissions, as shown in the above table. For example, let's take an example and the updated permission of the file data is as follows:

Mathematical representation of the permission:

User	Read + Write	Permission is represented as **665**
	4+2=6	
Group	Read + Write	
	4+2=6	
Other	Read + Execute	
	4+1=5	

Now, we are going to remove read permission from the user and others, and the final calculation is:

User	Read + Write	-Read (-4)	Updated permission is represented as

	4+2=6	6-4=2	**261**
Group	Read + Write	–	
	4+2=6	6	
Other	Read + Execute	-Read (-4)	
	4+1=5	5-4=1	

To update the permission, use the following chmod command:

```
chmod -v 261 os.txt
```

Change User Ownership of the File

Apart from changing the file permission, you may also have a situation where you have to change the file ownership. For this, the chown is used which represents the change owner.

The file details represent the following details:

```
<filetype> <file_permission> <user_name> <group_name> <file_name>
```

So, in the above example, the owner's or user name is 'prateek', and you can change the user name that only exists on your system. Before changing the username, first list all the users using the following command:

```
cat /etc/passwd
```

Or

```
awk -F ':' '{print $1}' /etc/passwd
```

Now, you can change the username of your current or new file between these names. The general syntax to change the file owner is as follows:

```
sudo chown <new_username> <filename>
```

Note: Sudo permission is required in some cases.

Based on the above result, we want to change the username from ‘prateek’ to ‘proxy.’ To do this, we run the below command in the terminal:

```
sudo chown proxy os.txt
```

Change Group Ownership of the File

First, list all the groups that are present in your system using the following command:

```
cat /etc/group | cut -d: f1
```

The ‘chgrp’ command (change group) changes the filegroup. Here, we change the group name from ‘prateek’ to ‘disk’ using the following command:

```
sudo chgrp disk os.txt
```