# File permissions in Linux

## Project description

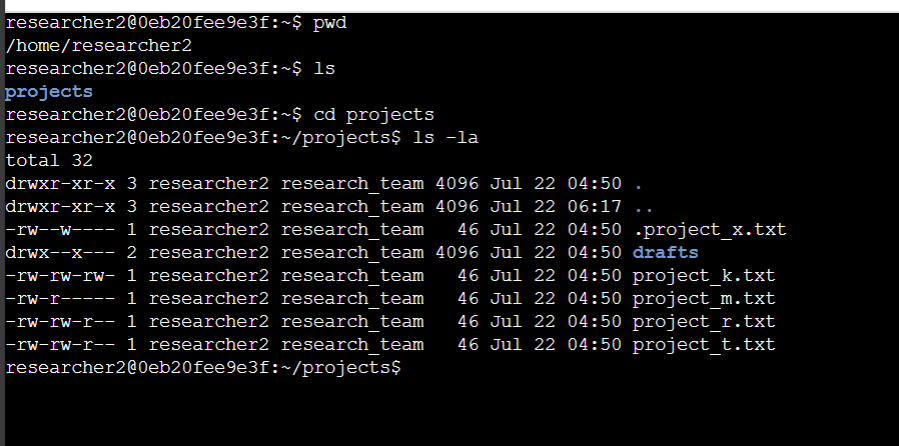
I’m going to access the current Access within the projects directory. The permissions do not currently reflect the level of authorization that should be given. Checking and updating these permissions will help keep their system secure. To complete this task, I performed the following tasks:

## Check file and directory details

In the /home/researcher2/projects directory, there are five files with the following

names and permissions:

Here is the screenshot for checking the permissions:



The permissions given for each files and subdirectories included in /home/researcher2/projects directory are:

● project\_k.txt

○ User = read, write,

○ Group = read, write

○ Other = read, write

● project\_m.txt

○ User = read, write

○ Group = read

○ Other = none

● project\_r.txt

○ User= read, write

○ Group = read, write

○ Other = read

● project\_t.txt

○ User = read, write

○ Group = read, write

○ Other = read

● .project\_x.txt

○ User = read, write

○ Group = write

○ Other = none

There is also one subdirectory inside the projects directory named drafts. The

permissions on drafts are:

● User = read, write, execute

● Group = execute

● Other = none

## Describe the permissions string

To work with granting or removing permissions , we use 10 words string which is ➖

***drwxrwxrwx***

* Here **d** represents the *directory* ; if its regular file then there will be hyphen(-) used.
* First **r** represents that the *user* has the Read permission , if the user did not have the read permission their will be - used.
* First **w** represents that the *user* has the write permission , if the user lacks this permission their will be - used.
* First **x** represents that the *user* has the execute permission , if the user lacks permission their will be - used.
* Second **r** represents that the *Group* has the Read permission , but if the group lacks the read permission their will be - used.
* Second **w** represents that the *group* has the Write permission , but if the Group lacks the write permission their will be - used
* Second **x** represents the *Group* has the execute permission , but if the Group did not have permission their will be - used
* Third **r** represents that the *Others* have the Read permission , if they did’t their will be - used
* Third **w** represents that the *others* have the Write permission , but if the other lacks this permission their will be - used.
* Third **x** represents that the *others* have the *execute* permission but if they lacks this permission their will be - used instead of x

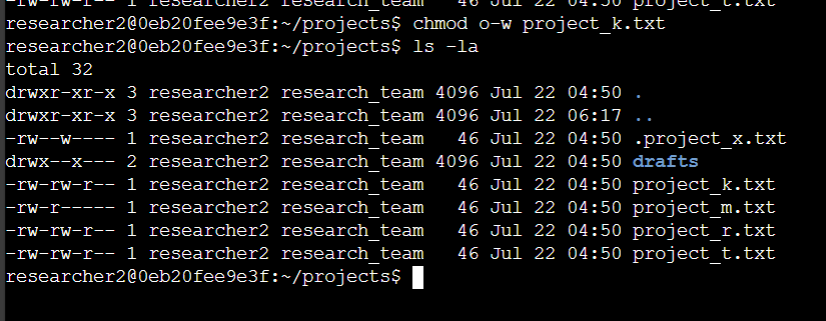
For example, the file permissions for project\_t.txt are -rw-rw-r--. Since the first character is a hyphen (-), this indicates that project\_t.txt is a file, not a directory. The second, fifth, and eighth characters are all r, which indicates that user, group, and other all have read permissions. The third and sixth characters are w, which indicates that only the user and group have write permissions. No one has execute permissions for project\_t.txt.

## Change file permissions

The command i use is :

**chmod o-w projects-k.txt**

Here is the screenshot of changing the file permission so that others will not have the write permission:



The first two lines of the screenshot display the commands I entered, and the other lines display the output of the second command. The chmod command changes the permissions on files and directories. The first argument indicates what permissions should be changed, and the second argument specifies the file or directory. In this example, I removed write permissions from other for the project\_k.txt file. After this, I used ls -la to review the updates I made

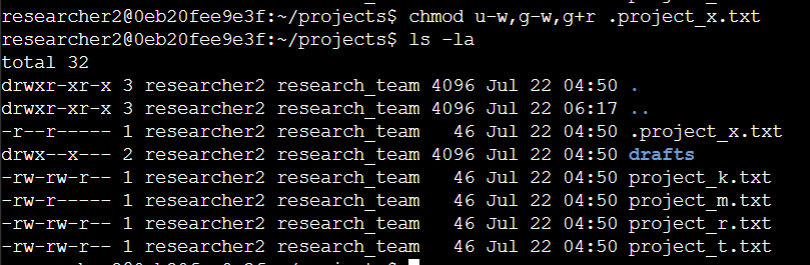
## Change file permissions on a hidden file

Using this command to change file permissions on a hidden file:

**chmod u-w,g-w,g+r**

So that the user and group doesn't have the write permission , they only read the file.

Here’s the screenshot:



The first two lines of the screenshot display the commands I entered, and the other lines display the output of the second command. I know .project\_x.txt is a hidden file because it starts with a period (.). In this example, I removed write permissions from the user and group, and added read permissions to the group. I removed write permissions from the user with u-w. Then, I removed write permissions from the group with g-w, and added read permissions to the group with g+r.

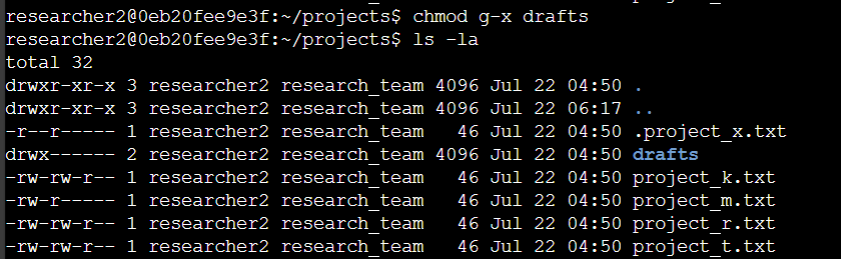
## Change directory permissions

Using this command to change the directory permissions:

**chmod g-w drafts**

So that only researcher2 who’s the user , only access the drafts file .

Here’s the screenshot:-



The first two lines of the screenshot display the commands I entered, and the other lines display the output of the second command. I previously determined that the group had execute permissions, so I used the chmod command to remove them. The researcher2 user already had execute permissions, so they did not need to be added.

## Summary

Checking the current permissions in the projects directory for this I use the first step in this was using ls -la to check the permissions for the directory. This informed my decisions in the following steps. I then used the chmod command multiple times to change the permissions on files and directories command we changes the permissions regarding their jobs and tasks