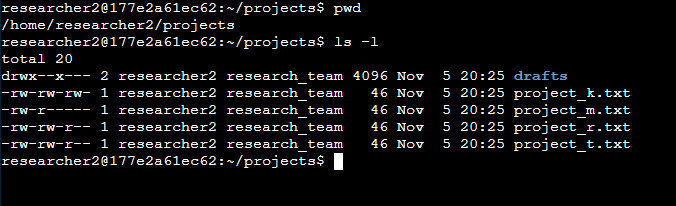
# File permissions in Linux

## Project description

This is a document to describe and show how one can go about viewing and modifying file and directory permissions through Linux CLI.

## Check file and directory details

In this example we are checking the file permissions for the files inside:  
**/home/researcher2/projects**

****

Here you can see that we use the **ls** command with the argument **-l**. **Ls** lists the contents of the directory or sub directories and the **-l** argument adds the permission lines to each item supplied by **ls**.

## Describe the permissions string

The string provided on the left side gives you several pieces of information including whether or not the item is a subdirectory or a file. The permissions for the **U**ser, **G**roup, and **O**thers.



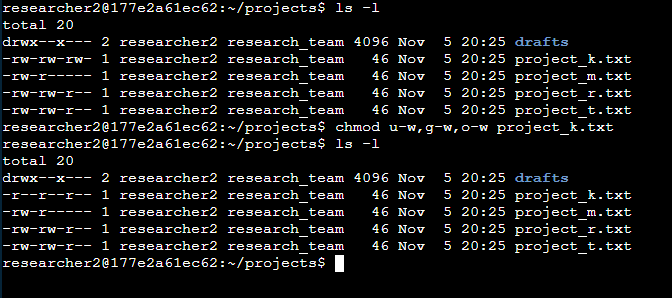
The first letter will either be a **d** or **-**, a **d** means it’s a subdirectory/directory and a **-** means that it is a file. The next set of three blocks of text are for the **U**ser, the next three after that refer to the **G**roup, and the last three are for **O**thers. **R** means that they have **R**ead permissions, **W** means they have **W**rite permissions, and **X** means they have permission to e**X**ecute; **-** means they are missing one of the said permissions based on where the dash is placed in the blocks.

For example “project\_k.txt” is a file since it has a dash in the first slot, the next three show the user has read and write permissions but not execute permissions.

The section where it says “research\_team” are the owners of the file or directory.

## Change file permissions

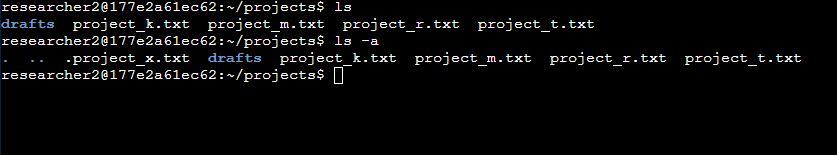
The command for changing file permissions is **chmod** and you use arguments to modify the permissions and dictate which file is being changed specifically. For example if we wanted to remove the Write permissions for everyone for the file “project\_k.txt” the command would be:  
**chmod u-w,g-w,o-w project\_k.txt**



The arguments are separated with commas and no spaces after and you use plus or minus to indicate if your adding (+) or removing (-) a permission and the letter specifies the group (u,g,o) then after that you list the file you are changing the permissions for. This command specifically removes the write permissions for each group and only leaves the read permission.

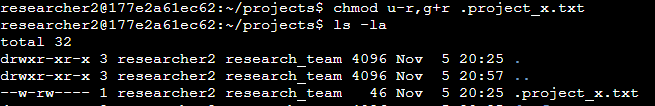
## Change file permissions on a hidden file

To find hidden files you must first show them and we use the same **ls** command but now with the argument **-a** to just list hidden files or **-la** to list permissions and hidden files.



As you can see there’s two new directories and a new file named “.project\_x.txt. Hidden files and directories start with a period.

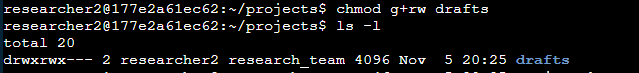


Let's say I wish the give the group permission to include read and remove the users permission to read the file we would use this command:  


Now the user can only write to the file and the group can now read the file.

## Change directory permissions

All the above information can also be used on directories also! For example we have:  
  
We want to give the group permission to not only execute the directory but also read and write to it. So we remember since we want to **ADD** permission we use the **+** and since it’s changing permission for the group we use **g** so we would use the command:  
**chmod g+rw drafts**

****

As you can see the group now has the included read and write permissions. Since it has come up in this example one issue you may run into is needing to add and remove permission in a single command. You can include multiple additional permissions in one block such as with g+rw but if we also wanted to remove the execution permission in the same command you must separate it with a comma.

chmod g+rw,g-x

This would give us the above additions to permissions while also removing one:  


## Summary

The ability to add and remove permissions is powerful and the commands are the same between files and directories and since this can prevent access or give access you must be very careful since you may remove access for someone who needs it to work and give access to someone who should have access!