# File permissions in Linux

## Project description

In this lab, I was tasked with reviewing and managing the permissions of the files and subdirectories /home/researcher2/projects directory for the user researcher2. Using various Linux commands in the command line interface I was able to examine the extant permissions of the files and subdirectories and modify them as needed in order to align with the authorizations they should have. Screenshots are provided when relevant to illustrate the commands and their outputs used to accomplish the necessary tasks.

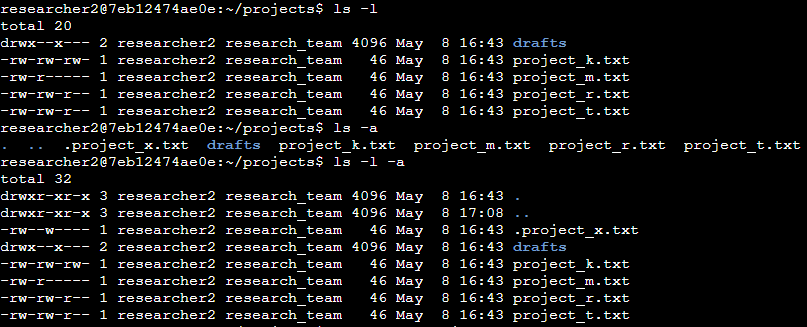
## Check file and directory details

First, in order to check the file and directory permissions in the projects directory, we must navigate to the correct directory. This is accomplished with the cd projects command, displayed here:



Next, we can check the permissions of all non-hidden files and directories with the ls -l command, display all files and directories, including hidden ones with the ls -a command, and finally display permissions of all files, hidden and non-hidden with the ls -l -a command.

The results of all three of these commands are shown below in order:



## Describe the permissions string

Using the permissions for the file project\_k.txt as our example, we can see what permissions are assigned on the file.



The string is composed of 10 characters, each representing a permission given to a different user or group of users, except for the first character, in this case a hyphen, which denotes whether or not the permissions relate to a directory (represented by a ‘d’) or a file.

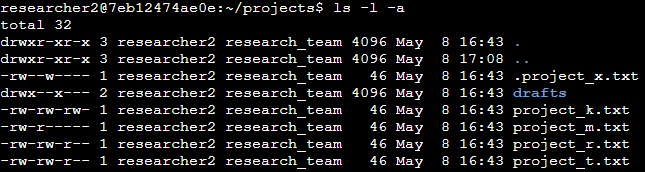
Characters two through four, rw-, denote the permissions for the individual user, in this case researcher2. The r displayed represents that the user has read permissions, the w represents write permissions, and the hyphen in the fourth space represents the user does not have execute permissions (which would otherwise be represented by an ‘x’).

Characters five through seven denote the permissions for the group, in this case research\_team. The group is given the same permissions as the user in this case, which are read and write permissions, but not execute permissions.

Finally, characters eight through ten denote the permissions for other users. In this example, other users possess read and write permissions, but lack execute permissions which is the same as both the user and group permissions.

## Change file permissions

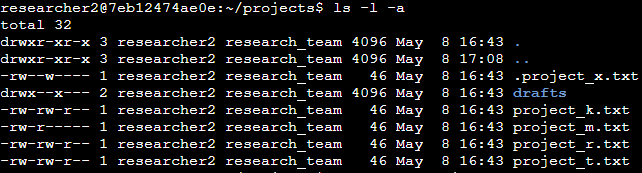
In this exercise, no files or directories should allow for other users to have write permissions. Referencing the permissions for all files in the projects directory, we can see that project\_k.txt have write permissions set for other users.



This can be changed by using the chmod command with the appropriate arguments. This is displayed in the following screenshot:



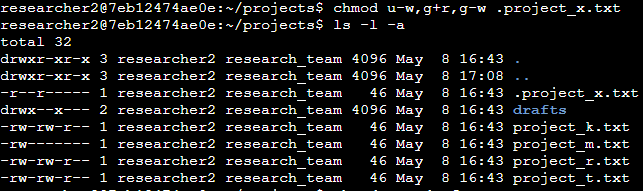
The argument following directly after chmod tell the command to remove write permissions from other users, represented by the ‘o’ in this case. The second, and final, argument tell the command which file or directory to modify these permissions on, which in this case is project\_k.txt. The result of this command can be seen in the following screenshot:



Thanks to the previous command, now no files in the projects directory allow other users to have write permissions.

## Change file permissions on a hidden file

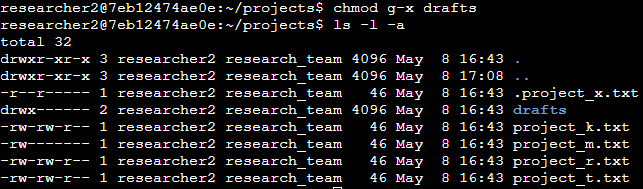
The next change that needs to be made is to the hidden file .project\_x.txt. This file should allow for both the user and group to read the file, but neither should be able to write to the file. Currently, both the user and group can write to the file, but only the user can read it. This can be corrected with the command chmod u-w,g+r,g-w .project\_x.txt. This command is followed by arguments to remove write permissions from the user and group, and assign read permissions to the group. The final argument specifies the file to be edited which in this case must be prefaced with a ‘.’ to signify that it is a hidden file. The command and its output are shown in the following screenshot:



The permissions string for .project\_x.txt now display that the user and group can read the file, but neither can write to it, which is in line with the needs of the organization.

## Change directory permissions

Next the permissions for the drafts subdirectory must be changed so that only the user researcher2 can execute it, rather than the entire group. This change can be accomplished with the command chmod g-x drafts. The arguments following this command tell the command to remove execute permissions from the group for the drafts subdirectory. The command and its output are displayed in the following screenshot:



The permissions string for the drafts subdirectory now display that the group no longer has execute permissions, which is appropriate for the needs of the user.

## Summary

Through this exercise I was able to examine and modify the permissions of users, groups, and other users for several files, both hidden and non-hidden, and subdiretories. I developed a deeper understanding of commands related to viewing permissions of content in directories along with a better knowledge of the permissions string used to represent this information. I also gained experience modifying the permissions of this content where needed.