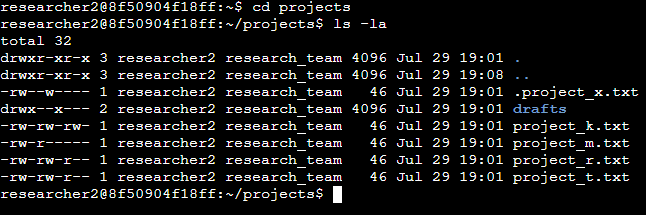
# File permissions in Linux

## Project description

I am a security professional at a large organization. I primarily work with the research team and my responsibilities include making sure that users on this team are authorized with the appropriate permissions in order to help keep the system secure. I am tasked with examining permissions on the file system to determine if the permissions match the authorization that should be given. If they don’t match, I will have to modify the permissions to authorize the appropriate users and remove any unauthorized access.

## Check file and directory details

This screenshot shows code that demonstrates how I used Linux commands to show existing permission sets for the project directory in this file system.



The second line shows the command I used, while the subsequent lines present the output. The code lists all items within the projects directory. Combining the "ls" command with the "-la" option, it reveals a comprehensive list of file contents, including hidden files. The outcome of this command reveals the existence of a directory labeled "drafts," a concealed file called ".project\_x.txt," and five additional project files. Notably, the 10-character string in the initial column represents the permissions assigned to each file or directory.

## Describe the permissions string

The 10-character string can be broken down to determine who is authorized to access the file and their specific permissions. The characters and what they represent are as follows:

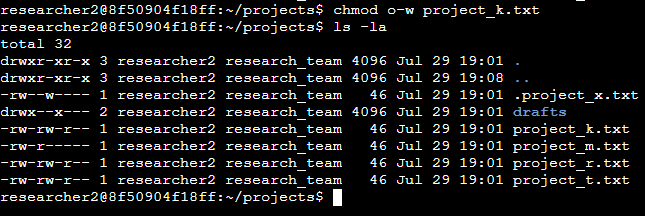
* 1st character: This character is either a “d” or hyphen “-” and indicates the file type. If it’s a “d”, it’s a directory. If it’s a hyphen “-”, it’s a regular file.
* 2nd-4th characters: These characters indicate the read “r”, write “w”, and execute “x” permissions for the user. When one of these characters is a hyphen “-” instead, it indicates that this permission is not granted to the user.
* 5th-7th characters: These characters indicate the read “r”, write “w”, and execute “x” permissions for the group. When one of these characters is a hyphen “-” instead, it indicates that this permission is not granted for the group.
* 8th-10th characters: These characters indicate the read “r”, write “w”, and execute “x” permissions for other. This owner type consists of all other users on the system apart from the user and the group. When one of these characters is a hyphen “-” instead, that indicates that this permission is not granted for other.

For example, the file permissions for “project\_k.txt” are “-rw-rw-rw-”. Since the first character is a hyphen (-), this indicates that “project\_k.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, sixth, and ninth characters are w, which indicates that the user, group, and other have write permissions. No one has execute permissions for “project\_k.txt”.

## Change file permissions

The organization determined that other shouldn't have write access to any of their files. To comply with this, I referred to the file permissions that I previously displayed. I determined “project\_k.txt” must have the write access removed for other.

The following code demonstrates how I used Linux commands to do this:

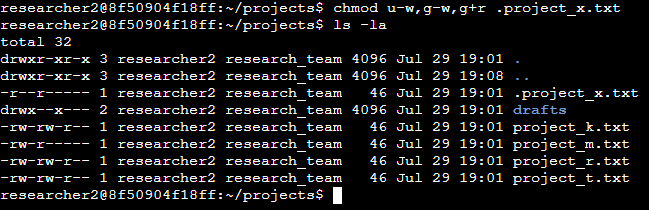


The first two lines in the screenshot exhibit the commands I entered, while the subsequent lines depict the output of the second command. By utilizing the "chmod" command, I altered the permissions of files and directories. The first argument specifies the permissions to be modified, and the second argument designates the file or directory affected. In this instance, I revoked write permissions from the "project\_k.txt" file for others. Following this adjustment, I employed "ls -la" to review the changes I had made.

## Change file permissions on a hidden file

The research team at my organization recently archived “project\_x.txt”. They do not want anyone to have write access to this project, but the user and group should have read access.

The following code demonstrates how I used Linux commands to change the permissions:

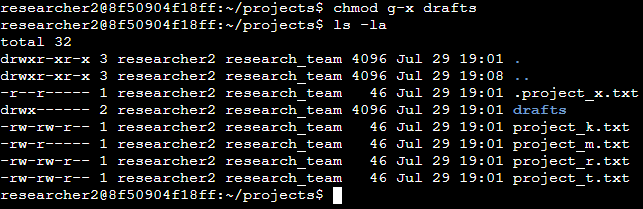


The first two lines show the commands I entered, while the subsequent lines present the output. ".project\_x.txt" is recognized as a hidden file because of its leading period ".". In this particular case, I adjusted the permissions by revoking write access from both the user and the group, and concurrently granting read permissions to the group. To achieve this, I first removed write permissions from the user using "u-w". Following that, I proceeded to remove write permissions from the group with "g-w" and then added read permissions to the group with "g+r".

## Change directory permissions

My organization only wants the “researcher2” user to have access to the “drafts” directory and its contents. This means that no one other than “researcher2” should have execute permissions.

The following code demonstrates how I used Linux commands to change the permissions:



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. I then used “ls -la” to check my work.

## Summary

I adjusted various permissions in the “projects” directory to align with my organization's authorization requirements. To begin, I utilized “ls -la” to examine the existing permissions for the directory, which influenced my subsequent actions. Utilizing the “chmod” command multiple times, I proceeded to modify permissions for files and directories accordingly.