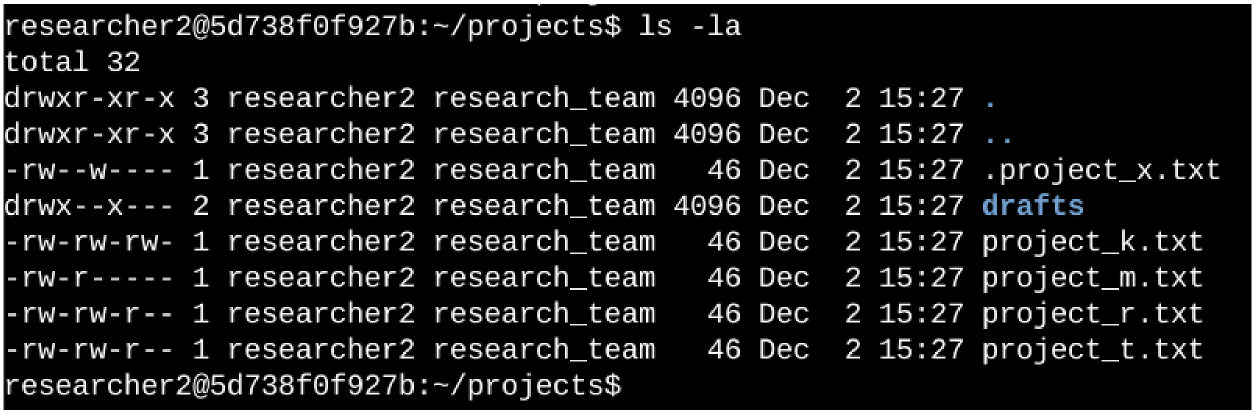
File permissions in Linux

Project description

The team needs to update the file permissions for certain files and folders in the”projects” director. The permissions don’t currently reflect the desired level of authorization. Checking and updating the file and directory permissions will help keep the system secure. To complete this task, I performed the following operations:

Check file and directory details

Using the Linux CLI, I first checked the existing permissions set for a specific directory.



```
researcher2@5d738f0f927b:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec  2 15:27 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec  2 15:27 ..
-rw--w---- 1 researcher2 research_team   46 Dec  2 15:27 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Dec  2 15:27 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Dec  2 15:27 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec  2 15:27 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_t.txt
researcher2@5d738f0f927b:~/projects$
```

The first line of the screenshot shows the command I entered, while the rest shows the output returned. Using the “ls” command with the ‘-la” option displays a detailed listing of the contents. The output indicates that there is one directory named “drafts”, one hidden file named “.project_a.txt” and five other project files.

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 listed here:

- **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.

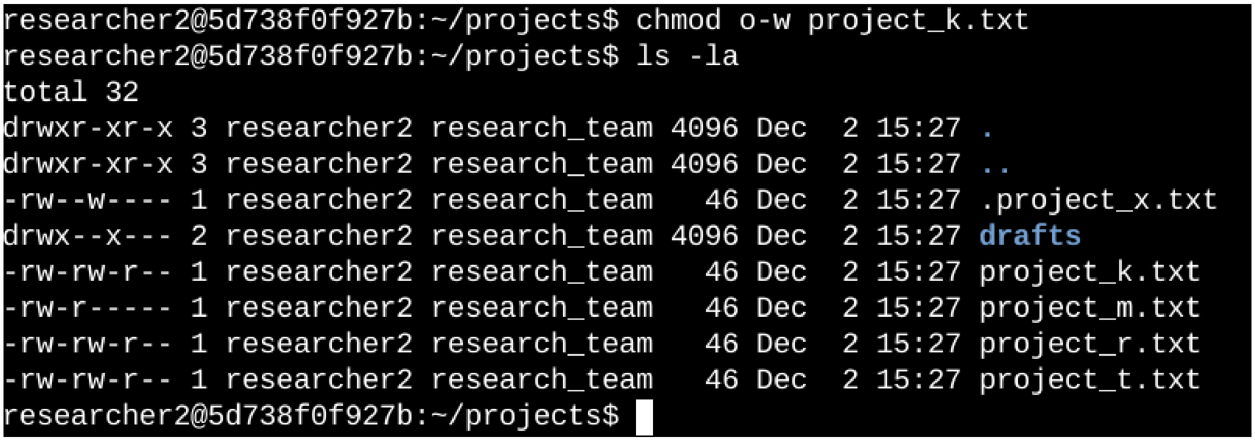
- **5th - 7th characters:** These characters indicate the read (`r`), write (`w`), and execute (`x`) permissions for the group. When one of these characters is a hyphen (`-`) instead, it indicates that this permission is not granted for the group.
- **8th - 10th characters:** These characters indicate 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_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 this file.

Change file permissions

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

The following command demonstrates how I used the CLI to do this:



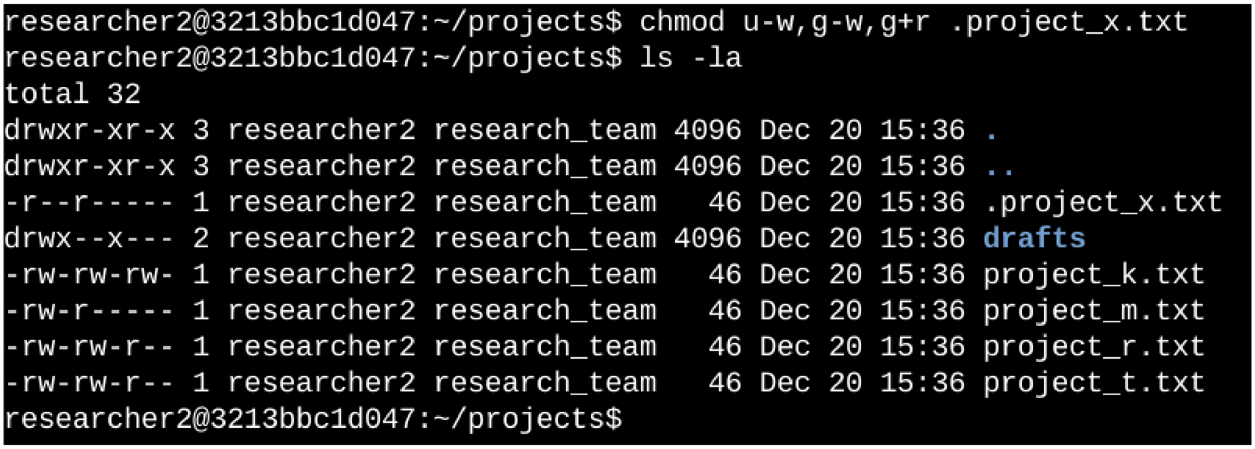
```
researcher2@5d738f0f927b:~/projects$ chmod o-w project_k.txt
researcher2@5d738f0f927b:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec  2 15:27 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec  2 15:27 ..
-rw--w---- 1 researcher2 research_team   46 Dec  2 15:27 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Dec  2 15:27 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec  2 15:27 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_t.txt
researcher2@5d738f0f927b:~/projects$
```

The first two lines of the screenshot display the commands I entered, and the other lines show 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

The research team 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 command shows how I used Linux CLI to change the permissions:



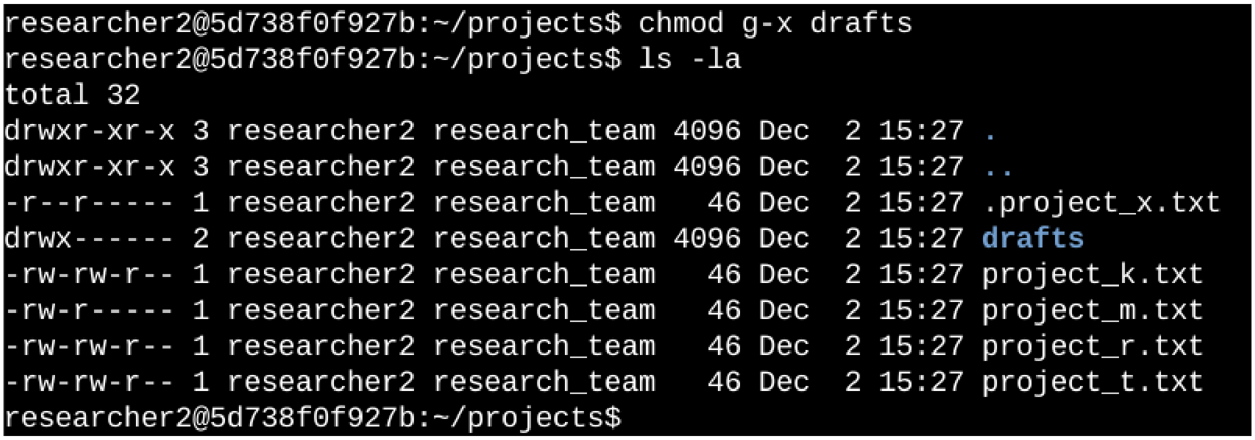
```
researcher2@3213bbc1d047:~/projects$ chmod u-w,g-w,g+r .project_x.txt
researcher2@3213bbc1d047:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec 20 15:36 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec 20 15:36 ..
-r--r----- 1 researcher2 research_team   46 Dec 20 15:36 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Dec 20 15:36 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Dec 20 15:36 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec 20 15:36 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec 20 15:36 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec 20 15:36 project_t.txt
researcher2@3213bbc1d047:~/projects$
```

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

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

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



```
researcher2@5d738f0f927b:~/projects$ chmod g-x drafts
researcher2@5d738f0f927b:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec  2 15:27 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec  2 15:27 ..
-r--r----- 1 researcher2 research_team   46 Dec  2 15:27 .project_x.txt
drwx------ 2 researcher2 research_team 4096 Dec  2 15:27 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec  2 15:27 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_t.txt
researcher2@5d738f0f927b:~/projects$
```

The first two lines of the screenshot display the commands I entered, and the other lines show 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

I changed multiple permissions to match the level of authorization my organization wanted for files and directories in the “projects” directory. The first step in this was using listing the files and permissions in the directory to see its current standing. This informed the following decisions in the remaining steps. I then used the “chmod” command multiple times to change the permissions on files and directories.