File permissions in Linux

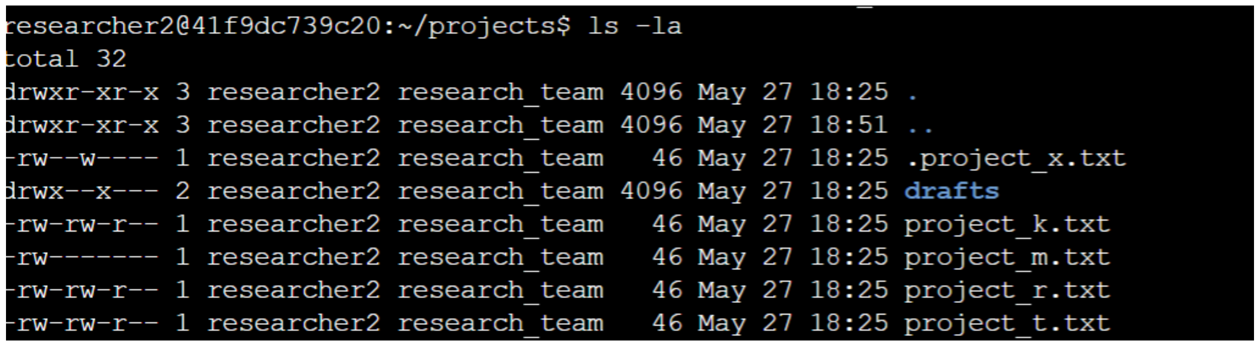
In this project I participated in a scenario managing linux file permissions for a fictional organization.

Project description

The research team at my organization needs to update the file permissions for certain files and directories 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, I performed the following tasks:

Check file and directory details

The screenshot below demonstrates how I used Linux commands to determine the existing permissions set for a specific directory in the file system.



```
esearcher2@41f9dc739c20:~/projects$ ls -la
otal 32
drwxr-xr-x 3 researcher2 research_team 4096 May 27 18:25 .
drwxr-xr-x 3 researcher2 research_team 4096 May 27 18:51 ..
-rw--w---- 1 researcher2 research_team   46 May 27 18:25 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 May 27 18:25 drafts
-rw-rw-r-- 1 researcher2 research_team   46 May 27 18:25 project_k.txt
-rw------- 1 researcher2 research_team   46 May 27 18:25 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 May 27 18:25 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 May 27 18:25 project_t.txt
```

The first line of the screenshot displays the command I entered, and the other lines display the output. The code lists all contents of the projects directory. I used the ls command with the -la option to display a detailed listing of the file contents that also returned hidden files. The output of my command indicates that there is one directory named drafts, one hidden file named .project_x.txt, and five other project files. The 10-character string in the first column represents the permissions set on each file or directory.

Description of permissions string

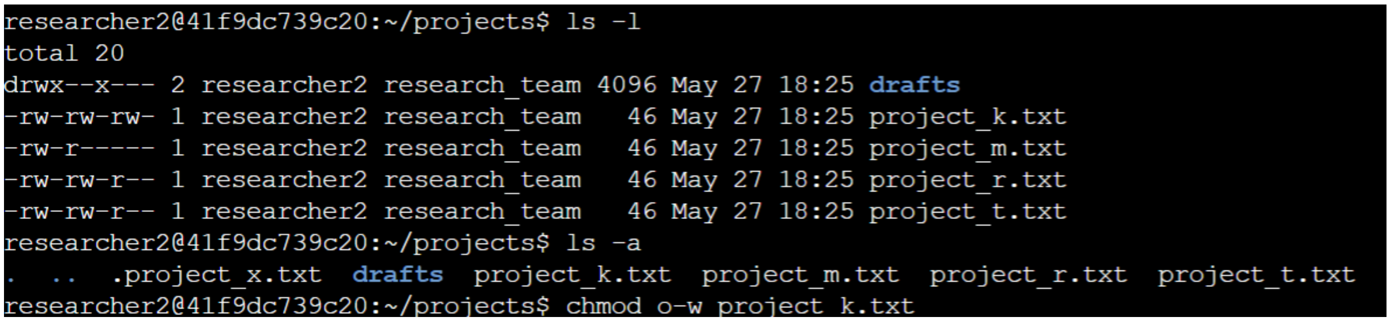
The 10-character string can be deconstructed to determine who is authorized to access the file and their specific permissions. The characters and what they represent are as follows:
● 1st character: This character is either a d or hyphen (-) and indicates the file type. If it's a d, it's a directory. 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.

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 returned. I determined project_k.txt must have the write access removed for other.

The screenshot below demonstrates how I used Linux commands to do this:



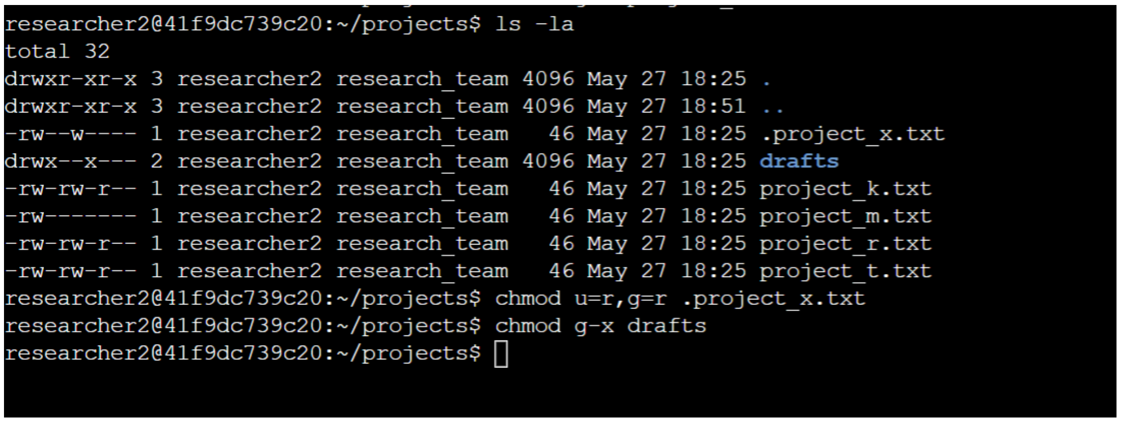
```
researcher2@41f9dc739c20:~/projects$ ls -l
total 20
drwx--x--- 2 researcher2 research_team 4096 May 27 18:25 drafts
-rw-rw-rw- 1 researcher2 research_team   46 May 27 18:25 project_k.txt
-rw-r----- 1 researcher2 research_team   46 May 27 18:25 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 May 27 18:25 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 May 27 18:25 project_t.txt
researcher2@41f9dc739c20:~/projects$ ls -a
.  ..  .project_x.txt  drafts  project_k.txt  project_m.txt  project_r.txt  project_t.txt
researcher2@41f9dc739c20:~/projects$ chmod o-w project_k.txt
```

The last line 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

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 screenshot below demonstrates how I used Linux commands to change the permissions:



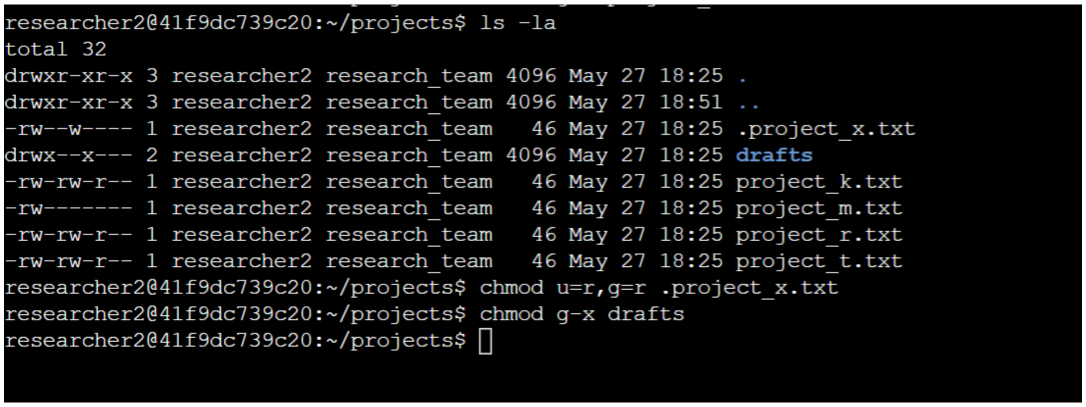
```
researcher2@41f9dc739c20:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 May 27 18:25 .
drwxr-xr-x 3 researcher2 research_team 4096 May 27 18:51 ..
-rw--w---- 1 researcher2 research_team   46 May 27 18:25 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 May 27 18:25 drafts
-rw-rw-r-- 1 researcher2 research_team   46 May 27 18:25 project_k.txt
-rw------- 1 researcher2 research_team   46 May 27 18:25 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 May 27 18:25 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 May 27 18:25 project_t.txt
researcher2@41f9dc739c20:~/projects$ chmod u=r,g=r .project_x.txt
researcher2@41f9dc739c20:~/projects$ chmod g-x drafts
researcher2@41f9dc739c20:~/projects$ 
```

The last 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=r. Then, I removed write permissions 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 researcher2 should have execute Permissions.

The screenshot below demonstrates how I used Linux commands to change the permissions:



```
researcher2@41f9dc739c20:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 May 27 18:25 .
drwxr-xr-x 3 researcher2 research_team 4096 May 27 18:51 ..
-rw--w---- 1 researcher2 research_team   46 May 27 18:25 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 May 27 18:25 drafts
-rw-rw-r-- 1 researcher2 research_team   46 May 27 18:25 project_k.txt
-rw------- 1 researcher2 research_team   46 May 27 18:25 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 May 27 18:25 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 May 27 18:25 project_t.txt
researcher2@41f9dc739c20:~/projects$ chmod u=r,g=r .project_x.txt
researcher2@41f9dc739c20:~/projects$ chmod g-x drafts
researcher2@41f9dc739c20:~/projects$ 
```

It was 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 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.