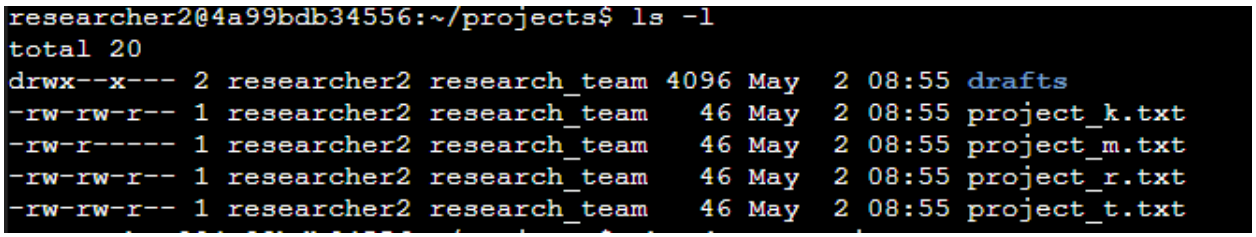
File permissions in Linux

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 task, I performed the following tasks:

Check file and directory details



To check for files and directory permission details under a particular directory we first need to navigate to the directory from the user directory. In this case we navigated to the /home/researcher2/project directory. Then, we check for files and directory permission details using the **ls -l** command as shown on the screenshot above. The **ls -l** command prints all details pertaining to the files and directory which includes:

- The permission details and type of each file or directory at the first column e.g drwx—x—
- The number of links of the file e.g 1
- The name of the user which is in this case researcher2
- The group the user belongs to which is research_team

The size of the files or directory in bytes e.g 46

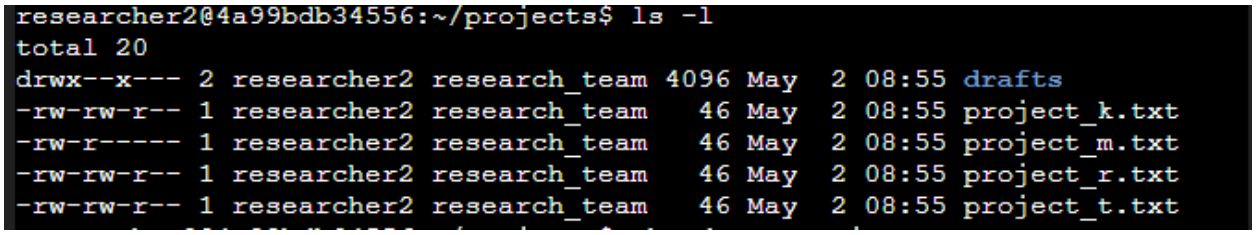
- Last modified including date/time
- The name of the file e.g project_k.txt

Describe the permissions string

The 10 string character is denoted by drwxrwxrwx when all file permissions are given
The 1st character indicates the file type. The d indicates it's a directory. When this character is a hyphen (-), it's a regular file.

- The 2nd-4th 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.
- The 5th-7th 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.
- The 8th-10th characters indicate the read (`r`), write (`w`), and execute (`x`) permissions for the owner type of 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 others.

Change file permissions



The prerequisite for the file project_m.txt authorization is to give only the user the permission to read and write to the file. Using the command **ls -l** to confirm that the proper permissions are in place, it was noticed that the project_m.txt file had permission for the group to read the file. So the **chmod g-r project_m.txt** command was used to remove the read permission from

the group as shown below:

```
researcher2@4a99bdb34556:~/projects$ ls -l
total 20
drwx--x--- 2 researcher2 research_team 4096 May  2 08:55 drafts
-rw-rw-r-- 1 researcher2 research_team   46 May  2 08:55 project_k.txt
-rw-r----- 1 researcher2 research_team   46 May  2 08:55 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 May  2 08:55 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 May  2 08:55 project_t.txt
researcher2@4a99bdb34556:~/projects$ chmod g-r project_m.txt
researcher2@4a99bdb34556:~/projects$ ls -l
total 20
drwx--x--- 2 researcher2 research_team 4096 May  2 08:55 drafts
-rw-rw-r-- 1 researcher2 research_team   46 May  2 08:55 project_k.txt
-rw------- 1 researcher2 research_team   46 May  2 08:55 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 May  2 08:55 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 May  2 08:55 project_t.txt
researcher2@4a99bdb34556:~/projects$
```

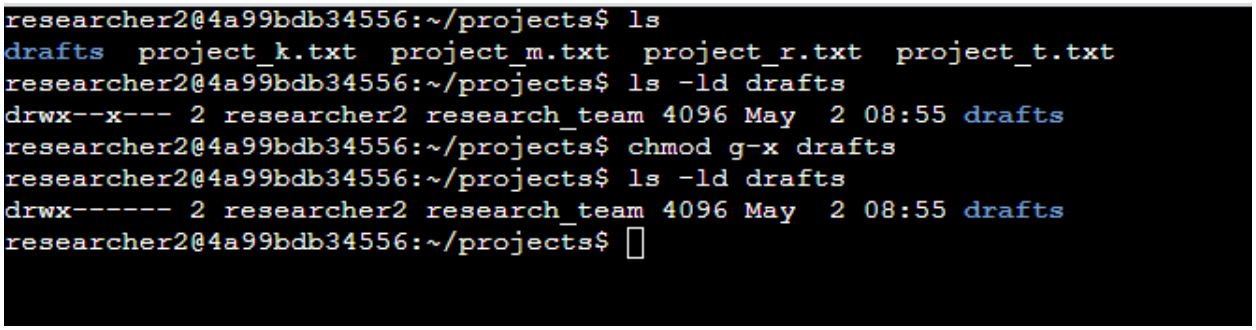
In the picture above, **chmod g-r project_m.txt** has been used to remove the read permission for the project_m.txt file.

Change file permissions on a hidden file

```
researcher2@4a99bdb34556:~/projects$ ls -a
.   .project_x.txt  project_k.txt  project_r.txt
..  drafts          project_m.txt  project_t.txt
researcher2@4a99bdb34556:~/projects$ ls -l .project_x.txt
-rw--w---- 1 researcher2 research_team 46 May  2 08:55 .project_x.txt
researcher2@4a99bdb34556:~/projects$ chmod u=r,g=r .project_x.txt
researcher2@4a99bdb34556:~/projects$ ls -l .project_x.txt
-r--r----- 1 researcher2 research_team 46 May  2 08:55 .project_x.txt
researcher2@4a99bdb34556:~/projects$
```

To find the hidden files we used the command **ls -a** to list all files and directories including the hidden ones. So after running the command, the .project_x.txt appeared which is a hidden file. It is a hidden file so the user and group should be given the permission to read the file only. The command **ls -l .project_x.txt** was used to check the file permission status as used above. After running the command, it was noticed that both the user and the group were having the permission to write to the .project_x.txt file. So the **chmod u=r,g=r .project_x.txt** was used to assign only the read permission to the user and group. The **ls -l .project_x.txt** was then used to confirm the permission status of the .project_x.txt.

Change directory permissions



The drafts directory is a directory that should only be accessed by the user. So using the **ls -ld drafts** command to confirm the permission status, it was noticed that there was an unauthorized execute permission given to the group. Using the **chmod g-x drafts** command, the execute permission for the group was removed.to confirm, the **ls -ld drafts** was used to check the permissions to see if all the permissions aligned with the authorization periqusite

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.