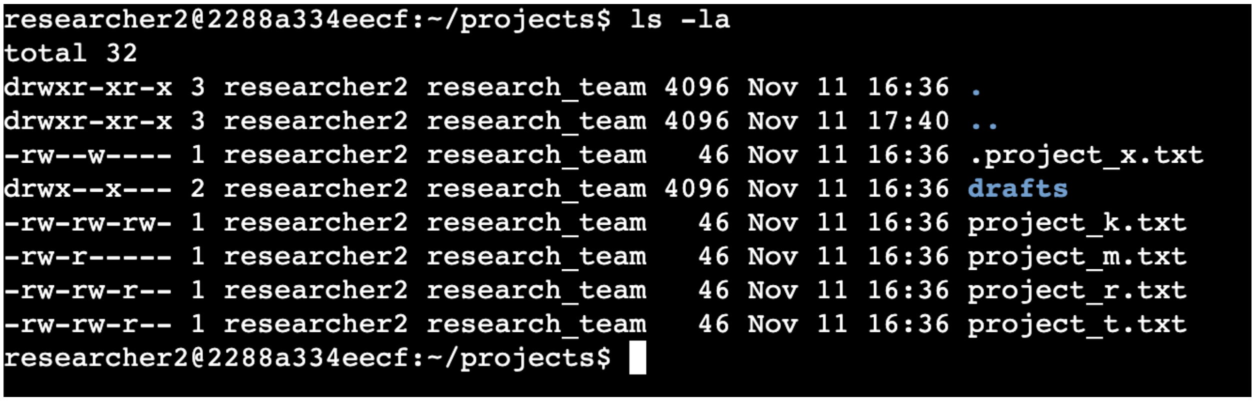
File permissions in Linux

Project description

The research group in our company is undergoing a security update for their data management practices. They've identified that the current access levels granted for various files and directories, specifically within the projects directory, are not aligned with the intended authorization hierarchy. To enhance the security of their systems, I have been assigned the responsibility to review and modify these file permissions accordingly. This step is essential to ensure that the access rights are correctly established, thereby maintaining the integrity and confidentiality of the data.

Check file and directory details

The code snippet provided exemplifies the process I used to inspect the current permissions assigned to a particular directory within the file system, utilizing a set of commands in the Linux environment. This was a necessary step in evaluating and ensuring that the directory's access levels are appropriate and secure.



```
researcher2@2288a334eecf:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Nov 11 16:36 .
drwxr-xr-x 3 researcher2 research_team 4096 Nov 11 17:40 ..
-rw--w---- 1 researcher2 research_team   46 Nov 11 16:36 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Nov 11 16:36 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Nov 11 16:36 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Nov 11 16:36 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Nov 11 16:36 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Nov 11 16:36 project_t.txt
researcher2@2288a334eecf:~/projects$ 
```

I used the **ls** command with the **-la** flags. This combination of flags is used to list all the files in the projects directory in long format, which includes hidden files (those beginning with a `.`). The output reveals a directory named drafts and a hidden file .project_x.txt, along with several other files related to various projects. The output also includes a 10-character string for each item, indicating the permission attributes for files and directories. This information is critical in understanding who has read, write, and execute permissions for these items.

Describe the permissions string

Here's a breakdown of the Linux file permission representation, as exemplified by the file project_t.txt:

- The 10-character string at the beginning of the listing provides essential security information about the file or directory.

- The first character indicates the type: `d` for a directory and `-` for a regular file.

- Characters 2-4 represent the owner's permissions, where `r` stands for read, `w` for write, and `x` for execute. A hyphen (`-`) means the permission is not granted.

- Characters 5-7 show the group's permissions, using the same `r`, `w`, and `x` notation.

- Finally, characters 8-10 list the permissions for others, meaning users who are neither the owner nor part of the group.

For the `project_t.txt` file:

- The `-` at the start confirms it's a regular file.

- `rw-` for the owner means they can read and write but not execute.

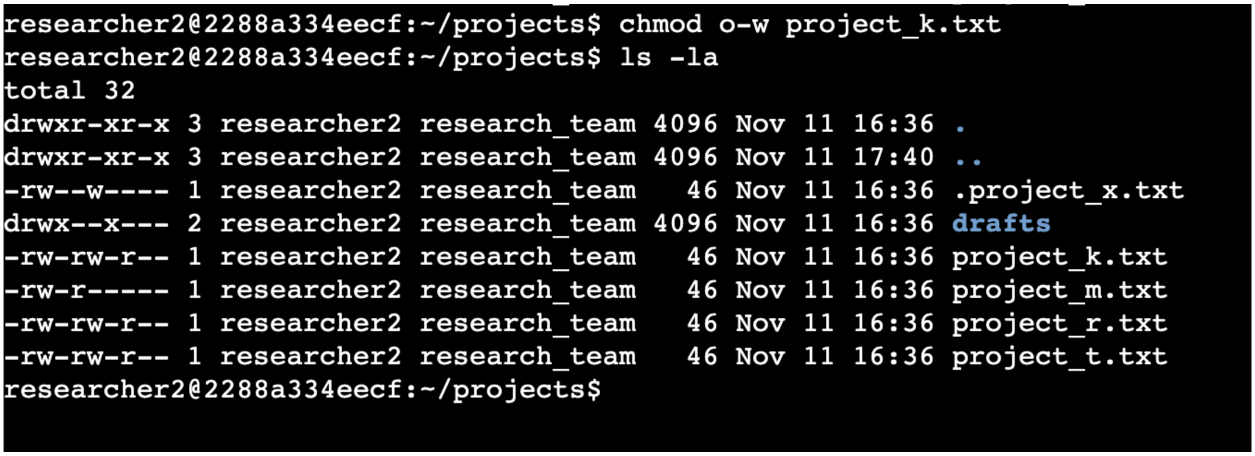
- `rw-` for the group indicates the same permissions as the owner.

– `r--` for others suggests they can only read the file, with no permissions to write or execute.

Change file permissions

The organization has decided to revise its file permissions policy to ensure that users who are not part of the owner group, referred to as "other", should not have write access to files. To implement this new security protocol, I analyzed the existing permissions for the files in question. Specifically for the file named `project_k.txt`, it was necessary to revoke write access for those who are classified as "other".

To execute this change, I utilized Linux commands to alter the file permissions accordingly. This involved modifying the permissions such that "other" would retain read access without the ability to modify the file, aligning with the organization's updated security measures.

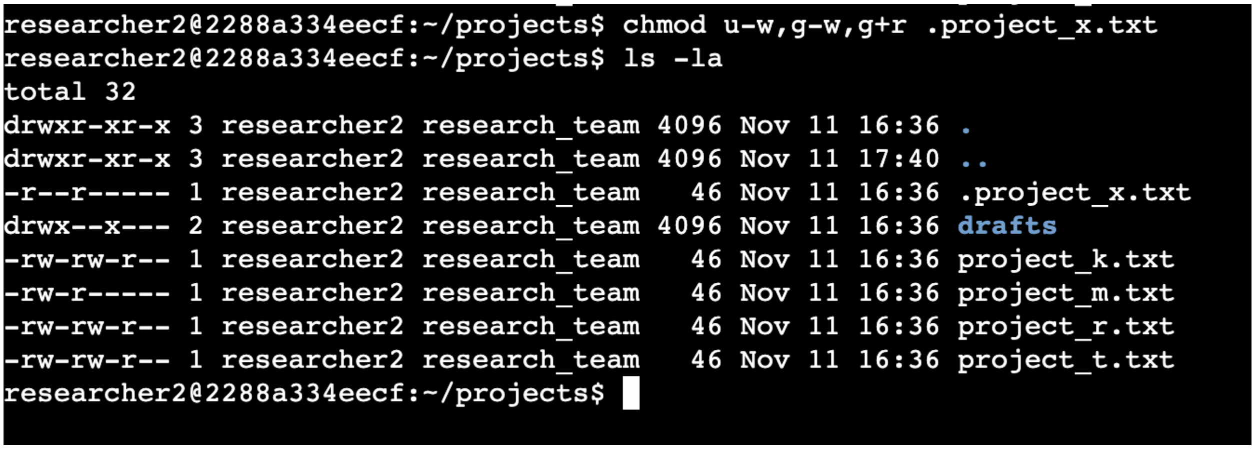


```
researcher2@2288a334eecf:~/projects$ chmod o-w project_k.txt
researcher2@2288a334eecf:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Nov 11 16:36 .
drwxr-xr-x 3 researcher2 research_team 4096 Nov 11 17:40 ..
-rw--w---- 1 researcher2 research_team   46 Nov 11 16:36 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Nov 11 16:36 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Nov 11 16:36 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Nov 11 16:36 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Nov 11 16:36 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Nov 11 16:36 project_t.txt
researcher2@2288a334eecf:~/projects$
```

In the command sequence shown, the chmod command was used to modify the permissions of the file `project_k.txt` to remove write access for "other". After altering the permissions, the ls −la command was executed to verify the changes. This process ensures that unauthorized users cannot modify the file, bolstering the system's security.

Change file permissions on a hidden file

To secure the `project_x.txt` file, I used Linux commands to remove write permissions while maintaining read access for the user and group.

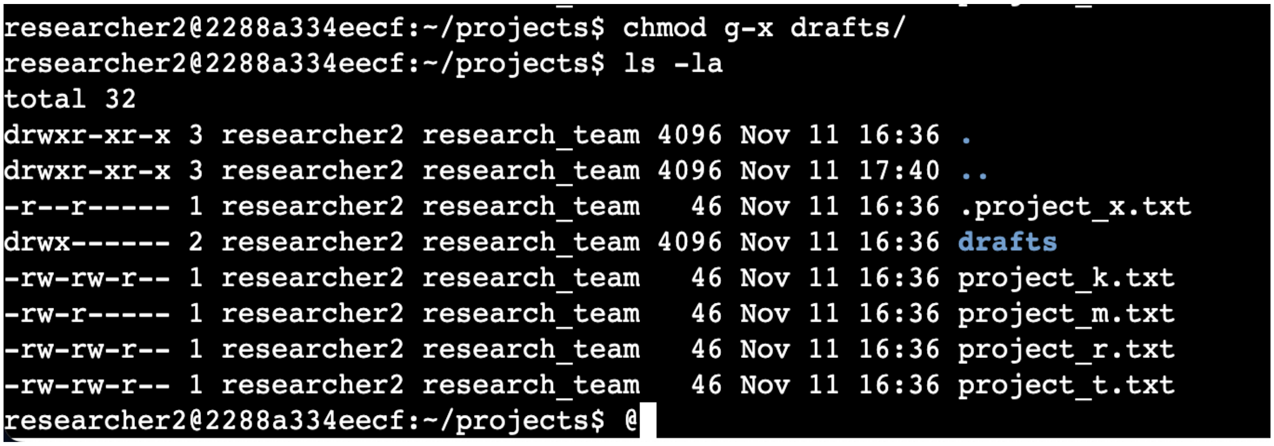


```
researcher2@2288a334eecf:~/projects$ chmod u-w,g-w,g+r .project_x.txt
researcher2@2288a334eecf:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Nov 11 16:36 .
drwxr-xr-x 3 researcher2 research_team 4096 Nov 11 17:40 ..
-r--r----- 1 researcher2 research_team   46 Nov 11 16:36 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Nov 11 16:36 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Nov 11 16:36 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Nov 11 16:36 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Nov 11 16:36 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Nov 11 16:36 project_t.txt
researcher2@2288a334eecf:~/projects$ 
```

Using chmod, I changed `.project_x.txt` permissions: `u−w` to remove user write, `g−w` to remove group write, and `g+r` to grant group read.

Change directory permissions

My organization required that only the user `researcher2` have execute permissions for the drafts directory. I utilized Linux commands to modify the access rights accordingly.



```
researcher2@2288a334eecf:~/projects$ chmod g-x drafts/
researcher2@2288a334eecf:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Nov 11 16:36 .
drwxr-xr-x 3 researcher2 research_team 4096 Nov 11 17:40 ..
-r--r----- 1 researcher2 research_team   46 Nov 11 16:36 .project_x.txt
drwx------ 2 researcher2 research_team 4096 Nov 11 16:36 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Nov 11 16:36 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Nov 11 16:36 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Nov 11 16:36 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Nov 11 16:36 project_t.txt
researcher2@2288a334eecf:~/projects$ @
```

My organization has directed that only the user `researcher2` should have access to the drafts directory. I modified the permissions using chmod, ensuring `researcher2` retained execute permissions while others were excluded.

Summary

I updated the permissions in the projects directory to meet our organization's authorization needs. Starting with `ls –la` to assess current permissions, I then executed `chmod` to apply the necessary changes to files and directories accordingly.