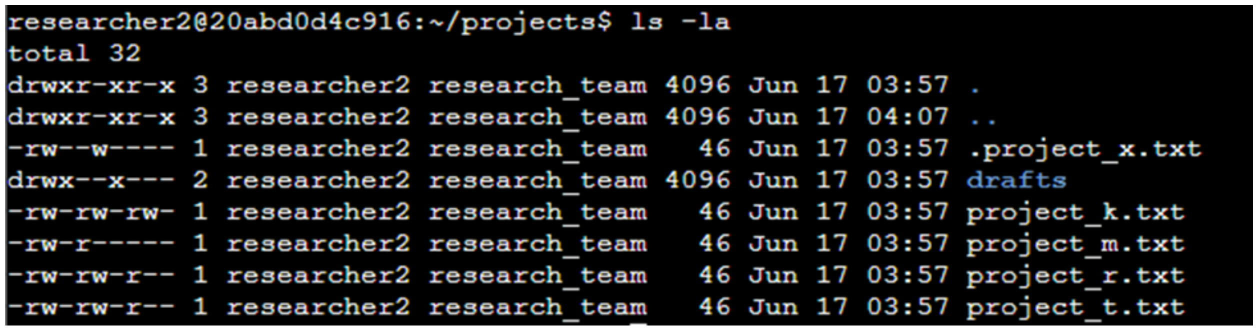
File permissions in Linux

Project description

I am a security professional at a large organization. I mainly work with their research team. Part of my job is to ensure users on this team are authorized with the appropriate permissions. This helps keep the system secure.
My task is to examine existing permissions on the file system. I'll need to determine if the permissions match the authorization that should be given. If they do not match, I'll need to modify the permissions to authorize the appropriate users and remove any unauthorized access.
I use common Bash commands such as ls, chmod, cd, etc. to modify all files and ensure the principle of least privilege by limiting each file's access to only the users who need to access them.

Check file and directory details

First, I used ls -la find all files, including hidden files, and view their current permissions by the Linux User, Group, and Other permission types.



```
researcher2@20abd0d4c916:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Jun 17 03:57 .
drwxr-xr-x 3 researcher2 research_team 4096 Jun 17 04:07 ..
-rw--w---- 1 researcher2 research_team   46 Jun 17 03:57 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Jun 17 03:57 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Jun 17 03:57 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Jun 17 03:57 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jun 17 03:57 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jun 17 03:57 project_t.txt
```

Describe the permissions string

The permission strings output by the ls -la command tell me if the file is a directory or a file and what permissions the User, Group, and Other permission types have. Each permission type has set permissions for read, write, and execute.

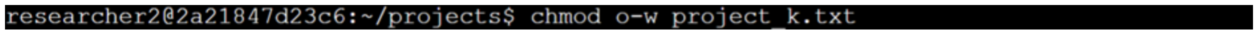
For example, the project_k.txt file has the permission string “-rw-rw-rw-“ meaning that it is a file (and not a directory as directory would be “drw-rw-rw”) and each permission group has “rw-“ permissions, which allows each group to read, write, but not execute the file.

Many of the listed files and directories currently have incorrect permissions. For example, files like .project_x.txt still allow Group users to write to the file but this should not be allowed as it is an archived file.

Change file permissions

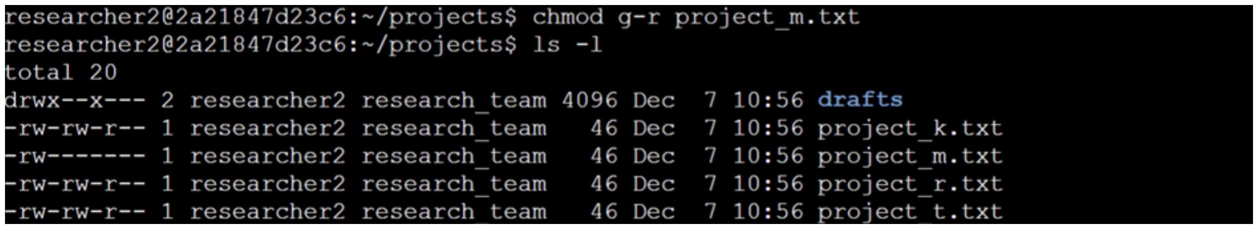
The organization does not allow the Other permission group to have write access to any files. As we can see from the permission strings, the file project_k.txt still has write access for the Other group.

I removed write access for the Other group on the file using the chmod bash command as shown below:



```
researcher2@2a21847d23c6:~/projects$ chmod o-w project_k.txt
```

Then I can confirm that project_k.txt now only allows users in the Other group to read the file.



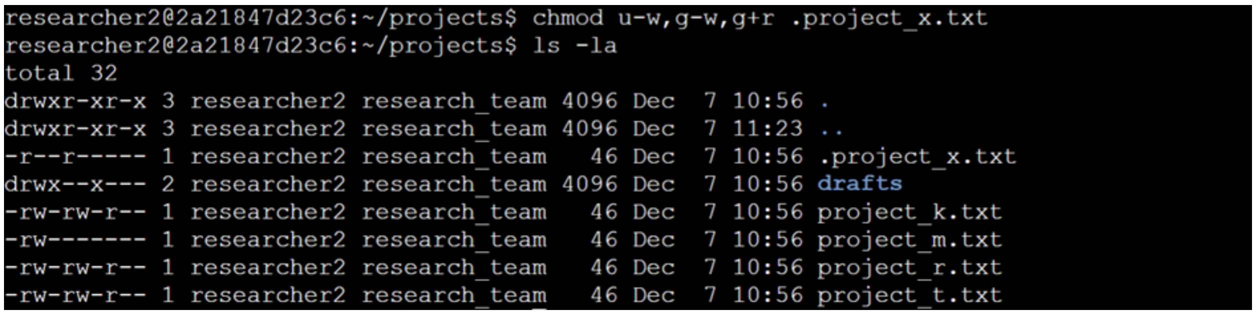
```
researcher2@2a21847d23c6:~/projects$ chmod g-r project_m.txt
researcher2@2a21847d23c6:~/projects$ ls -l
total 20
drwx--x--- 2 researcher2 research_team 4096 Dec  7 10:56 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Dec  7 10:56 project_k.txt
-rw------- 1 researcher2 research_team   46 Dec  7 10:56 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  7 10:56 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  7 10:56 project_t.txt
```

Change file permissions on a hidden file

As mentioned earlier, the .project_x.txt file is an archived file that should not have write permissions for anyone. However, it still should be able to be read by the User and Group permission groups.

I first modified the access permissions to the file by removing any current write permissions and ensuring that the User and Group permission groups can read the file.

Then I verified that the correct groups have read access.

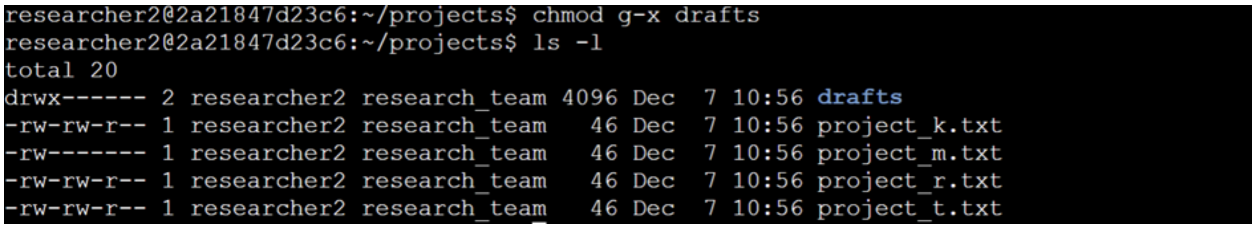


```
researcher2@2a21847d23c6:~/projects$ chmod u-w,g-w,g+r .project_x.txt
researcher2@2a21847d23c6:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec  7 10:56 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec  7 11:23 ..
-r--r----- 1 researcher2 research_team   46 Dec  7 10:56 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Dec  7 10:56 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Dec  7 10:56 project_k.txt
-rw------- 1 researcher2 research_team   46 Dec  7 10:56 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  7 10:56 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  7 10:56 project_t.txt
```

Change directory permissions

Finally, I modified the access permissions of any directories as they only belong to the User, no other permission groups should have permissions with the directories.

Using chmod, I updated the permissions of the draft directory to ensure that the current execute permissions for the Group permission group are removed.



```
researcher2@2a21847d23c6:~/projects$ chmod g-x drafts
researcher2@2a21847d23c6:~/projects$ ls -l
total 20
drwx------ 2 researcher2 research_team 4096 Dec  7 10:56 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Dec  7 10:56 project_k.txt
-rw------- 1 researcher2 research_team   46 Dec  7 10:56 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  7 10:56 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  7 10:56 project_t.txt
```

Summary

As a cybersecurity analyst, managing file and directory access permissions is crucial. In this project, I utilized Bash commands to ensure the organization’s filesystem adhered to proper access control standards. Many files initially had excessive permissions, violating the principle of least privilege—a significant security risk that could expose sensitive information to unauthorized individuals. Using Bash commands like **ls** and **chmod**, I reviewed and adjusted permissions by groups, ensuring each user was granted only the necessary access according to the organization’s authorization policies.