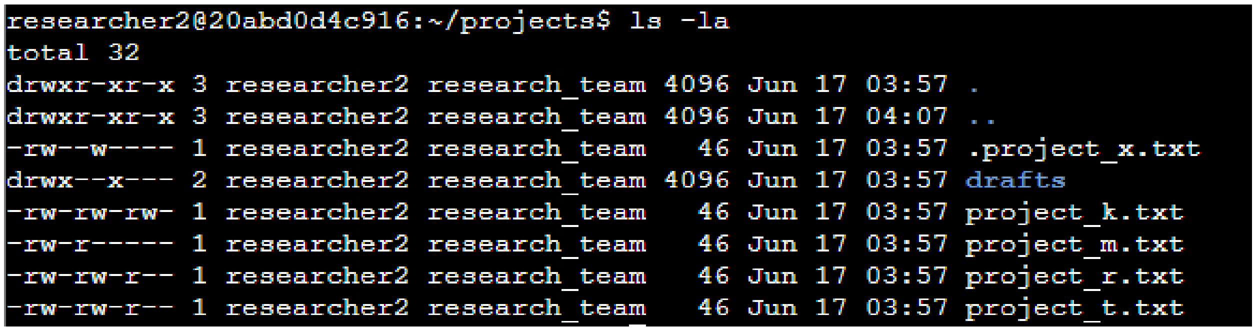
File permissions in Linux

Project description

The research team at my organization needs to update the file permissions for certain files and directories, and in this project I used Bash commands to secure the fiels based on authorization levels. 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.

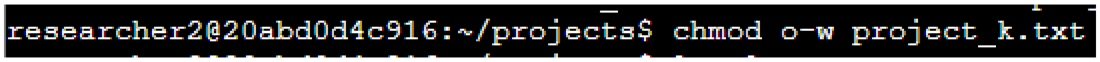
For example, the *project_k.txt* file has the permission string “-rw-rw-rw-“ meaning that it is a file (and not a directory) and each permission group has “rw-“ permissions, which allows each group to read, write, but not execute the file.

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

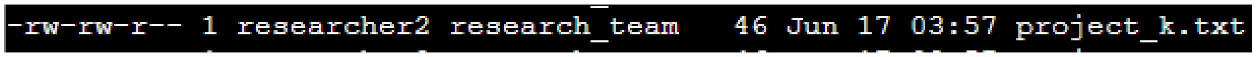
The organization does not allow the Other permission group to have write access to any files. As we can see from the permission strings, the file *project_k.txt* still has write access for the Other group.

I removed write access for the Other group on the file using the **chmod** bash command:



```
researcher2@20abd0d4c916:~/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.

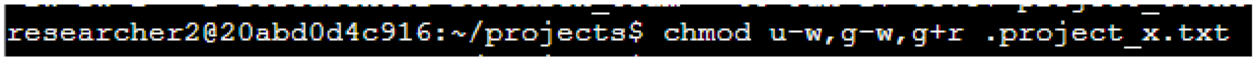


```
-rw-rw-r-- 1 researcher2 research_team   46 Jun 17 03:57 project_k.txt
```

Change file permissions on a hidden file

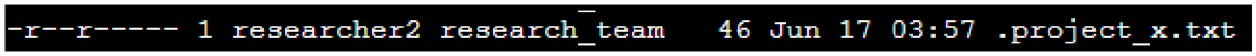
As mentioned earlier, the *.project_x.txt* file is an archived file that should not have write permissions for anyone. However, it still should be able to be read by the User and Group permission groups.

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.



```
researcher2@20abd0d4c916:~/projects$ chmod u-w,g-w,g+r .project_x.txt
```

Then I verified that the correct groups have read access.

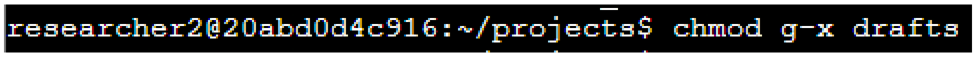


```
-r--r----- 1 researcher2 research_team   46 Jun 17 03:57 .project_x.txt
```

Change directory permissions

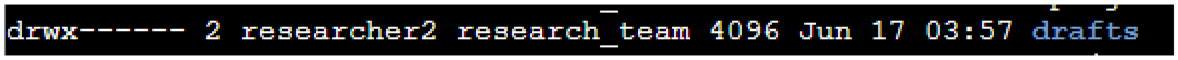
Finally, I modified the access permissions of any directories as they only belong to the User, no other permission groups should have permissions with the directories.

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@20abd0d4c916:~/projects$ chmod g-x drafts
```

Now the directory only allows the User to have any access permissions.



```
drwx------ 2 researcher2 research_team 4096 Jun 17 03:57 drafts
```

Summary

File and directory access permissions are very important to manage as a cybersecurity analyst and in this project, I used Bash commands to ensure the organization's filesystem has the correct access permissions. Many files had more permissions than they needed, which violates the principle of least privilege, and this can be a serious security concern as sensitive information could potentially be leaked to unwanted parties. Bash commands such as ls and chmod helped me view and modify access permissions by permission groups so that each user has the correct permissions based on the organization’s authorization standards.