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

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

```
researcher2@e31363b6e8c3:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec  1 15:22 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec  1 16:07 ..
-rw--w---- 1 researcher2 research_team   46 Dec  1 15:22 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Dec  1 15:22 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Dec  1 15:22 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec  1 15:22 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  1 15:22 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  1 15:22 project_t.txt
researcher2@23df7791779c:~/projects$
```

The first screenshot displays the command I entered and the second one displays the output. The command is used to display all the contents of the `projects` directory. I used `ls` command combined with `la` command to show the detailed listing of the files that also returned the hidden files. The directory consists of a hidden file `.project_x.txt` , a directory named drafts and other five project files. The 10 characters at the beginning tells us about the types of permission a file or the directory has.

Describe the permissions string

a 10-character string begins each entry and indicates how the permissions on the file are set. For instance, a directory with full permissions for all owner types would be `drwxrwxrwx:`

- **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 other.

Change file permissions

The organization does not allow other to have write access to any files. Based on the permissions of the files we can see that project_k.txt has the write access for the “other” so it must have its write access removed.

```
researcher2@23df7791779c:~/projects$ chmod o-w project_k.txt
```

```
researcher2@23df7791779c:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec  1 15:22 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec  1 16:07 ..
-rw--w---- 1 researcher2 research_team   46 Dec  1 15:22 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Dec  1 15:22 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Dec  1 15:22 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec  1 15:22 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  1 15:22 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  1 15:22 project_t.txt
```

The first screenshot displays the command I entered and the second one displays the output. The `chmod` command is used to modify the permissions of files and directories in Unix-like operating systems. In the provided command `chmod o-w project_k.txt`, the first argument 'o-w' specifies the modification to be made. Here, 'o' refers to 'others,' and '-w' indicates the removal of the write permission.

The second argument, 'project_k.txt,' specifies the file for which the permissions are to be changed. Therefore, the command is instructing the system to revoke the write permission for others on the '`project_k.txt`' file. This means that users who are not the owner of the file and not in the file's group will no longer have the ability to write to or modify the 'project_k.txt' file.

In summary, the `chmod o-w` command is modifying the permissions of the 'project_k.txt' file, specifically removing the write permission for others, thereby restricting write access to the file for users outside the file's owner and group."

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 following code demonstrates how I used Linux commands to change the permissions:

```
researcher2@23df7791779c:~/projects$ chmod u-w,g-w,g+r .project_x.txt
researcher2@23df7791779c:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec  1 15:22 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec  1 16:07 ..
-r--r----- 1 researcher2 research_team   46 Dec  1 15:22 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Dec  1 15:22 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Dec  1 15:22 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec  1 15:22 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  1 15:22 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  1 15:22 project_t.txt
```

The first screenshot displays the command I entered and the second one displays the output. I know `.project_x.txt` is a hidden file because it starts with a period (.). So by using this command I removed “write”(w) permission for both the group (`g-w`) and the user(`u-w`). Then I have granted read permission to the group(`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.

```
researcher2@23df7791779c:~/projects$ chmod g-x drafts
researcher2@23df7791779c:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec  1 15:22 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec  1 16:07 ..
-r--r----- 1 researcher2 research_team   46 Dec  1 15:22 .project_x.txt
drwx------ 2 researcher2 research_team 4096 Dec  1 15:22 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Dec  1 15:22 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec  1 15:22 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  1 15:22 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  1 15:22 project_t.txt
```

The first screenshot displays the command I entered and the second one displays the output. I previously determined that the drafts have the executable(x) permission for the group, so I used the `chmod` command to remove it. Now, only the user i.e `researcher2` has access to the drafts directory.

Summary

I modified the access levels for files and directories within the "projects" directory to align with the desired authorization levels set by my organization. Initially, I employed the `ls -la` command to inspect the existing permissions, guiding my subsequent actions. Subsequently, I utilized the `chmod` command multiple times to adjust permissions for both files and directories according to the specified criteria