File permissions in Linux

Project description

The research team at the organization needed to update permissions for certain files and directories within the project's directory. The permissions need to be edited to keep the system secure.

Check file and directory details

The following showcases the commands I initially used to determine the existing permissions:

```
researcher2@d4d15738b4e6:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Jan 19 18:22 .
drwxr-xr-x 3 researcher2 research_team 4096 Jan 19 18:47 ..
-rw--w---- 1 researcher2 research_team   46 Jan 19 18:22 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Jan 19 18:22 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Jan 19 18:22 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Jan 19 18:22 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jan 19 18:22 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jan 19 18:22 project_t.txt
```

The first line of the screenshot is the *ls—la* line, which shows a detailed listing of all the content files. The output showcases all the files and directories in the projects folder, including a hidden file named ".project_x.txt". The first 10-character string showcases the permissions set for each file/directory.

Describe the permissions string

The 10-character string can be deconstructed to show the specific permissions for each group.

- 1st character
  - either a d or a hyphen to indicate the file type. d for directory and hyphen for regular file
- 2nd - 4th characters
  - indicate the user's permission to read, write, and execute. if any of the characters are a hyphen instead, it shows that this permission is not granted to the user
- 5th - 7th characters
  - indicate the group's permission to read, write, and execute. if any of the characters are a hyphen instead, it shows that this permission is not granted to the group

- 8th - 10th characters
    - indicate the other's permission to read, write, and execute. if any of the characters are a hyphen instead, it shows that this permission is not granted to the other

Taking a specific example we can see *project_t.txt* permissions are -rw-rw-r--. The first hyphen shows it is a file, the user permissions are read and write and no execute, the group permissions are read and write and no execute, and others permission is only read.

Change file permissions

The organization wanted to change permissions so that other should not have any write access. I then went ahead to remove the write permission for *project_k.txt*.

```
researcher2@d4d15738b4e6:~/projects$ chmod o-w project_k.txt
researcher2@d4d15738b4e6:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Jan 19 18:22 .
drwxr-xr-x 3 researcher2 research_team 4096 Jan 19 18:47 ..
-rw--w---- 1 researcher2 research_team   46 Jan 19 18:22 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Jan 19 18:22 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Jan 19 18:22 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Jan 19 18:22 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jan 19 18:22 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jan 19 18:22 project_t.txt
researcher2@d4d15738b4e6:~/projects$
```

The first line shows the *chmod* command which I used to change mode / permissions for that specific file. I determined that other (o) should not have write access anymore, so I specified that and then used *ls -la* to look at the newest changes.

Change file permissions on a hidden file

The research time has recently archived the file *project_x.txt*. They determined they did not want anyone to have write access to the project, but the user and group should have read access.

```
researcher2@d4d15738b4e6:~/projects$ chmod u-w,g-w,g+r .project_x.txt
researcher2@d4d15738b4e6:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Jan 19 18:22 .
drwxr-xr-x 3 researcher2 research_team 4096 Jan 19 18:47 ..
-r--r----- 1 researcher2 research_team   46 Jan 19 18:22 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Jan 19 18:22 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Jan 19 18:22 project_k.txt
-rw------- 1 researcher2 research_team   46 Jan 19 18:22 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jan 19 18:22 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jan 19 18:22 project_t.txt
researcher2@d4d15738b4e6:~/projects$
```

The first line shows the removal of the write access from the user, write access from the group, and adding on the read access for the group for the archived file *.project_x.txt*. These commands are all separated by commas. The *ls -la* command is then used to display the changes in the permissions.

Change directory permissions

The organization only wants researcher2 to have access to the drafts directory and contents. There should be no permissions for anyone else other than researcher2 (the user).

```
researcher2@d4d15738b4e6:~/projects$ chmod g-x drafts
researcher2@d4d15738b4e6:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Jan 19 18:22 .
drwxr-xr-x 3 researcher2 research_team 4096 Jan 19 18:47 ..
-r--r----- 1 researcher2 research_team   46 Jan 19 18:22 .project_x.txt
drwx------ 2 researcher2 research_team 4096 Jan 19 18:22 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Jan 19 18:22 project_k.txt
-rw------- 1 researcher2 research_team   46 Jan 19 18:22 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jan 19 18:22 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jan 19 18:22 project_t.txt
```

The output shows a change made with the *chmod* command to remove execute permission for the group. This leaves the permission (drwx------) for the drafts directory, which shows that only researcher2 (the user) has permission to read, write, and execute the drafts directory.

Summary

I was tasked with changing the permissions of various files and directories per the changes. I have showcased the use of commands such as *chmod* and *ls* to conduct these changes for the permissions on files and directories.