File permissions in Linux

Project description

I was tasked with verifying permissions on various files within the projects directory to assure there are no authorization issues within the organization. If any files do not match the correct level of authorization, I will update them accordingly in order to help keep the system secure.

Check file and directory details

The first step was to navigate to the projects directory and determine the current permissions on the files within, using Linux.

```
researcher2@cf63117e9792:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 May 14 23:35 .
drwxr-xr-x 3 researcher2 research_team 4096 May 15 00:02 ..
-rw--w---- 1 researcher2 research_team   46 May 14 23:35 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 May 14 23:35 drafts
-rw-rw-rw- 1 researcher2 research_team   46 May 14 23:35 project_k.txt
-rw-r----- 1 researcher2 research_team   46 May 14 23:35 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 May 14 23:35 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 May 14 23:35 project_t.txt
researcher2@cf63117e9792:~/projects$
```

The command '**ls -la**' on the first line was used in order to list the contents of the projects directory, including any hidden files. The information under it displays the file names along with their permissions at the very beginning of each line.

Describe the permissions string

File permissions are displayed as a string of letters. The first letter that may or may not appear will be a '**d**' if we're looking at another directory, or a '**-**' if we're looking at a single file.

The 2nd, 3rd, and 4th letters may show a '**r**', '**w**', or '**x**' respectively. These are the '**read**', '**write**', and '**execute**' permissions for the User. Any letter replaced with a **hyphen** (**-**) indicates the user does not have the corresponding permissions.

The 4th - 7th characters will show the exact same letters and/or symbols as the User permissions on the 2nd - 4th characters, however these will be used for the Group category.

And lastly the 8th - 10th characters will again work the same way as the 2nd - 7th, however this will be for the Other category. This category consists of every other user on the system that is not from the user or group.

Change file permissions

The organization does not want the **other** category to have write access to any files. As demonstrated in the code below, I used Linux commands in order to change the permissions and remove write access.

```
researcher2@cf63117e9792:~/projects$ chmod o-w project_k.txt
researcher2@cf63117e9792:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 May 14 23:35 .
drwxr-xr-x 3 researcher2 research_team 4096 May 15 00:02 ..
-rw--w---- 1 researcher2 research_team   46 May 14 23:35 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 May 14 23:35 drafts
-rw-rw-r-- 1 researcher2 research_team   46 May 14 23:35 project_k.txt
-rw-r----- 1 researcher2 research_team   46 May 14 23:35 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 May 14 23:35 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 May 14 23:35 project_t.txt
researcher2@cf63117e9792:~/projects$
```

Change file permissions on a hidden file

The first line in the code below changes the **User** and **Group** permissions on the recently archived hidden file **.project_x.txt**. u=r and g=r sets the permissions for the user and group to read only respectively. Now as requested by the organization, no groups will be able to **write** or **execute** the hidden file, and the **Other** group will not have **write** access to any files.

```
researcher2@cf63117e9792:~/projects$ chmod u=r,g=r .project_x.txt
researcher2@cf63117e9792:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 May 14 23:35 .
drwxr-xr-x 3 researcher2 research_team 4096 May 15 00:02 ..
-r--r----- 1 researcher2 research_team   46 May 14 23:35 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 May 14 23:35 drafts
-rw-rw-r-- 1 researcher2 research_team   46 May 14 23:35 project_k.txt
-rw-r----- 1 researcher2 research_team   46 May 14 23:35 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 May 14 23:35 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 May 14 23:35 project_t.txt
researcher2@cf63117e9792:~/projects$
```

Change directory permissions

The **drafts** directory belongs to **researcher2** and should only be accessible to them. Previously the **group** category had **execute** permissions, however the code below demonstrates the **Linux** commands used in order to fix the issue. On line 1 you see **chmod g-x drafts/**, this is removing **execute** (**x**) permissions from the **group** (**g**) category.

```
researcher2@cf63117e9792:~/projects$ chmod g-x drafts/
researcher2@cf63117e9792:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 May 14 23:35 .
drwxr-xr-x 3 researcher2 research_team 4096 May 15 00:02 ..
-r--r----- 1 researcher2 research_team   46 May 14 23:35 .project_x.txt
drwx------ 2 researcher2 research_team 4096 May 14 23:35 drafts
-rw-rw-r-- 1 researcher2 research_team   46 May 14 23:35 project_k.txt
-rw-r----- 1 researcher2 research_team   46 May 14 23:35 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 May 14 23:35 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 May 14 23:35 project_t.txt
researcher2@cf63117e9792:~/projects$
```

Summary

As requested by the organization, I changed permissions for various groups on multiple files and / or directories inside the **projects** directory. Starting with the first line of code **ls -la** in order to verify the permissions at the beginning, I was able to make informed decisions on how to proceed and used **chmod** to change the requested permissions.