File permissions in Linux

Project description

I have been tasked to examine existing permissions on the file system within my organization. Currently my research team needs to update file permissions for certain files and directories within the `projects` directory. After examining the current authorization levels, the permissions do not adhere to company standards and I have modified the permissions to authorize the appropriate users and removed any unauthorized access. I have performed these tasks by:

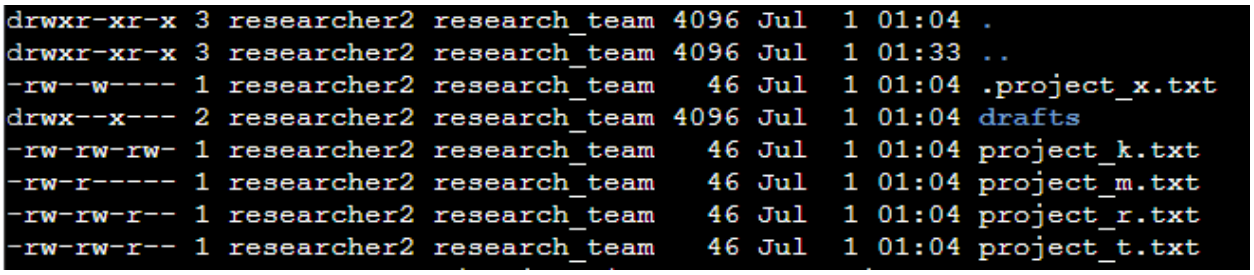
Check file and directory details

To check the file and directory details, I had to first determine which directory I am in. To do so, I have imputed `pwd`. Next I needed to know if the correct file was within my directory. The ls command came into play to display `projects` directory.

```
researcher2@769682a85275:~$ pwd
/home/researcher2
researcher2@769682a85275:~$ ls
projects
researcher2@769682a85275:~$ cd projects
researcher2@769682a85275:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Jul  1 00:37 .
drwxr-xr-x 3 researcher2 research_team 4096 Jul  1 01:08 ..
-rw--w---- 1 researcher2 research_team   46 Jul  1 00:37 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Jul  1 00:37 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Jul  1 00:37 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Jul  1 00:37 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jul  1 00:37 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jul  1 00:37 project_t.txt
researcher2@769682a85275:~/projects$
```

After changing the directory, by using `cd` command, I have imputed the following command `ls -la`
This command displays a detailed list of the file contents and hidden files including the 10-character string which represents the permissions for each file or directory.

Describe the permissions string



The 10-character string, better known as the permissions string, is a command-line based permissions layout in Linux which can be used to determine the authorization levels for each file or directory. To read the 10 character string we must first know the rules:
There are 3 permissions that can be given to a file or directory - Read, Write, and Execute
They are given the letters [r] for Read, [w] for Write, and [x] for Execute
Within the 10 character strings there are different types of users within each file - User, Group, and Other
Let's put an example to this, lets look for the file `projects_k.txt`

- The 1st character can be either [d] or [-] which indicate the file type.
  ([d] for directory and [-] for file)
  As you can see in the image above, the `projects_k.txt` is considered a file due to its [-] symbol indicating file type.
- The 2nd-4th characters are represented by [r] Read, [w] Write, and [x] Execute, specifically in that order, for the permissions authorized for the **User**. There are instances when a hyphen [-] is used within the permissions to indicate that a certain type of user is not allowed a specific authorization.
  For example, the `projects_k.txt` file, the user is given read and write permissions only. The [x] (Execute) is not there and is replaced by [-]
- The 5th-7th characters are also represented by [r] Read, [w] Write, and [x] Execute or [-] Unauthorized, but this time for **Group** permissions.
  In the image above, within the `projects_k.txt` file, the group is given the same permissions read and write.
- The 8th-10th characters are also represented by [r] Read, [w] Write, and [x] Execute or [-] Unauthorized, but this time for **Other** permissions. In the image above, within the `projects_k.txt` file, Other has permissions to read and write within the file.

A better example of this would be in the project_m.txt file. As shown in the image above, `projects_k.txt` has permissions represented by `-rw-r-----`. Since the first character is a hyphen [-] this would mean that `projects_k.txt` is a file not a directory. The next 3 characters are permissions for **User**. This would mean that **User** has Read and Write permissions (Not Execute permissions), which are represented by [r] [w] and [-]. The following 3 characters are for **Group**. This would mean that that **Group** has Read only files, due to the 2

hyphens [-] representing that they do not have Write and Execute permissions. And the last 3 characters are for **Other**. This would mean that **Other** does not have any permissions.

Change file permissions

Within our organization, we do not allow **Other** to have write permissions to any files. Based on the current authorization levels within our directory, we have found that file project_k.txt has write permissions, which is a security concern. To remedy this we have used `chmod` to disallow `project_k.txt` to have access to write permissions.

```
researcher2@48246a2bf9c1:~/projects$ chmod o-w project_k.txt
researcher2@48246a2bf9c1:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Jul  1 01:04 .
drwxr-xr-x 3 researcher2 research_team 4096 Jul  1 01:33 ..
-rw--w---- 1 researcher2 research_team   46 Jul  1 01:04 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Jul  1 01:04 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Jul  1 01:04 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Jul  1 01:04 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jul  1 01:04 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jul  1 01:04 project_t.txt
```

How was this done? The `chmod` command changes the permissions on files and directories. In the image above I have used two commands in which `chmod` was one of them. Using chmod, the command `chmod o-w project_k.txt` changes the Write permissions for **Other** in file `project_k.txt` .The command `ls -la` lists the files and directories in a given directory and was used to review updates I made.

Change file permissions on a hidden file

The research team at my organization archived the file `project_x.txt` which is considered a hidden file. They informed me that no one should have Write access to this file, however the User and Group should have read access.

The image below shows the commands used:

```
researcher2@48246a2bf9c1:~/projects$ chmod u-w,g-w,g+r .project_x.txt
researcher2@48246a2bf9c1:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Jul  1 01:04 .
drwxr-xr-x 3 researcher2 research_team 4096 Jul  1 01:33 ..
-r--r----- 1 researcher2 research_team   46 Jul  1 01:04 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Jul  1 01:04 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Jul  1 01:04 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Jul  1 01:04 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jul  1 01:04 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jul  1 01:04 project_t.txt
```

The command `chmod` was used to change the permissions for the hidden file and the command `ls -la` was used to display the output for verification of task completion. What defines a hidden file is the [.] symbol in front of the file name. For this task, Write permissions were removed from **User** and **Group** by the `[u-w],[g-w]` argument after `chmod` and **Group** was given access to Read from the `[g+r]` argument following after.

Change directory permissions

In addition to the previous task, my organization only wants the `researcher2` user to have access to the `drafts` directory and the contents within it. This would mean that the **User** should be the only one with Execute permissions. This is because having Execute permissions on a directory authorizes you to look at extended information on files in the directory. Lacking these permissions would limit the other permissions from operating accordingly.

```
researcher2@48246a2bf9c1:~/projects$ chmod g-x drafts
researcher2@48246a2bf9c1:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Jul  1 01:04 .
drwxr-xr-x 3 researcher2 research_team 4096 Jul  1 01:33 ..
-r--r----- 1 researcher2 research_team   46 Jul  1 01:04 .project_x.txt
drwx------ 2 researcher2 research_team 4096 Jul  1 01:04 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Jul  1 01:04 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Jul  1 01:04 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jul  1 01:04 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jul  1 01:04 project_t.txt
```

The image above demonstrates how I used the commands to change the permissions. The command `chmod` was used again to change the permissions for Group to disallow them to use execute permissions. The command `ls -la` was used to display the permissions and validate that the `chmod` command was executed correctly.

Summary

In this task, I have changed multiple permissions to correlate with the organization's standards. Using `chmod` and `ls -la` i have completed my task to change permissions and validating that they were completed correctly on files and directories.