File permissions in Linux

Project description:

The team of researchers within my institution requires a revision of the file access rights for specific files and folders located in the projects directory. The current permissions fail to align with the necessary authorization level. Reviewing and adjusting these permissions will contribute to maintaining the security of their system. To accomplish this objective, I undertook the subsequent actions:

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@5d738f0f927b:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec  2 15:27 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec  2 15:27 ..
-rw--w---- 1 researcher2 research_team   46 Dec  2 15:27 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Dec  2 15:27 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Dec  2 15:27 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec  2 15:27 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_t.txt
researcher2@5d738f0f927b:~/projects$
```

The initial line of the screenshot exhibits the command I input, while the next lines exhibit the results obtained. The code includes all items within the projects directory. I added the 'ls' command with the '-la' flag to reveal an elaborate enumeration of file contents, which encompasses concealed files as well. The results stemming from my command reveal the existence of a solitary directory named "drafts," an obscured file denoted as ".project_x.txt," and an additional five project-related files. The sequence of ten characters in the primary column signifies the permissions assigned to each individual file or directory.

Describe the permissions string

The sequence of ten characters can be analyzed to ascertain the individuals authorized to access the file along with their specific permissions. The characters hold the following meanings:

First character: It can be either 'd' or '-' and signifies the type of the file. 'd' indicates a directory, while '-' denotes a regular file.

Second to fourth characters: These indicate the user's read (r), write (w), and execute (x) permissions. A '-' in place of any of these characters suggests the absence of that particular permission for the user.

Fifth to seventh characters: These represent the group's read (r), write (w), and execute (x) permissions. Similar to above, a '-' indicates the lack of the specified permission for the group.

Eighth to tenth characters: These correspond to the read (r), write (w), and execute (x) permissions for others. This category encompasses all non-user and non-group entities on the system. An appearance of '-' signifies the nonexistence of the designated permission for others.

As an illustration, let's take the file permissions for "**project_t.txt**" as an example, which are depicted as "**-rw-rw-r--**". In this case:

The initial character "-" denotes that "**project_t.txt**" is indeed a file, not a directory.
The second, fifth, and eighth characters, all being "r", indicate that the user, group, and others possess read permissions.
The third and sixth characters, both "w", signify that exclusively the user and group hold write permissions.
No entity holds execute permissions for "**project_t.txt**".

Change file permissions

The organization's decision was to restrict write access for "other" on all their files. To adhere to this directive, I consulted the earlier provided file permissions information. Subsequently, I identified that "project_k.txt" required the removal of write access for "other."

Below is the representation of how I added Linux commands to accomplish this:

chmod o-w project_k.txt

This command ensures that the write permission for "other" is removed from the "project_k.txt" file.

The initial couple of lines in the screenshot exhibit the commands I input, while the subsequent lines showcase the output originating from the second command. The purpose of the 'chmod' command is to alter permissions for both files and directories. The first parameter signifies the permissions to be modified, and the second parameter designates the target file or directory. In this specific case, I eliminated the write permissions for "other" on the "**project_k.txt**" file. Following this, I employed the **'ls -la'** command to assess the modifications I enacted.

Change file permissions on a hidden file

The research team within my institution recently archived "project_x.txt." Their intent is to disallow any write access to this project, while still permitting read access for both the user and the group.

Below, you'll find the example of how I employed Linux commands to achieve this:

chmod o-w project_x.txt
chmod ug+r project_x.txt
The first command, chmod o-w project_x.txt, ensures that write permission is revoked from "other" users for the "project_x.txt" file.

The second command, chmod ug+r project_x.txt, grants read permission to both the user (u) and the group (g) for the "project_x.txt" file.

This sequence of commands facilitates the desired permission changes as specified by the research team.

```
researcher2@3213bbc1d047:~/projects$ chmod u-w,g-w,g+r .project_x.txt
researcher2@3213bbc1d047:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec 20 15:36 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec 20 15:36 ..
-r--r----- 1 researcher2 research_team   46 Dec 20 15:36 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Dec 20 15:36 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Dec 20 15:36 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec 20 15:36 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec 20 15:36 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec 20 15:36 project_t.txt
researcher2@3213bbc1d047:~/projects$
```

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.

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

```
researcher2@5d738f0f927b:~/projects$ chmod g-x drafts
researcher2@5d738f0f927b:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec  2 15:27 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec  2 15:27 ..
-r--r----- 1 researcher2 research_team   46 Dec  2 15:27 .project_x.txt
drwx------ 2 researcher2 research_team 4096 Dec  2 15:27 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec  2 15:27 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_t.txt
researcher2@5d738f0f927b:~/projects$
```

The initial two lines of the screenshot present the commands I entered, while the subsequent lines showcase the output stemming from the second command. Previously, I established that the group possessed execute permissions. In response, I utilized the 'chmod' command to eliminate these permissions.

To clarify further, the "researcher2" user was already endowed with execute permissions, rendering any additional action unnecessary.

Your command sequence and adjustments seem to align with the existing permissions and requirements of the situation.

Summary

To align the permissions with your organization's desired level of authorization for the files and directories within the **projects** directory, you executed a series of steps. The initial action involved utilizing the **'ls -la'** command to inspect the permissions of the directory. This examination served as a foundation for your subsequent decisions.

Employing this information, you then executed the **'chmod'** command multiple times to enact changes in permissions for various files and directories. By proceeding in this manner, you systematically adjusted the permissions to match your organization's specifications.