File permissions in Linux

Project description

The organization's research team must revise the file permissions for specific files and directories within the `projects` directory. The existing settings do not align with the required level of authorization. Verifying and adjusting permissions is essential to enhance the security of their system. To accomplish this, I undertook the following steps:

Check file and directory details

The provided code illustrates my utilization of Linux commands to ascertain the 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$
```

In the first line, the `ls` command with the `-la` option is utilized for displaying a detailed list of file contents, including hidden files. The resulting command output reveals the existence of a directory named `drafts`, one hidden file labeled `.project_x.txt`, and an additional five project files. The 10-character string in the far-left column signifies the permissions configured for each directory or file.

Describe the permissions string

Breaking down the 10-character string allows us to ascertain which users are authorized to access the file and their specific permissions. Each character has the following significance:

1st characters: Either (`d`) or a hyphen (`-`):

- If `d`, it signifies a directory.
- If `-`, it indicates a regular file.

2nd-4th characters: Represent the user's permissions: read (`r`), write (`w`), and execute (`x`). If a permission is not granted to the user, it is denoted by a hyphen `-`.

5th-7th characters: Convey the same type of permissions, but extended for the group.

8th-10th characters: Same type of permissions, but now excluding the user & group. For instance, when considering the file permissions for `project_t.txt` as `-rw-rw-r--`:

- The first character `-`, denotes it as a file.
- The 2nd, 5th, & 8th characters are `r`, signifying that all users can read the file.
- The 3rd & 6th characters are `w`, so the user & group can write to the file.
- Since none of the characters are `x`, nobody has the permission to execute.

Change file permissions

The organization determined that other users should not have any write permissions for their files. In alignment with this decision, I examined the previously retrieved file permissions. After examining the previous output, I identified that `project_k.txt` has write permissions for others, which needs to be eliminated.

The following code illustrates the commands and output:

```
researcher2@5d738f0f927b:~/projects$ chmod o-w project_k.txt
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-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 first two lines that are depicted in the screenshot showcase the commands that I inputted. The first command utilizes `chmod` to modify the permissions for others (`o`), restricting their ability to write (`w`) to the file `project_k.txt`. Following this command,

the use of `ls -la` allows for a review of the applied update. The subsequent lines exhibit the output of the second command.

Change file permissions on a hidden file

My organization's research team recently archived `project_x.txt`.Their requirement is to deny write access to anyone, but allow for read access to the user and group.

Below are the Linux commands I employed to modify the permissions:

```
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$
```

The first two lines that are depicted in the screenshot showcase the commands that I inputted, while the subsequent lines exhibit the output of the second command, which reveals all of the authorized actions on files and directories within the specified directory. Note that `.project_x.txt` is a hidden file denoted by the dot (`.`) at the front of the file's name. The first command utilizes `chmod` in order to configure `.project_x.txt` to the archived permission settings mentioned above.

Change directory permissions

My organization intends to only grant `researcher2` access to the `drafts` directory and its contents.

The subsequent commands were employed:

```
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 top 2 lines are the commands that were entered. First command uses `chmod g-x` removes the permission for the group to (`x`) execute `drafts`. The second command checks that the changes were made and the rest of the image shows the output.

The first two lines that are depicted in the screenshot represent the commands that I inputted. The initial command, `chmod g-x`, removes the group's permission to execute (`x`) `drafts`. The second command is a verification check to ensure the changes were successfully implemented. The subsequent lines display the corresponding output.

Summary

To conclude, I utilized several commands in order to adhere to the organization's permission regulations for files and directories within the projects directory. The initial step involved identifying the files and directories that required adjustments. To do this, I used `ls -la` to inspect the permissions of all files and directories. Subsequently, I used the `chmod` command multiple times in order to modify these permissions and thus align them with the specified requirements for the directory.