File permissions in Linux

Project description

It is essential to verify that each team member has the appropriate level of access. If discrepancies are found, I will update the permissions to grant access only to authorized users and restrict any unauthorized access within the project directory.

Check file and directory details

The code below shows how I used Linux commands to check the current permission settings of 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 first line in the screenshot shows the command I executed, while the subsequent lines display its output. I used the ls -la command to list all files in the projects directory, including hidden ones, in a detailed format. The output reveals one directory named drafts, a hidden file called .project_x.txt, and five other project files. The 10-character string in the first column represents the permission settings for each file or directory.

Describe the permissions string

The 10-character permission string can be broken down to determine who has access to a file and what type of access they have. Here's what each segment represents:

- 1st character: Indicates the file type. A d means it’s a directory, while a hyphen (-) indicates a regular file.

- 2nd–4th characters: Represent the permissions for the user (owner) — read (r), write (w), and execute (x). A hyphen (-) in place of any letter means that specific permission is not granted.
- 5th–7th characters: Represent the permissions for the group — again using r, w, and x, or - if the permission is not granted.
- 8th–10th characters: Represent the permissions for others (all users who are neither the owner nor in the group). These also follow the r, w, x pattern.

Example:
For the file project_t.txt, the permission string is -rw-rw-r--.

- The first character is a hyphen (-), meaning it's a regular file.
- The user and group both have read (r) and write (w) permissions.
- Others have read-only access (r--).
- No one has execute (x) permission.

Change file permissions

The organization requires that "others" should not have write access to any files. To ensure compliance, I reviewed the previously returned file permissions and identified that project_k.txt had write access granted to others, which needed to be removed.

The following command shows how I used Linux to update the permissions accordingly:

```
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 in the screenshot show the commands I entered, while the remaining lines display the output of the second command. The chmod command is used to modify permissions on files and directories. Its first argument defines the permission change, and the second specifies the target file or directory. In this case, I removed

write permissions for "others" on the project_k.txt file. I then used ls -la to confirm that the changes were applied successfully.

Change file permissions on a hidden file

The research team at my organization recently archived .project_x.txt and requested that no users have write access to it. However, both the user and group should retain read access.

The following command demonstrates how I used Linux to update the file permissions accordingly:

```
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 in the screenshot show the commands I entered, and the remaining lines display the output of the second command. I identified .project_x.txt as a hidden file because its name begins with a period (.). In this example, I removed write permissions from both the user and the group, and added read permissions for the group. I accomplished this using the following commands: u-w to remove write access from the user, g-w to remove write access from the group, and g+r to grant the group read access.

Change directory permissions

My organization requires that only the researcher2 user has access to the drafts directory and its contents. This means no other users should have execute permissions for that directory.

The following command demonstrates how I used Linux to update the permissions accordingly:

```
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 output displays the permission listings for several files and directories.

- **Line 1** represents the current directory (projects),
- **Line 2** shows the parent directory (home),
- **Line 3** lists a hidden regular file named .project_x.txt,
- **Line 4** shows the drafts directory with restricted permissions.

As shown, only the researcher2 user has execute permissions on the drafts directory. Previously, the group also had execute permissions, which I removed using the chmod command. Since researcher2 already had the necessary permissions, no additional changes were required for the user.

Summary

I updated several permissions to align with my organization's required access levels for files and directories within the `projects` directory. First, I ran `ls -la` to review the current permissions, which guided my subsequent actions. Then, I used the `chmod` command multiple times to modify permissions on the relevant files and directories.