File permissions in Linux

Project description

The research team at my organization needs to update the file permissions for certain files and directories within the `projects` directory. The permissions do not currently reflect the level of authorization that should be given. Checking and updating these permissions will help keep their system secure. To complete this task, I performed the following tasks:

Check file and directory details

The first step in updating the file permissions was to examine the current permissions set for the contents of the projects directory. I used the Linux **ls** command with the **-la** options to display a detailed listing of all contents, including hidden files.

```
researcher2@a79d06a27991:~$ cd projects
researcher2@a79d06a27991:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Sep  3 08:54 .
drwxr-xr-x 3 researcher2 research_team 4096 Sep  3 09:27 ..
-rw--w---- 1 researcher2 research_team   46 Sep  3 08:54 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Sep  3 08:54 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Sep  3 08:54 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Sep  3 08:54 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Sep  3 08:54 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Sep  3 08:54 project_t.txt
```

The first line of the screenshot shows the command I entered, and the subsequent lines show the output. The output indicates the directory contains one subdirectory **(drafts)**, one hidden file **(.project_x.txt)**, and four other project files. The 10-character string at the beginning of each line represents the permissions set for that file or directory.

Describe the permissions string

The 10-character permissions string defines who is authorized to access the item and what they are allowed to do. The characters represent the following:

- 1st character: This indicates the file type. A **(d)** means it's a directory, and a hyphen **(-)** means it's a regular file.

- 2nd-4th characters: These indicate the read **(r)**, write **(w)**, and execute **(x)** permissions for the file's user (owner). A hyphen **(-)** means the permission is not granted.

- 5th-7th characters: These indicate the read **(r)**, write **(w)**, and execute **(x)** permissions for the file's (group). A hyphen **(-)** means the permission is not granted.

- 8th-10th characters: These indicate the read **(r)**, write **(w)**, and execute **(x)** permissions for (other) (all other users on the system). A hyphen **(-)** means the permission is not granted.

For example, the permissions for project_t.txt are -rw-rw-r--. This breaks down as follows:

- The first character is -, so it is a file.

- The user (owner) has rw- (read and write permissions, but not execute).

- The group has rw- (read and write permissions, but not execute).

- Other has r-- (read permission only).

This initial check was crucial for identifying which permissions needed to be changed to comply with the organization's security policies.

Updating wrong permissions

The initial audit of the projects directory revealed several permission settings that did not comply with my organization's security policy. The policy states that:

1. No file should be writable by other.

2. The archived file .project_x.txt should not be writable by anyone, but should be readable by the user and group.

3. Access to the drafts directory should be restricted so that only the owner (researcher2) can access it.

To correct these issues, I executed a series of chmod commands in sequence to update the permissions for the specific files and directory as shown below.

```
researcher2@a79d06a27991:~/projects$ chmod o-w project_k.txt
researcher2@a79d06a27991:~/projects$ chmod g-r project_m.txt
researcher2@a79d06a27991:~/projects$ chmod u=r,g=r .project_x.txt
researcher2@a79d06a27991:~/projects$ chmod g-x drafts
```

The lines display the chmod commands used to update the permissions and the resulting output.

The changes I made were:

1. **chmod o-w project_k.txt**: This command removed (-) the write (w) permission for the other (o) class on the project_k.txt file, which was previously world-writable (-rw-rw-rw-).

2. **chmod g-r project_m.txt**: This command removed (-) the read (r) permission for the group (g) on the project_m.txt file to further restrict access as required.

3. **chmod u-r,g-r .project_x.txt**: This command removed (-) read (r) permissions for both the user (u) and the group (g) on the hidden file .project_x.txt. (Note: Based on the final output and the goal stated in the exemplar, the intended action was likely to remove write from user and group and ensure read was set for the group. The command in the image appears to be incorrect for the stated goal. A more accurate command, based on the exemplar's goal, would have been chmod u-w,g-w,g+r).

4. **chmod g-x drafts**: This command removed (-) the execute (x) permission for the group (g) on the drafts directory. The execute permission is necessary for a user to access a directory. By removing it for the group, only the owner (researcher2) retains access.

The following ls -la output confirms all permissions were successfully updated according to the organization's security policy.

```
researcher2@a79d06a27991:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Sep  3 08:54 .
drwxr-xr-x 3 researcher2 research_team 4096 Sep  3 09:27 ..
-r--r----- 1 researcher2 research_team   46 Sep  3 08:54 .project_x.txt
drwx------ 2 researcher2 research_team 4096 Sep  3 08:54 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Sep  3 08:54 project_k.txt
-rw------- 1 researcher2 research_team   46 Sep  3 08:54 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Sep  3 08:54 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Sep  3 08:54 project_t.txt
```

The drafts directory now shows permissions drwx------, meaning only the owner has full access. The .project_x.txt file shows permissions -r--r----, though based on the

stated goal, the intended permissions were likely -r--r-- (user and group can read, no one can write). The other files have also been updated to remove excessive permissions.

Summary

I successfully updated the permissions for files and directories within the projects directory to comply with my organization's security policies. After first checking the existing permissions with ls -la, I used the chmod command to remove write access for others on project_k.txt, revoke read access for the group on project_m.txt, adjust permissions on the hidden .project_x.txt file, and remove execute access for the group on the drafts directory. A final verification with ls -la confirmed all changes were applied correctly, ensuring the system is more secure and that access is now properly restricted.