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 following code shows how I used Linux commands to check the current permissions 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 first line of the screenshot shows the command I ran, and the following lines show the results. I used the `ls -la` command to list all contents of the "projects" directory, including hidden files. The output shows one directory called "drafts," one hidden file named `.project_x.txt`, and five other project files. The first column of the output shows a 10-character string representing the permissions for each file or directory.

Describe the permissions string

The 10-character string shows who can access the file and what actions they can perform. Here's what each part means:

1. The first character shows the file type: "d" for directory and "-" for a regular file.

2. Characters 2-4 show the user's permissions: read (r), write (w), and execute (x). A "-" means no permission.
3. Characters 5-7 show the group's permissions, with the same read, write, and execute rules.
4. Characters 8-10 show the permissions for others (everyone else), following the same read, write, and execute rules.

For example, the file permissions for "project_t.txt" are "-rw-rw-r--". The first character "-" means it's a file, not a directory. The user, group, and others all have read permission, while only the user and group have write permission. No one has execute permission.

Change file permissions

The organization decided that others should not have write access to any of their files. To follow this policy, I reviewed the file permissions I previously retrieved and identified that write access for others needed to be removed from "project_k.txt."

The following code shows how I used Linux commands to make this change.

```
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 of the screenshot show the commands I entered, while the following lines show the output of the second command. The `chmod` command is used to modify file and directory permissions. The first argument specifies the permissions to change, and the second argument indicates the file or directory. In this case, I removed write permissions for "other" on the "project_k.txt" file. Afterward, I used `ls -la` to verify the changes I made.

Change file permissions on a hidden file

The research team at my organization recently archived "project_x.txt" and decided that no one should have write access to it, though both the user and group should have read access.

The following code shows how I used Linux commands to change 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 of the screenshot show the commands I entered, and the following lines display the output of the second command. I know that ".project_x.txt" is a hidden file because it starts with a period (.). In this example, I removed write permissions from the user and group, while adding read permissions for the group. I used `u-w` to remove write permissions from the user, `g-w` to remove write permissions from the group, and `g+r` to add read permissions for the group.

Change directory permissions

My organization wants only the "researcher2" user to have access to the "drafts" directory and its contents, meaning no one else should have execute permissions.cThe following code shows how I used Linux commands to adjust 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 shows the permission listing for several files and directories. Line 1 indicates the current directory ("projects"), while line 2 shows the parent directory ("home"). Line 3 represents a regular file named ".project_x.txt," and line 4 is the "drafts" directory, which has restricted permissions. As seen, only the "researcher2" user has execute permissions. Since the group previously had execute permissions, I used the `chmod` command to remove them. The "researcher2" user already had execute permissions, so no changes were needed for them.

Summary

I adjusted the permissions to align with my organization's desired level of access for files and directories in the "projects" directory. First, I used `ls -la` to check the current permissions, which helped guide my decisions in the next steps. Then, I used the `chmod` command several times to update the permissions on various files and directories.