File permissions in Linux

Project description

The research team in my organization needs to update the file permissions for certain files and directories within the `'projects'` directory. Currently, the permissions do not reflect the level of authorization that should be granted. Reviewing and updating these permissions will help keep the system secure. To complete this task, I performed the following actions:

Check file and directory details

The following code shows how I used Linux commands to determine 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$
```

The first line of the screenshot shows the command I entered, and the other lines show the output. The code lists all contents of the `projects` directory. I used the `ls` command with the `-la` option to show a detailed list of file contents, including hidden files. The output of my command indicates that there is a directory named `drafts`, a hidden file named `.project_x.txt`, and five other project files. The 10-character string in the first column represents the permissions set on each file or directory.

Describe the permissions string

The 10-character string can be broken down to determine who has authorization to access the file and what specific permissions they have. The characters and what they represent are as follows:

- **1st character:** is a `d` or a hyphen (`-`) and indicates the file type. If it's a `d`, it indicates a directory. If it's a hyphen (`-`), it indicates a regular file.
- **2nd to 4th characters:** indicate read (`r`), write (`w`), and execute (`x`) permissions for the user. If one of these characters is a hyphen (`-`) instead of a letter, it indicates that the user does not have that specific permission.
- **5th to 7th characters:** indicate read (`r`), write (`w`), and execute (`x`) permissions for the group. If one of these characters is a hyphen (`-`) instead of a letter, it indicates that the group does not have that specific permission.
- **8th to 10th characters:** indicate read (`r`), write (`w`), and execute (`x`) permissions for other users. This category includes all other users of the system, apart from the user and the group. If one of these characters is a hyphen (`-`) instead of a letter, it indicates that other users do not have that specific permission.

For example, suppose the file permissions for `project_t.txt` are `-rw-rw-r--`. Since the first character is a hyphen (`-`), it indicates that `project_t.txt` is a file, not a directory. The second, fifth, and eighth characters are `r`, indicating that the user, group, and other users have read permissions. The third and sixth characters are `w`, indicating that only the user and group have write permissions. No one has execute permissions for `project_t.txt`.

Change file permissions

The organization determined that other users should not have write access to any of their files. To achieve this, I based my actions on the file permissions that I obtained as a result. I determined that I needed to remove the write permission from other users for `project_k.txt`.

The following code shows how I used Linux commands to do this:

```
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 other lines show the output of the second command. The `chmod` command changes the permissions on files and directories. The first argument indicates which permissions to change, and the second argument specifies the file or directory. In this example, I removed the write permissions for others on the file `project_k.txt`. Then, I used `ls -la` to review the updates I made.

Change file permissions on a hidden file

Recently, my organization's research team archived `project_x.txt`. They do not want anyone to have write access to this project, but the user and group should have read access.

The following code shows how I used Linux commands to change 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 of the screenshot show the commands I entered, and the other lines show the output of the second command. I know that `.project_x.txt` is a hidden file because it begins with a dot (`.`). In this example, I removed the write permissions for the user and group and added read permissions for the group. I removed the write permissions for the user with `u-w`. Then, I removed the write permissions for the group with `g-w` and added read permissions for the group with `g+r`.

Change directory permissions

My organization only wants the user `researcher2` 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 shows 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 first two lines of the screenshot show the commands I entered. The other lines show the output of the second command. I had previously determined that the group had execute permissions, so I used the `chmod` command to remove these permissions. The user `researcher2` already had execute permissions, so it was not necessary to add them.

Summary

I changed several permissions to match the level of authorization my organization wanted for files and directories in the `projects` directory. The first step in this process was using `ls -la` to check the permissions for the directory. I based my decisions on this information for the next steps. Then, I used the `chmod` command multiple times to change the permissions on files and directories.