File permissions in Linux

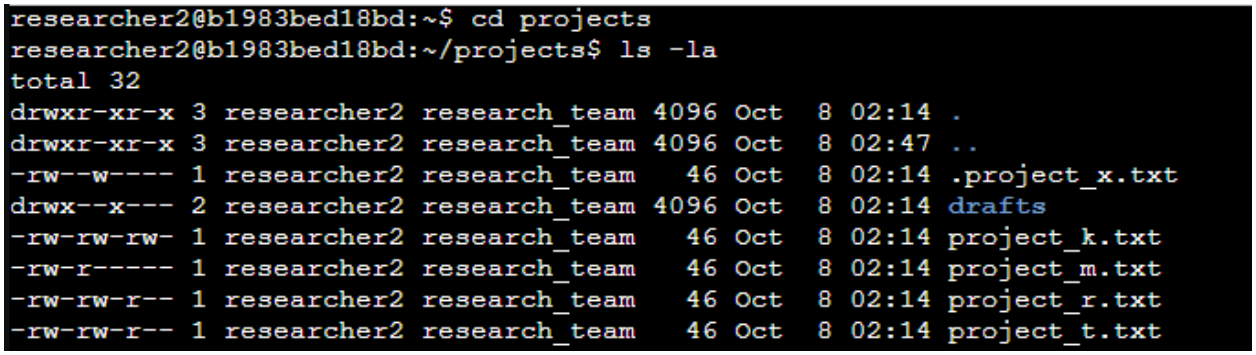
Project description

You are a security professional at a large organization. You mainly work with their research team. Part of your job is to ensure users on this team are authorized with the appropriate permissions. This helps keep the system secure.

Your task is to examine existing permissions on the file system. You'll need to determine if the permissions match the authorization that should be given. If they do not match, you'll need to modify the permissions to authorize the appropriate users and remove any unauthorized access.

Check file and directory details

The following screenshot below shows how I used Linux commands to see the existing permissions set for a directory within the system:



Starting off, I used the cd command to change my directory to the right one. I wanted to see the files & secret files within the directory so I used the `ls` command with the `-la` option to show more details about the files. The output shows one folder called "drafts," one hidden file called ".project_x.txt," and five other project files. The 10-character string in the first column shows the permissions for each file or folder.

Describe the permissions string

The permission string consist of 10-character to indicate how the permissions on the file are set. This is the string that I follow:

The 1st character indicates the file type. The d indicates it’s a directory. When this character is a hyphen (-), it's a regular file.

The 2nd-4th characters indicate the read (r), write (w), and execute (x) permissions for the user. When one of these characters is a hyphen (-) instead, it indicates that this permission is not granted to the user.

The 5th-7th characters indicate the read (r), write (w), and execute (x) permissions for the group. When one of these characters is a hyphen (-) instead, it indicates that this permission is not granted for the group.

The 8th-10th characters indicate the read (r), write (w), and execute (x) permissions for the owner type of other. This owner type consists of all other users on the system apart from the user and the group. When one of these characters is a hyphen (-) instead, that indicates that this permission is not granted for other.

To demonstrate my understanding of the string permissions, the project file project_k.txt has -rw-rw-rw- file permissions. Since the first character is a (-), project_k.txt is a file and it’s not a directory. The second, fifth and eighth characters in the string are all (r) , which indicates that the user, group and others all have read permissions. The third, sixth, ninth characters in the string are all (w) indicates that the user group and others all have write permissions. There was no (x) in the string so no one has the execute permissions.

Change file permissions

The organization decided that "other" shouldn't be able to write to any files. To fix this, I looked at the file permissions I found earlier. I saw that "project_k.txt" needs to have write access taken away from "other."

```
researcher2@b1983bed18bd:~/projects$ chmod o-w project_txt
chmod: cannot access 'project_txt': No such file or directory
researcher2@b1983bed18bd:~/projects$ chmod o-w project_k.txt
researcher2@b1983bed18bd:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Oct  8 02:14 .
drwxr-xr-x 3 researcher2 research_team 4096 Oct  8 02:47 ..
-rw--w---- 1 researcher2 research_team   46 Oct  8 02:14 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Oct  8 02:14 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Oct  8 02:14 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Oct  8 02:14 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Oct  8 02:14 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Oct  8 02:14 project_t.txt
```

The first two lines of the screenshot show the commands I typed, and the other lines show what happened after the second command. The `chmod` command is used to change file and folder permissions. The first part says what to change, and the second part tells which file or folder. In this case, I took away write permissions from "other" for the file "project_k.txt." After that, I used `ls -la` to check the changes I made.

Change file permissions on a hidden file

The research team at my organization recently saved "project_x.txt" as an archive. They don't want anyone to be able to write to it, but the user and group should still be able to read it. Here's how I used Linux commands to change the permissions:

```
researcher2@b1983bed18bd:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Oct  8 02:14 .
drwxr-xr-x 3 researcher2 research_team 4096 Oct  8 02:47 ..
-rw--w---- 1 researcher2 research_team   46 Oct  8 02:14 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Oct  8 02:14 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Oct  8 02:14 project_k.txt
-rw------- 1 researcher2 research_team   46 Oct  8 02:14 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Oct  8 02:14 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Oct  8 02:14 project_t.txt
researcher2@b1983bed18bd:~/projects$ chmod u-w,g-w,g+r .project_x.txt
researcher2@b1983bed18bd:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Oct  8 02:14 .
drwxr-xr-x 3 researcher2 research_team 4096 Oct  8 02:47 ..
-r--r----- 1 researcher2 research_team   46 Oct  8 02:14 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Oct  8 02:14 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Oct  8 02:14 project_k.txt
-rw------- 1 researcher2 research_team   46 Oct  8 02:14 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Oct  8 02:14 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Oct  8 02:14 project_t.txt
```

The first two lines of the screenshot show the commands I typed, and the other lines show what happened after the second command. I know ".project_x.txt" is hidden because it starts with a dot (.). In this example, I took away write permissions from both the user and group. I also gave the group read permissions. First, I removed write access from the user with `u-w`. Then, I took away write access from the group using `g-w` and gave the group read access with `g+r`.

Change directory permissions

My organization only wants the "researcher2" user to access the "drafts" folder and everything inside it. This means nobody else should have execute permissions. Here's how I used Linux commands to change the permissions:

```
drwxr-xr-x 3 researcher2 research_team 4096 Oct  8 02:14 .
drwxr-xr-x 3 researcher2 research_team 4096 Oct  8 02:47 ..
-r--r----- 1 researcher2 research_team   46 Oct  8 02:14 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Oct  8 02:14 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Oct  8 02:14 project_k.txt
-rw------- 1 researcher2 research_team   46 Oct  8 02:14 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Oct  8 02:14 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Oct  8 02:14 project_t.txt
researcher2@b1983bed18bd:~/projects$ chmod g-x drafts
researcher2@b1983bed18bd:~/projects$ ls- l
-bash: ls-: command not found
researcher2@b1983bed18bd:~/projects$ ls -l
total 20
drwx------ 2 researcher2 research_team 4096 Oct  8 02:14 drafts
```

You can see that only "researcher2" has execute permissions now. Before, the group had execute permissions too, so I used the `chmod` command to take them away. "Researcher2" already had execute permissions, so I didn't need to add them.

Summary

I changed several permissions to match what my organization wanted for files and folders in the projects directory. First, I used `ls -la` to check the current permissions. This helped me decide what changes to make. Then, I used the `chmod` command multiple times to adjust the permissions on the files and folders.