File permissions in Linux

Project description

I was assigned by the research team to review the current permissions of a Linux file system. My task is to ensure that the existing permissions align with the correct access rights for users. If not, I will need to adjust the permissions to grant access only to authorized users and remove any unauthorized access. This process is essential for maintaining security.

Check file and directory details

To begin, I examine the permissions of the `projects` directory and the files it contains.

```
researcher2@f9886f189c5d:~$ cd projects
researcher2@f9886f189c5d:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Oct 14 10:10 .
drwxr-xr-x 3 researcher2 research_team 4096 Oct 14 10:46 ..
-rw--w---- 1 researcher2 research_team   46 Oct 14 10:10 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Oct 14 10:10 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Oct 14 10:10 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Oct 14 10:10 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Oct 14 10:10 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Oct 14 10:10 project_t.txt
researcher2@f9886f189c5d:~/projects$
```

The first two lines of code are what I entered into the command line, the rest is the output. I use `cd projects` to navigate to the `projects` directory. Then I run `ls -la` to list detailed file permissions, including the hidden ones.

Describe the permissions string

Each file entry begins with a 10-character string and defines its type and access permissions.

For example, a directory with full access for all user categories would be `drwxrwxrwx`:

- **Character 1**: File type
    - `d` ➔ directory

    - `-` ➔ regular file
- **Characters 2–4**: Permissions for the **user**
- **Characters 5–7**: Permissions for the **group**
- **Characters 8–10**: Permissions for **others** (everyone else on the system)

Permission characters:

- `r` ➔ read permission
- `w` ➔ write permission
- `x` ➔ execute permission
- `-` ➔ permission not granted.

For example, the permissions given to file `project_r.txt` are listed as `-rw-rw-r--`, which means the following:

- **Character 1:** A hyphen (`-`) indicates that `project_r.txt` is a regular file, not a directory.
- **Characters 2-4:** User has read (`r`) and write (`w`) access, but no execute permissions.
- **Characters 5–7**: Group also has read (`r`) and write (`w`) access, but no execute permissions.
- **Characters 8–10:** Others are limited to only read (`r`) access, with no write or execute permissions.

Change file permissions

The organization decided that others should not have any access to write permissions on any files. To implement this I need to use the `chmod` command. The only file identified that has access to write permissions is `project_k.txt.`

```
researcher2@f9886f189c5d:~/projects$ chmod o-w project_k.txt
researcher2@f9886f189c5d:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Oct 14 10:10 .
drwxr-xr-x 3 researcher2 research_team 4096 Oct 14 10:46 ..
-rw--w---- 1 researcher2 research_team   46 Oct 14 10:10 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Oct 14 10:10 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Oct 14 10:10 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Oct 14 10:10 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Oct 14 10:10 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Oct 14 10:10 project_t.txt
researcher2@f9886f189c5d:~/projects$
```

The first two lines of code are what I entered into the command line, the rest is the output. The `chmod` command changes permissions on files and directories, and it requires two arguments. The first argument indicates how to change permissions, and the second argument indicates the file or directory that you want to change permissions. For example, my first argument is `o-w`, which means I want to take away write permissions from others. Then comes my second argument, `project_k.txt`, which is the file I want to modify. Then I verified with `ls -la` to analyze my changes.

Change file permissions on a hidden file

The research team has archived `.project_x.txt`, that's why it is designated as a hidden file. This file should not have write permissions for anyone, but the user and group should be able to read the file. This is how I assign the appropriate authorization to the file.

```
researcher2@f9886f189c5d:~/projects$ chmod u-w,g-w,g+r .project_x.txt
researcher2@f9886f189c5d:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Oct 14 10:10 .
drwxr-xr-x 3 researcher2 research_team 4096 Oct 14 10:46 ..
-r--r----- 1 researcher2 research_team   46 Oct 14 10:10 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Oct 14 10:10 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Oct 14 10:10 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Oct 14 10:10 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Oct 14 10:10 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Oct 14 10:10 project_t.txt
researcher2@f9886f189c5d:~/projects$
```

The first two lines of code are what I entered into the command line, the rest is the output. I identify `.project_x.txt` as hidden because the filename starts with a period (`.`). The first argument indicates what permissions should be changed, and the second argument specifies

the file or directory. In this example, I removed write permissions from the user with `u-w`. Then, I removed write permissions from the group with `g-w`, and added read permissions to the group with `g+r`. Then I verified with `ls -la` to analyze my changes.

Change directory permissions

The files and directories in the `projects` directory belong to the `researcher2` user. To ensure proper access control, only `researcher2` should be able to access the `drafts` directory and its contents. This means no other users should have execute permissions on that directory. This is how I assign the appropriate authorization to the file.

```
researcher2@f9886f189c5d:~/projects$ chmod g-x drafts
researcher2@f9886f189c5d:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Oct 14 10:10 .
drwxr-xr-x 3 researcher2 research_team 4096 Oct 14 10:46 ..
-r--r----- 1 researcher2 research_team   46 Oct 14 10:10 .project_x.txt
drwx------ 2 researcher2 research_team 4096 Oct 14 10:10 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Oct 14 10:10 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Oct 14 10:10 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Oct 14 10:10 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Oct 14 10:10 project_t.txt
researcher2@f9886f189c5d:~/projects$
```

By looking at the previous outputs, you can see that the group had execute permissions on `drafts`, which needed to be removed. The first argument indicates what permissions should be changed, and the second argument specifies the file or directory. In this example, I use `g-x` to remove execute permission from the group. The user `researcher2` already had the necessary execute permissions, so no other changes needed to be made. Then I verified with `ls -la` to analyze my changes.

Summary

This task was conducted in a Linux system to update key file and directory permissions within the `projects` directory, aligning with organizational access policies. Write access for unauthorized users was removed, read only settings were applied where appropriate, and exclusive access was granted to designated users. These changes were implemented using

Linux commands like `chmod` and `ls -la`, strengthening system security and ensuring proper access control across sensitive resources.