File permissions in Linux

Project description

The research team needs to update the file permissions for certain files and directories within the `projects` directory. The permission does not currently reflect level of authorization that should be given. Checking and updating the permissions is believed to be helpful in keeping the system secure. To complete the task, the following has been performed:

Check file and directory details

The code demonstrates how Linux commands are utilized to determine the existing permission set for a specific directory in the file system.

```
researcher2@c64659b3717a:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Nov  1 02:45 .
drwxr-xr-x 3 researcher2 research_team 4096 Nov  1 04:57 ..
-rw--w---- 1 researcher2 research_team   46 Nov  1 02:45 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Nov  1 02:45 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Nov  1 02:45 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Nov  1 02:45 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Nov  1 02:45 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Nov  1 02:45 project_t.txt
researcher2@c64659b3717a:~/projects$
```

The first line of the code displays the command that will list all contents of the `projects` directory. The `ls` command with the `-la` option to display a detailed listing of the file contents that also returned hidden files. The output of the command indicates that there is one directory named `drafts`, one 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 deconstructed to determine who is authorized to access the file and their specific permissions. The characters and what they represent are as follows:

- **1st character**: This character is either a `d` or hyphen (`-`) and indicates the file type. If it's a `d`, it's a directory. If it's a hyphen (`-`), it's a regular file.
- **2nd-4th characters**: These 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.

- **5th-7th characters:** These 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.
- **8th-10th characters:** These characters indicate the read (`r`), write (`w`), and execute (`x`) permissions for 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.

For example, the file permissions for `project_t.txt` are `-rw-rw-r--`. Since the first character is a hyphen (`-`), this indicates that `project_t.txt` is a file, not a directory. The second, fifth, and eighth characters are all `r`, which indicates that user, group, and other all have read permissions. The third and sixth characters are `w`, which indicates 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 others shouldn't have write access to any of their files. To comply with this, I referred to the file permissions that I previously returned. I determined `project_k.txt` must have the write access removed for others.

The following code demonstrates how Linux commands was used to do this:

```
researcher2@c64659b3717a:~/projects$ chmod o-w project_k.txt
researcher2@c64659b3717a:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Nov  1 02:45 .
drwxr-xr-x 3 researcher2 research_team 4096 Nov  1 04:57 ..
-rw--w---- 1 researcher2 research_team   46 Nov  1 02:45 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Nov  1 02:45 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Nov  1 02:45 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Nov  1 02:45 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Nov  1 02:45 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Nov  1 02:45 project_t.txt
researcher2@c64659b3717a:~/projects$
```

The first two lines of code displayed the commands entered, and the other lines was the displayed output for the command 2. The `chmod` command changes the permissions on files and directories. The first argument indicates what permissions should be changed, and the second argument specifies the file or directory. In the executed command, the write permission was removed from other for the `project_k.txt` file. After this, the `ls -la` command was used to review the updates made.

Change file permissions on a hidden file

The research team at my organization recently 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 demonstrates how Linux commands was used to change the permissions:

```
researcher2@c64659b3717a:~/projects$ chmod u-w,g-w,g+r .project_x.txt
researcher2@c64659b3717a:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Nov  1 02:45 .
drwxr-xr-x 3 researcher2 research_team 4096 Nov  1 04:57 ..
-r--r----- 1 researcher2 research_team   46 Nov  1 02:45 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Nov  1 02:45 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Nov  1 02:45 project_k.txt
-rw------- 1 researcher2 research_team   46 Nov  1 02:45 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Nov  1 02:45 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Nov  1 02:45 project_t.txt
researcher2@c64659b3717a:~/projects$
```

The first two lines of the image displayed the commands entered, and the other lines display the output of the second command. `.project_x.txt` is a hidden file because it starts with a period (`.`). In this, The write permissions was removed from the user and group, and added read permissions to the group. The write permissions was removed from the user with `u-w`. Then, removed write permissions from the group with `g-w`, and added read permissions to the group with `g+r`.

Change directory permissions

My organization only wants the `researcher2` user 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 demonstrates how Linux commands is used to change the permissions:

```
researcher2@c64659b3717a:~/projects$ chmod g-x drafts
researcher2@c64659b3717a:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Nov  1 02:45 .
drwxr-xr-x 3 researcher2 research_team 4096 Nov  1 04:57 ..
-r--r----- 1 researcher2 research_team   46 Nov  1 02:45 .project_x.txt
drwx------ 2 researcher2 research_team 4096 Nov  1 02:45 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Nov  1 02:45 project_k.txt
-rw------- 1 researcher2 research_team   46 Nov  1 02:45 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Nov  1 02:45 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Nov  1 02:45 project_t.txt
researcher2@c64659b3717a:~/projects$
```

The first two lines of the screenshot display the commands entered, and the other lines displayed the output of the second command. I previously determined that the group had execute permissions, so the `chmod` command is used to remove them. The `researcher2` user already had execute permissions, so they did not need to be added.