File permissions in Linux

Dillon Tall

Project description

The main purpose of the project is to demonstrate Linux file permissions through the use of the “Manage authorization” Qwiklab. The format and use of file and directory permissions will be explained with accompanying documentation of screenshots that shows an understanding of these permissions. There will then be multiple tests of changing file and directory permissions at the end of the project.

The scenario for this project is for a security professional at an organization that needs to ensure that users on a certain team are authorized with appropriate permissions to keep the system secure. The main task is to view each of the permissions on the files and directories in the file system and determine if they match the actual authorization that is given. Modifications will need to be made in the case where the permissions and authorizations don't match to remove any unauthorized access.

Check file and directory details

Using the “ls” command combined with -al shows all of the files in the current linux working directory. Using this command outputs the following text:

```
researcher2@f4d4d33d1def:~/projects$ ls -al
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Jul 24 03:36 .
drwxr-xr-x 3 researcher2 research_team 4096 Jul 24 04:46 ..
-rw--w---- 1 researcher2 research_team   46 Jul 24 03:36 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Jul 24 03:36 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Jul 24 03:36 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Jul 24 03:36 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jul 24 03:36 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Jul 24 03:36 project_t.txt
```

This text appears as the output of the bash shell and represents all of the files and directories within the directory “projects” along with their permissions in regards to the user, group, and all other users.

Describe the permissions string

Focusing specifically on one of the files above graphic, being “project_k.txt” which is the 4th file listed, we can explain how permissions work in the format of the output. The first part of

each output is represented through 10 characters, which is all of the permissions of the file/directory. The first character is to represent if the permissions are for a file or a directory, where “d” is directory and the character “-” is a file. The rest of the 9 characters are split into 3 groups, being the user, group, and other users permissions in that respective order. They all have 3 potential permissions, which are the read write and execute permissions. For example, the project_k.txt permissions contain “ r w -”. Since this is the first of the 3 permission groups, it represents the user having read and write permissions and the character of “-” to show that the user does not have execute permissions. This follows for the rest of the groups.

Change file permissions

Based on the scenario of the project, we can observe that the “other” group, denoted by the final set of three characters in each beginning string, should not have the permission to write on any of the files. From the first figure of the project, we can see that “project_k.txt” includes write permissions, “w”, for the other group, which should be changed. To change this, we can use the command “chmod” along with “o-w”, meaning others should have the write permission removed, denoted by the “-” symbol. The full command should read as: “chmod o-w project_k.txt”, which fixes the issue

Change file permissions on a hidden file

The scenario also describes the archived “.project_x.txt” file, which should not have write permissions for anyone, but should still be accessible to the user and the group. We can observe this file by once again using “ls -al”, which reveals the hidden file.

```
-rw--w---- 1 researcher2 research_team   46 Jul 24 03:36 .project_x.txt
```

Using multiple linux commands, we can describe a simple step by step solution to the problem:

Chmod u-w .project_x.txt (removes the write privilege for the user)

Chmod g-w .project_x.txt (removes the write privilege for the group)

Chmod g+r .project_x.txt (Adds the read privilege for the group)

These 3 commands change the file permissions of the hidden file so that the user and the group can exclusively read the hidden file, but not modify or execute it.

Change directory permissions

For the final part of the project, we were tasked with changing a directories permission to only allow the user “researcher2” to access the directory. The directory “projects” is listed below, denoted with a character “d” to show that we are working with a directory.

```
drwxr-xr-x 3 researcher2 research_team 4096 Jul 24 03:36 projects
```

To make the necessary changes, we again use “chmod” to modify the directories permissions. For this specific directory, all we need to do is remove all of the permissions for the group and “other”, as the user already has all permissions.
chmod g-rx projects (removes the read and execute permission for the group)
chmod o-rx projects (removes the read and execute permission for other)

Using these two commands leaves only the user with full permissions to access the directory.

Summary

In this project, we were tasked with securing certain files and directories in a company by changing permissions using bash in linux and the “chmod” command. We were able to add, remove and change these permissions to fit the guidelines set by the scenario, securing the files and directories so that only groups that had access to them were able to interact with them.