File permissions in Linux

Project description

For this project, I am acting as a security professional at a large organization. I mainly work with their research team. Part of my job is to ensure users on this team are authorized with the appropriate permissions. This helps keep the system secure. My task is to examine existing permissions on the file system and determine if the permissions match the authorization that should be given. If they do not match, I'll need to modify the permissions to authorize the appropriate users and remove any unauthorized access.

Check file and directory details

```
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Feb  1 15:08 .
drwxr-xr-x 3 researcher2 research_team 4096 Feb  1 15:36 ..
-rw--w---- 1 researcher2 research_team   46 Feb  1 15:08 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Feb  1 15:08 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Feb  1 15:08 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Feb  1 15:08 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Feb  1 15:08 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Feb  1 15:08 project_t.txt
```

Describe the permissions string

Permission string: -rw-r----- 1 researcher2 research_team   46 Feb 1 15:08 project_m.txt

- a 10-character string begins each entry and indicates how the permissions on the file are set
- The first character is a (-) which indicates that this is a file named project_m.txt
- The second-fourth characters refers to the owner type: user which says the user has read and write permission but doesn't have 'x' permission which stands for execute permission.
- The fifth-seventh characters refers to the owner type: group which says that the group only has read permissions but no write or execute permissions
- The eighth-tenth characters refers to the owner type: other which says the other does not have any permissions as indicated by the 3 consecutive hyphens

- To see the permissions on the files and directories I used the ls command with the la argument to show the permissions for both unhidden and hidden files and directories

Change file permissions

Permission string: -rw-rw-rw- 1 researcher2 research_team 46 Feb 1 15:08 project_k.txt

- This file gives the owner type: other permission to write on the file
- Using the command: chmod o-w project_k.txt changes the permissions of this file by taking away the write permissions for the other owner type. The chmod command changes the permissions on a file or directory. o stands for owner, the (-) operator means to remove and the w stands for write permission. The second argument is denoting the file that you would like to change permissions on.

Change file permissions on a hidden file

Permission string: -rw--w---- 1 researcher2 research_team 46 Feb 1 15:08 .project_x.txt

- The file type should not have write permissions for anyone but the user and group should be able to read the file.
- Using the command chmod u-w,g-w,g+r .projects_x.txt removes the writing permissions on both the user and group but adds a read permission on the group.

Change directory permissions

Permission string: drwx--x--- 2 researcher2 research_team 4096 Feb 1 15:08 drafts

- This indicates that the group in addition to the user has execute permissions on the drafts directory and its contents.
- Using the command chmod g-x drafts removes the execute permissions for the owner type: group from the drafts directory.

Summary

- In this activity, I was tasked with reviewing the existing permissions on the file system and determine the appropriate level of authorization needed for each owner for the files and directories listed. With each scenario I had to read what permissions were allowed for each directory and file then input commands through the bash shell to change authorization for each user and group on the file system.