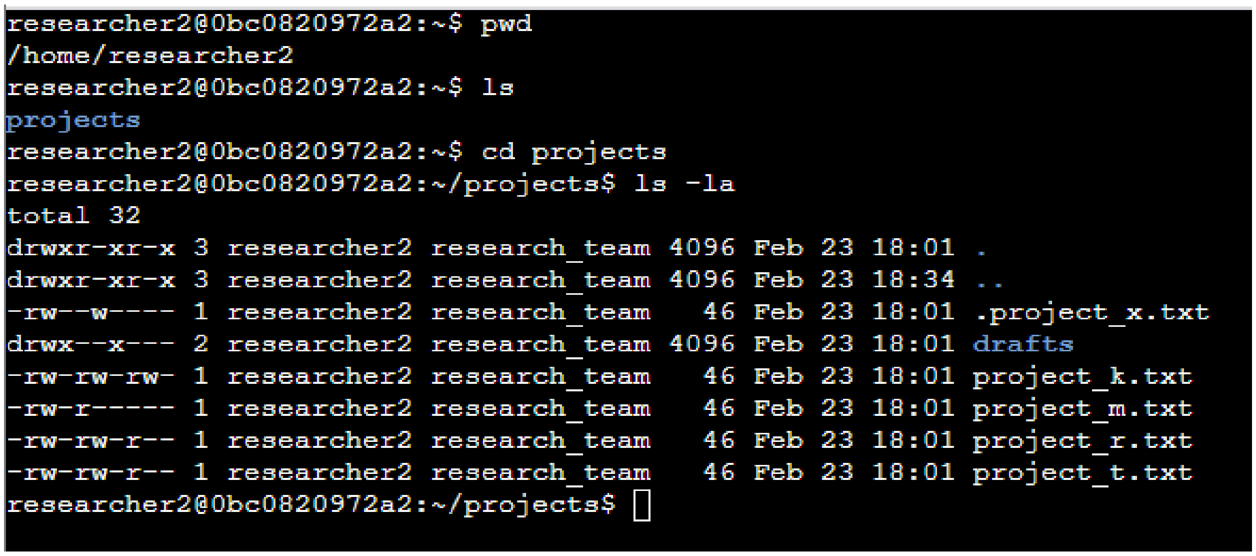
File permissions in Linux

Project description

The research team at my organization needs to review and update the file permissions for various files and directories within the project directory. Currently, the permissions do not accurately reflect the intended access levels for users, which could potentially lead to unauthorized access or insufficient privileges. Ensuring the correct permissions are in place is critical for maintaining the system's security and integrity.
To complete this task, I carried out the following actions:

Check file and directory details **(add more detail about first 3 lines of codes)**

The following code demonstrates how I used Linux commands to check the current permissions set for a specific directory in the file system.



```
researcher2@0bc0820972a2:~$ pwd
/home/researcher2
researcher2@0bc0820972a2:~$ ls
projects
researcher2@0bc0820972a2:~$ cd projects
researcher2@0bc0820972a2:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Feb 23 18:01 .
drwxr-xr-x 3 researcher2 research_team 4096 Feb 23 18:34 ..
-rw--w---- 1 researcher2 research_team   46 Feb 23 18:01 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Feb 23 18:01 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Feb 23 18:01 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Feb 23 18:01 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Feb 23 18:01 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Feb 23 18:01 project_t.txt
researcher2@0bc0820972a2:~/projects$ 
```

The screenshot above displays the command that was entered as well as the subsequent results. The code lists all the contents of the projects directory. I used the ls command with the -la option to show a detailed listing of the files, including hidden ones. The output reveals one directory called "drafts," one hidden file named .project_x.txt, and five other project files. The 10-character string in the first column indicates the permissions set for each file or directory.

Describe the permissions string

The 10-character string can be broken down to determine who has access to the file and their specific permissions. The characters and their meanings are as follows:

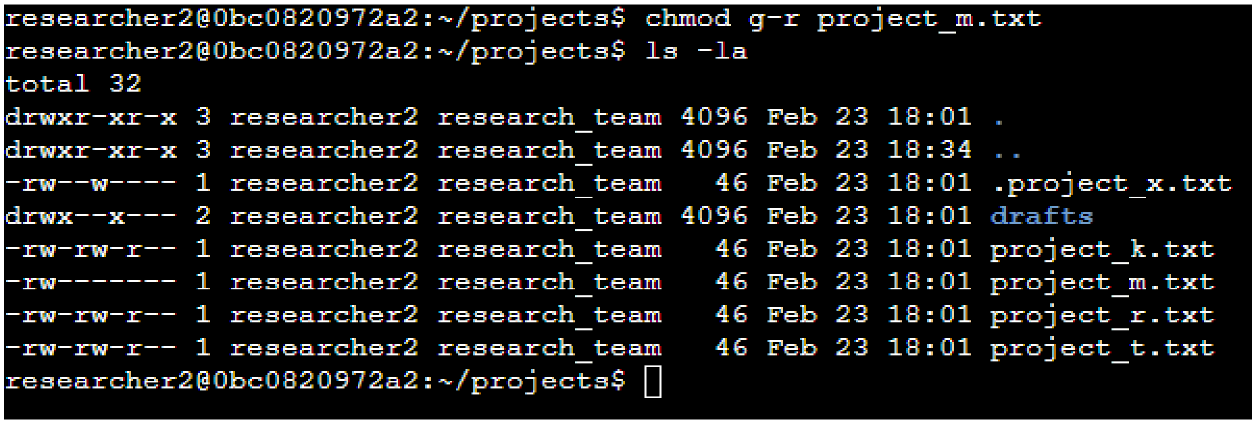
- **1st character**: This character is either a d or a hyphen (-) and denotes the file type. A d represents a directory, while a hyphen (-) indicates a regular file.
- **2nd-4th characters**: These characters represent the read (r), write (w), and execute (x) permissions for the user. If a character is a hyphen (-), it means that permission is not granted to the user.
- **5th-7th characters**: These characters represent the read (r), write (w), and execute (x) permissions for the group. A hyphen (-) indicates that permission is not granted to the group.
- **8th-10th characters**: These characters represent the read (r), write (w), and execute (x) permissions for others, referring to all users on the system excluding the file's user and group. A hyphen (-) here also means that the permission is not granted to others.

For example, the file permissions for project_t.txt are -rw-rw-r--. The first character is a hyphen (-), indicating that project_t.txt is a file, not a directory. The second, fifth, and eighth characters are all r, meaning the user, group, and others all have read permissions. The third and sixth characters are w, meaning only the user and group have write permissions. No one has execute permissions for project_t.txt.

Change file permissions

The organization decided that others should not have write access to any of their files. To adhere to this policy, I reviewed the file permissions I had previously retrieved and identified that the write access for others on project_m.txt needed to be removed.

The following code shows how I used Linux commands to accomplish this:

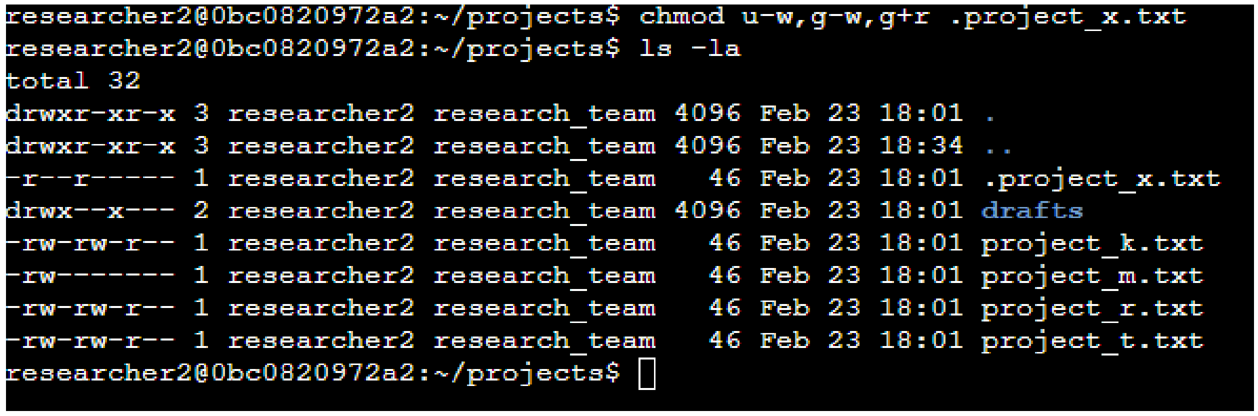


```
researcher2@0bc0820972a2:~/projects$ chmod g-r project_m.txt
researcher2@0bc0820972a2:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Feb 23 18:01 .
drwxr-xr-x 3 researcher2 research_team 4096 Feb 23 18:34 ..
-rw--w---- 1 researcher2 research_team   46 Feb 23 18:01 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Feb 23 18:01 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Feb 23 18:01 project_k.txt
-rw------- 1 researcher2 research_team   46 Feb 23 18:01 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Feb 23 18:01 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Feb 23 18:01 project_t.txt
researcher2@0bc0820972a2:~/projects$
```

The first two lines of the screenshot show the commands I entered, while the remaining lines display the output of the second command. The chmod command is used to modify file and directory permissions. The first argument specifies which permissions should be altered, and the second argument identifies the file or directory. In this case, I removed write permissions for others on the project_m.txt file. Afterward, I used ls -la to verify the changes I made.

Change file permissions on a hidden file

The research team at my organization recently archived .project_x.txt and wants to prevent anyone from having write access to this file, while ensuring that both the user and group retain read access.
The following code shows how I used Linux commands to update the permissions:

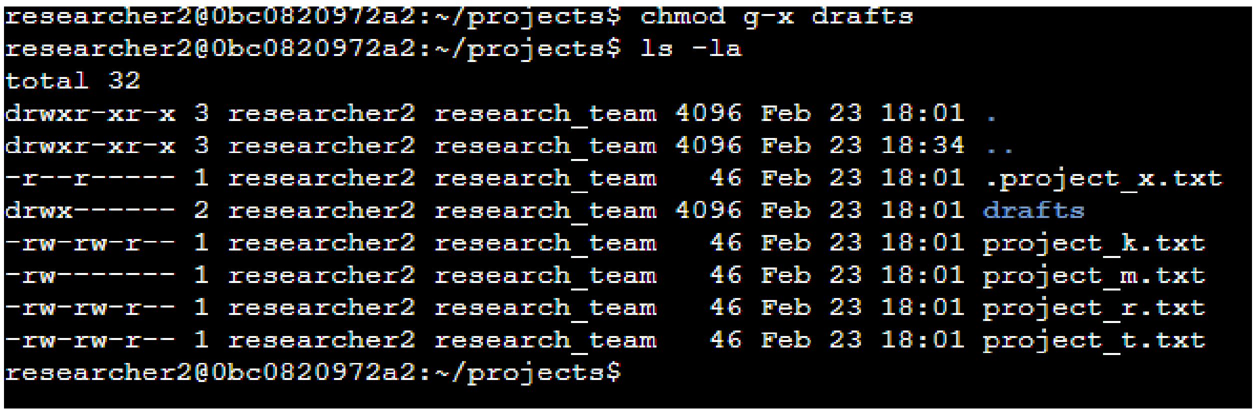


```
researcher2@0bc0820972a2:~/projects$ chmod u-w,g-w,g+r .project_x.txt
researcher2@0bc0820972a2:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Feb 23 18:01 .
drwxr-xr-x 3 researcher2 research_team 4096 Feb 23 18:34 ..
-r--r----- 1 researcher2 research_team   46 Feb 23 18:01 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Feb 23 18:01 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Feb 23 18:01 project_k.txt
-rw------- 1 researcher2 research_team   46 Feb 23 18:01 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Feb 23 18:01 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Feb 23 18:01 project_t.txt
researcher2@0bc0820972a2:~/projects$ 
```

The first two lines of the screenshot show the commands I entered, while the remaining lines display the output of the second command. I can identify that .project_x.txt is a hidden file because its name begins with a period (.). In this example, I removed write permissions from both the user and group and added read permissions for the group. I used u-w to remove write permissions from the user, g-w to remove write permissions from the group, and g+r to grant read permissions to the group.

Change directory permissions

My organization wants only the user researcher2 to have access to the drafts directory and its contents, ensuring that no one else has execute permissions.
The following code demonstrates how I used Linux commands to modify the permissions:



```
researcher2@0bc0820972a2:~/projects$ chmod g-x drafts
researcher2@0bc0820972a2:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Feb 23 18:01 .
drwxr-xr-x 3 researcher2 research_team 4096 Feb 23 18:34 ..
-r--r----- 1 researcher2 research_team   46 Feb 23 18:01 .project_x.txt
drwx------ 2 researcher2 research_team 4096 Feb 23 18:01 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Feb 23 18:01 project_k.txt
-rw------- 1 researcher2 research_team   46 Feb 23 18:01 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Feb 23 18:01 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Feb 23 18:01 project_t.txt
researcher2@0bc0820972a2:~/projects$
```

The first two lines of the screenshot show the commands I entered, while the remaining lines display the output of the second command. Earlier, I identified that the group had execute permissions, so I used the `chmod` command to remove them. Since the `researcher2` user already had execute permissions, there was no need to add them again.

Summary

In this project, I reviewed and updated file and directory permissions for the research team to ensure they align with organizational policies. I first examined the current permissions for various files using the `ls -la` command, which lists all the files and their detailed permissions. I then modified the permissions as needed using the chmod command to prevent unauthorized access and ensure only the appropriate users had the required access. Specifically, I removed unnecessary write permissions for others using `chmod o-w project_k.txt`, updated permissions for hidden files (like `.project_x.txt`) by using chmod `u-w`, `g-w .project_x.txt` and `chmod g+r .project_x.txt`, and adjusted directory access for a specific user by using `chmod g-x drafts` to remove the group's execute permissions. These actions help secure the system and ensure that access to sensitive files is properly controlled.