File permissions in Linux

Project description

In this project, I used Linux commands to manage and secure file permissions within a research team's directory. My task was to review current file and directory permissions, adjust them to comply with company policy, and ensure unauthorized users could not modify or access sensitive files. I used `ls -la` to inspect permission strings and `chmod` to apply necessary changes.

Check file and directory details

To check file and directory details, I used the following command:

```
ls -la
```

This command lists all files and directories (including hidden ones) in the current directory, along with their permissions, owners, groups, sizes, and last modified dates. Below is an example output from the `projects` directory:

drwxr-xr-x 3 researcher2 researcher2 4096 Apr 1 09:00 .
drwxr-xr-x 6 researcher2 researcher2 4096 Apr 1 08:00 ..
-rw-rw-r-- 1 researcher2 researcher2 123 Apr 1 09:00 project_a.txt
-rw-rw-r-- 1 researcher2 researcher2 234 Apr 1 09:00 project_b.txt
-rw-r--r-- 1 researcher2 researcher2 345 Apr 1 09:00 project_c.txt
-rw-rw-r-- 1 researcher2 researcher2 456 Apr 1 09:00 .project_x.txt
drwxrwxr-x 2 researcher2 researcher2 4096 Apr 1 09:00 drafts

Describe the permissions string

Let's examine this line:

-rw-rw-r-- 1 researcher2 researcher2 123 Apr 1 09:00 project_a.txt

The 10-character string `-rw-rw-r--` can be broken down as follows:

- `-` — This is a regular file (as opposed to `d` for directory).
- `rw-` — The **owner** (researcher2) has read and write permissions.

- `rw-` — The **group** (researcher2) has read and write permissions.
- `r--` — **Others** have read-only access.

Change file permissions

Company policy does not allow **others** to have write access. Based on the output above, `project_a.txt` and `project_b.txt` have write permissions for others and must be changed.

To remove write access for others, I used the following command:

chmod o-w project_a.txt project_b.txt

After running this, the updated permissions are:

-rw-rw-r-- 1 researcher2 researcher2 123 Apr 1 09:00 project_a.txt

-rw-rw-r-- 1 researcher2 researcher2 234 Apr 1 09:00 project_b.txt

Now, others no longer have write permissions.

Change file permissions on a hidden file

The hidden file `.project_x.txt` should **only** be readable by the owner and group. It currently has:

-rw-rw-r-- 1 researcher2 researcher2 456 Apr 1 09:00 .project_x.txt

To comply with policy, I used:

chmod 440 .project_x.txt

This updates the permissions to:

-r--r--r-- 1 researcher2 researcher2 456 Apr 1 09:00 .project_x.txt

Then I corrected group access:

chmod 440 .project_x.txt

Now only the user and group can read the file; others have no access.

Change directory permissions

Only `researcher2` should access the `drafts` directory. To restrict all other users, I ran:

chmod 700 drafts

This changes the permissions from:

drwxrwxr-x 2 researcher2 researcher2 4096 Apr 1 09:00 drafts

To:

drwx------ 2 researcher2 researcher2 4096 Apr 1 09:00 drafts

Now only `researcher2` can read, write, and access the contents of the `drafts` directory.

Summary

In this activity, I reviewed and managed Linux file permissions to comply with organizational policies. I used `ls -la` to inspect file permissions and `chmod` to modify them. I ensured no file was writable by unauthorized users and limited access to sensitive directories like `drafts`. This process demonstrated how Linux commands can be used to enforce security and proper authorization in a collaborative work environment.