Hands-on Lab: Access Control Commands

Estimated time needed: **10** minutes

Learning Objectives

After completing this lab, you will be able to:

- Understand the various access permissions a file or directory can have
- View the permissions for all files and directories within a directory
- Modify permissions for a file by user
- Describe the effect of changing permissions on a directory

About Skills Network Cloud IDE

Skills Network Cloud IDE (based on Theia and Docker) provides an environment for hands on labs for course and project related labs. Theia is an open source IDE (Integrated Development Environment), that can be run on desktop or on the cloud. To complete this lab, you will be using the Cloud IDE based on Theia.

Important notice about this lab environment

Please be aware that sessions for this lab environment are not persisted. Thus, every time you connect to this lab, a new environment is created for you and any data or files you may have saved in a previous session will be lost. To avoid losing your data, plan to complete these labs in a single session.

Exercise 1 - Viewing and modifying file access permissions

1.1 View file access permissions

Required files:

Run the following code to download the required files for this exercise:

```
1
2
```

```
cd /home/project
wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-LX0117EN-SkillsNetwork/labs/module%201/usdoi.txt
```

Copied!

Each file and each directory in your Linux system has permissions set for three permission categories: the 'user', the 'group', and 'all users' (or 'other').

The following permissions are set for each file and directory:

Permission	Symbol
read	`r`
write	`w`
execute	`x`

To see the permissions currently set for a file, run the `ls` command with the `-l` option.

For example, to see the permissions for the file named `usdoi.txt` in your current directory, enter the following:

```
1
```

```
1. ls -l usdoi.txt
```

Copied!

A sample output looks like the following:

```
-rw-r--r-- 1 theia theia 8121 May 31 16:45 usdoi.txt
```

The permissions set here are `rw-r--r--`. The `-` preceeding these permissions indicates that `usdoi.txt` is a file. If it were a directory, you would see a `d` instead of the `-`.

The first three entries correspond to the current user, the next three correspond to the group, and the last three are for all others. You can see the user has read and write permissions, while the user group only has read permission, and all other users have only read permission. No users have execute permission, as indicated by the `-` instead of an `x` in the third position for each user category.

1.2 Change file access permissions

`chmod`

The `chmod` or *change mode* command lets you change the permissions set for a file.

Specify which permissions to change with a combination of the following characters:

Option	Description
`r`, `w`, `x`	**Permissions**: read, write, and execute
`u`,`g`, `o`	**User categories**: user, group, and all others
`+`, `-`	**Operations**: grant and revoke

The following command *revokes* read permissions for *all users* (user, group, and other) on the file `usdoi.txt`:

```
1. 1
```

```
1. chmod -r usdoi.txt
```

Copied!

You can verify the changed permissions by entering:

```
1. 1
```

```
1. ls -l usdoi.txt
```

Copied!

To grant read access to *all users* on `usdoi.txt`, enter:

```
1. 1
```

```
1. chmod +r usdoi.txt
```

Copied!

Verify the changed permissions again with the following:

```
1. 1
```

```
1. ls -l usdoi.txt
```

Copied!

Now to remove the read permission only for 'other' category, enter the following:

```
1. 1
```

```
1. chmod o-r usdoi.txt
```

Copied!

Verify the changed permissions as follows:

```
1. 1
```

```
1. ls -l usdoi.txt
```

Copied!

Exercise 2 - Understanding directory access permissions

2.1 View default directory access permissions

Recall the following table, which illustrates the meanings of each permission for directories with examples of allowable operations for a given directory.

Directory Permission	Permissible action(s)
`r`	list directory contents using `ls` command
`w`	add/remove files or directories from directory
`x`	enter directory using `cd` command

For this exercise, first move to your `project` directory and create a new directory called `test`:

```
1. 1
2. 2
```

```
1. cd /home/project
2. mkdir test
```

Copied!

Check the default permissions that the system sets for your new `test` directory:

```
1. 1
```

```
1. ls -l
```

Copied!

As you can see from the resulting output:

```
1. 1
2. 2
3. 3
```

```
1. total 12
2. drwxr-sr-x 2 theia users 4096 May 15 14:06 test
3. -rw-r----- 1 theia users 8121 Sep 28  2022 usdoi.txt
```

Copied!

You, "theia", as the owner of `test`, have read, write, and execute permissions set by default. But all others only have read and execute permissions set and cannot write to your `test` directory. This means users outside your group can't add or remove files from `test`. They can, however, explore your directory to see what files and directories exist there.

> **Note:** You might be wondering what that `s` permission is in the execute slot for your group. The `s` stands for "*special* permission". It means that any new files created within the directory will have their group ownership set to be the same as the directory owner. We won't go into this level of detail in this course, but you can learn more about advanced Linux permissions here: Linux permissions: SUID, SGID, and sticky bit.

Go ahead and verify for yourself that you have permission to run the following commands. Change the directory to your `test` directory, create a new directory within it, then return to your parent directory:

```
1. 1
2. 2
3. 3
```

```
1. cd test
2. mkdir test2
3. cd ../
```

Copied!

2.2 Remove user execute permissions on your `test` directory

Remove your user execute permissions on `test` using the following command:

```
1. 1
```

```
1. chmod u-x test
```

Copied!

Now, what happens when you try to change directories to `test`?

```
1. 1
```

```
1. cd test
```

Copied!

You get an error message!

```
bash: cd: test: Permission denied
```

As you just removed execute permissions for yourself on your `test` directory, you can no longer make it your present working directory. However, you can still "read" it with the `ls` command:

```
1. 1
```

```
1. ls -l
```

Copied!

Even though you have "write" permissions set, you can't actually create a new directory within `test`, because removing execute permissions overrides write permissions. For example, entering,

```
1. 1
```

```
1. mkdir test/test3
```

Copied!

throws an error:

```
mkdir: cannot create directory ‘test/test’: Permission denied
```

This time, try restoring execute permissions on `test` and denying write permissions. Then verify your changes:

```
1. 1
2. 2
3. 3
```

```
1. chmod u+x test
2. chmod u-w test
3. ls -l
```

Copied!

Now you can go into `test`, but you still can't write to it! Entering

```
1. 1
2. 2
```

```
1. cd test
2. mkdir test_again
```

Copied!

throws the error:

```
mkdir: cannot create directory ‘test_again’: Permission denied
```

Practice exercises

1. List the permissions set for the file `usdoi.txt` that you downloaded to your project directory at the beginning of the lab.

▶ Click here for Hint
▶ Click here for Solution

2. Revoke the write permission on `usdoi.txt` for the user, and verify your result.

▶ Click here for Hint
▶ Click here for Solution

3. What happens if you try to delete `usdoi.txt` after revoking write permissions for the user?

▶ Click here for Hint
▶ Click here for Solution

4. Create a new directory called `tmp_dir` in your home directory.

▶ Click here for Hint
▶ Click here for Solution

5. View the permissions of the newly created directory, `tmp_dir`.

▶ Click here for Hint
▶ Click here for Solution

6. Revoke the user write permission for `tmp_dir`.

▶ Click here for Hint
▶ Click here for Solution

7. Check whether you can create a subdirectory of `tmp_dir` called `sub_dir`.

▶ Click here for Hint
▶ Click here for Solution 1
▶ Click here for Solution 2

Summary

In this lab, you learned that:

- Files and directories can have read, write, and execute permissions for the user, group, and others
- You can view the permissions for all files and directories within a directory by using `ls -l`
- You can modify permissions for a file by using `chmod`
- Changing permissions on a directory will change who can do what to the directory and its files

Authors

Ramesh Sannareddy
Sam Prokopchuk
Jeff Grossman

Other Contributors

Rav Ahuja