**1.1 View file access permissions**

**Required files:**

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

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

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. ls -l usdoi.txt

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. chmod -r usdoi.txt

You can verify the changed permissions by entering:

1. ls -l usdoi.txt

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

1. chmod +r usdoi.txt

Verify the changed permissions again with the following:

1. ls -l usdoi.txt

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

1. chmod o-r usdoi.txt

Verify the changed permissions as follows:

1. ls -l usdoi.txt

**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. cd /home/project
2. mkdir test

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

1. ls -l

As you can see from the resulting output:

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

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*](https://www.redhat.com/sysadmin/suid-sgid-sticky-bit#:~:text=Commonly%20noted%20as%20SUID%2C%20the,use%20an%20uppercase%20S%20here.)*.*

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

**2.2 Remove user execute permissions on your test directory**

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

1. chmod u-x test

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

1. cd test

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. ls -l

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. mkdir test/test3

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

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

1. cd test
2. mkdir test\_again

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.**

1. cd /home/project
2. ls -l usdoi.txt

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

1. 2
2. chmod u-w usdoi.txt
3. ls -l usdoi.txt

You should only see read permissions set for all three user categories.

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

If you run the following command:

1. rm usdoi.txt

You should see the following prompt:

1. rm: remove write-protected regular file \'usdoi.txt\'?

Entering y will override the write protection and delete the file, which you can verify as follows:

1. ls usdoi.txt

**4. Create a new directory called tmp\_dir in your home directory.**

1. mkdir tmp\_dir

**5. View the permissions of the newly created directory, tmp\_dir.**

1. ls -ld tmp\_dir

**6. Revoke the user write permission for tmp\_dir.**

1. chmod u-w tmp\_dir

**7. Check whether you can create a subdirectory of tmp\_dir called sub\_dir.**

1. cd tmp\_dir
2. mkdir sub\_dir

You should be able to enter the directory with the cd command, but you should see a "permission denied" error after entering the mkdir command.

1. mkdir tmp\_dir/sub\_dir

Again, you should see a "permission denied" error for the mkdir command.