Module III: Permission and Advance Permission

Part I

Basic file permissions in RHEL:

In Red Hat Enterprise Linux (RHEL), file permissions are controlled using the chmod command. The basic file permissions in RHEL are read (r), write (w), and execute (x) permissions for the file's owner, group, and others (other users on the system). These permissions can be represented numerically as well, with read being represented as 4, write as 2, and execute as 1. The permissions for a file can be viewed using the ls -l command.

You can use chmod command to change the permissions of a file or directory. For example:

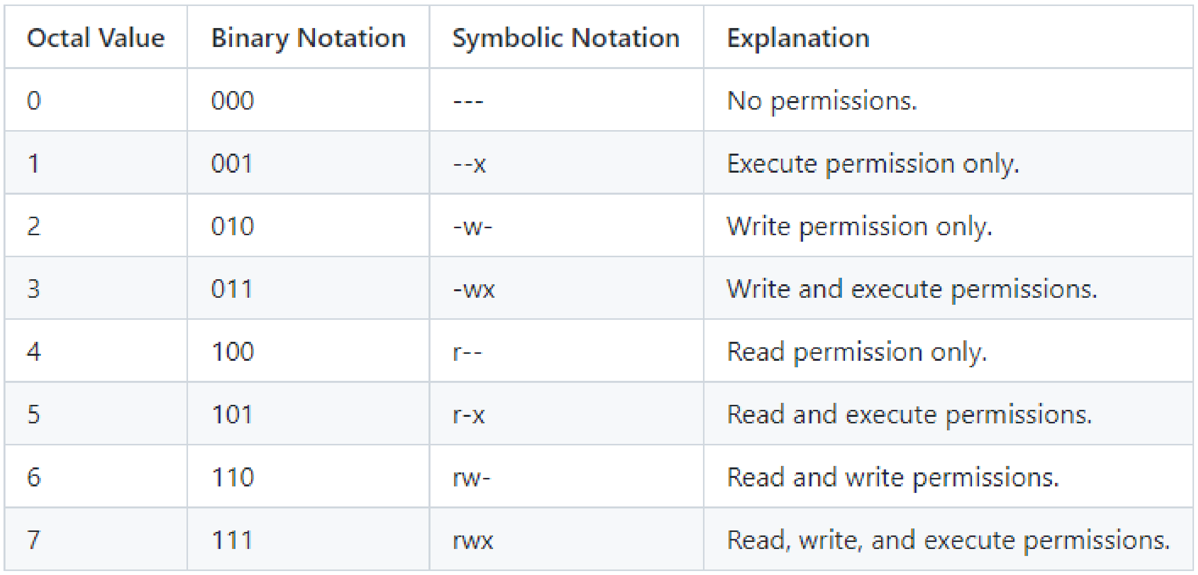
- chmod 755 file.txt
- chmod +x file.txt

The first command sets read, write, and execute permissions for the owner and read and execute permissions for the group and others. The second command adds execute permission to the file for the owner, group and others.

File Permissions and Ownership:

File permissions in Linux determine who can read, write, and execute a file. File ownership determines who owns a file and who has the ability to change the file's permissions.

File permissions are controlled using the chmod command and are represented by a combination of letters (rwx) or numbers (rwx = 7, rw = 6, rx = 5, etc.). The permissions can be set for the file's owner, the file's group, and others (other users on the system).



Octal Value	Binary Notation	Symbolic Notation	Explanation
0	000	---	No permissions.
1	001	--x	Execute permission only.
2	010	-w-	Write permission only.
3	011	-wx	Write and execute permissions.
4	100	r--	Read permission only.
5	101	r-x	Read and execute permissions.
6	110	rw-	Read and write permissions.
7	111	rwx	Read, write, and execute permissions.

To set the file permissions using UGO concepts, you can use the chmod command. For example:

- chmod 755 file.txt
- chmod u+rwx,g+rx,o+r file.txt

The first command sets read, write, and execute permissions for the owner and read and execute permissions for the group and others. The second command give read, write, and execute permissions to the owner, read and execute permissions to the group, and read permission to others. It's important to note that the owner of a file has the ability to change the file's permissions, but the group and others can only change the file's permissions if the write and execute permissions are set for the group or others.

File ownership is controlled using the chown command and is represented by a combination of a user and a group. The user and group can be specified by their name or by their numerical ID. The file's owner can change the file's permissions, but the group and others can only change the file's permissions if the write and execute permissions are set for the group or others.

It's important to note that in Linux, permissions and ownership are separate but related concepts. File permissions determine who can access a file, while file ownership determines who can change the file's permissions. It's also important to note that in some cases specific users or groups can be given special permissions like the SUID and SGID.

Examples of chmod command in rhel:

The chmod command is used to change the file permissions in Red Hat Enterprise Linux (RHEL). Here are some examples of how to use the chmod command:

1. To give read, write, and execute permissions to the owner, read and execute permissions to the group, and read permissions to others for a file named file.txt:

   chmod 755 file.txt

2. To add execute permission for the owner, group, and others for a file named file.txt:

   chmod +x file.txt

3. To remove write permission for the group and others on a file named file.txt:

   chmod g-w,o-w file.txt

4. To give read and execute permissions to the owner, and read permission to group and others for all the files in a directory named mydir:

   chmod -R 744 mydir

It's important to note that when using chmod command, you can use numeric values or the symbolic representation (rwx) or a combination of both. Also, the chown command can be used in combination with chmod command to change both ownership and permissions of the file.

The chown command is used to change the ownership of a file or directory in Red Hat Enterprise Linux (RHEL). Here are some examples of how to use the chown command:

1. To change the ownership of a file named file.txt to a user named user1:

   chown user1 file.txt

2. To change the ownership of a file named file.txt to a user named user1 and a group named group1:

   chown user1:group1 file.txt

3. To change the ownership of all the files and directories in a directory named mydir to a user named user1 and a group named group1:

   chown -R user1:group1 mydir

4. To change the ownership of a file named file.txt to the current user:

   ```
   chown . file.txt
   ```

5. To change the ownership of a file named file.txt to the current user and group:

   ```
   chown .:. file.txt
   ```

It's important to note that when using chown command, you need to have appropriate permissions to execute the command. Also, the chown command can be used in combination with chmod command to change both ownership and permissions of the file.

It's also important to note that changing the ownership of a file or a directory can have security implications, so it's important to be mindful of the permissions and ownership of files and directories on your system, and to use the chown command with caution.

The chgrp command in RHEL (Red Hat Enterprise Linux) is used to change the group ownership of a file or directory. Here are some examples of how the chgrp command can be used:

1. Change group ownership for a single file:

   ```
   $ chgrp groupname filename
   ```

2. Change group ownership for multiple files:

   ```
   $ chgrp groupname file1 file2 file3
   ```

3. Change group ownership for all files in a directory:

   ```
   $ chgrp -R groupname directory
   ```

In these examples, **groupname** is the name of the group to which you want to change the ownership, and **filename**, **file1**, **file2**, and **file3** are the names of the files you want to modify. The **-R** option tells the **chgrp** command to change the group ownership recursively, so all files and directories within the specified directory are also affected.

Note that you need to be the owner of the file or have sufficient privileges to change the group ownership of a file or directory.

The Loophole in Write Permissions:

It's easy to remove write permissions from a file. For example, if you wanted to make the license.txt file "read-only", the following command removes write permissions from that file:

```
$ chmod a-w license.txt
```

But the user that owns the file can still make changes. It won't work in GUI text editors such as gedit. It won't even work in the nano text editor. However, if a change is made in the vi text editor, the user who owns that file can override a lack of write permissions with the bang character, which looks like an exclamation point (!). In other words, while in the vi editor, the user who owns the file can run the following command to override the lack of write permissions:

```
!w
```