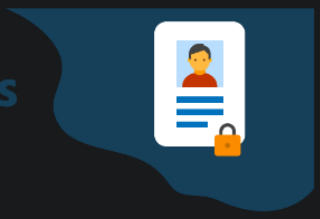

Permissions

Permissions determine the ways different users can interact with a file or directory. When listing a file with the `ls -l` command, the output includes permission information. For the example we will use a script called `hello.sh` located in the `Documents` directory:

Follow Along

Use the following command to switch to the `Documents` directory:

```
sysadmin@localhost:~$ cd ~/Documents
```

```
sysadmin@localhost:~/Documents$ ls -l hello.sh
-rw-r--r-- 1 sysadmin sysadmin 647 Dec 20  2017 hello.sh
```

Below is a review of the fields relevant to permissions.

File Type Field

```
- rw-r--r-- 1 sysadmin sysadmin 647 Dec 20  2017 hello.sh
```

The first character of this output indicates the type of a file. Recall that if the first character is a `-`, this is a regular file. If the character was a `d`, it would be a directory.

Permissions Field

```
- rw-r--r-- 1 sysadmin sysadmin 647 Dec 20  2017 hello.sh
```

After the file type character, the permissions are displayed. The permissions are broken into three sets of three characters:

Owner

```
- rw- r--r-- 1 sysadmin sysadmin 647 Dec 20  2017 hello.sh
```

The first set is for the user who owns the file. If your current account is the user owner of the file, then the first set of the three permissions will apply and the other permissions have no effect.

The user who owns the file, and who these permissions apply to, can be determined by the *user owner* field:

```
-rw-r--r-- 1 sysadmin sysadmin 647 Dec 20  2017 hello.sh
```

Group

```
-rw- r-- r-- 1 sysadmin sysadmin 647 Dec 20  2017 hello.sh
```

The second set is for the group that owns the file. If your current account *is not* the user owner of the file and you *are* a member of the group that owns the file, then the group permissions will apply and the other permissions have no effect.

The group for this file can be determined by the *group owner* field:

```
-rw-r--r-- 1 sysadmin sysadmin 647 Dec 20  2017 hello.sh
```

Other

```
-rw-r-- r-- 1 sysadmin sysadmin 647 Dec 20  2017 hello.sh
```

The last set is for everyone else, any one who that first two sets of permissions do not apply to. If you are not the user who owns the file or a member of the group that owns the file, the third set of permissions applies to you.

Permission Types

There are three different permissions that can be placed on a file or directory: read,

write, and execute. The manner in which these permissions apply differs for files and directories, as shown in the chart below:

Permission	Effects on File	Effects on Directory
read (`r`)	Allows for file contents to be read or copied.	Without execute permission on the directory, allows for a non-detailed listing of files. With execute permission, `ls -l` can provide a detailed listing.
write (`w`)	Allows for contents to be modified or overwritten. Allows for files to be added or removed from a directory.	For this permission to work, the directory must also have execute permission.
execute (`x`)	Allows for a file to be run as a process, although script files require read permission, as well.	Allows a user to change to the directory if parent directories have execute permission as well.

Consider This

Understanding which permissions apply is an important skill set in Linux. For example, consider the following set of permissions:

```
-r--rw-rwx. 1 sysadmin staff 999 Apr  10  2013 /home/sysadmin/
```

In this scenario, the user `sysadmin` ends up having less access to this file than members of the `staff` group or everyone else. The user `sysadmin` only has the permissions of `r--` . It doesn't matter if `sysadmin` is a member of the `staff` group; once user ownership has been established, only the user owner's permissions apply.

Previous Next