

ULI101: INTRODUCTION TO UNIX / LINUX AND THE INTERNET

WEEK 4: LESSON 2

FILE PERMISSIONS

PHOTOS AND ICONS USED IN THIS SLIDE SHOW ARE LICENSED UNDER [CC BY-SA](#)

LESSON 2 TOPICS

File Permissions

- Purpose
- **Directory** vs. **Regular File** Permissions
- Changing File Permissions (**chmod**)
- Setting File Permissions for Newly Created Directories and Regular Files (**umask**)
- Demonstration

Perform **Week 4** Tutorial

- Investigation 2
- Review Questions (Questions 6 – 12)

Complete **Assignment 1**

Work on **Assignment 2 (Section 2: Permissions)**

FILE PERMISSIONS

```
drwxr-xr-x 2 murray.saul users 6 Jan 19 14:06 mydir
-rw-r--r-- 1 murray.saul users 0 Jan 19 14:05 myregfile
```

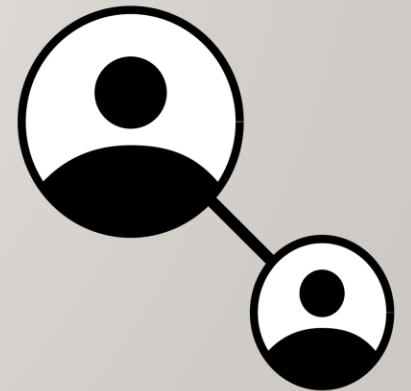
File Permissions

Since Unix / Linux operating systems allow for **multiple user accounts**, it is essential to have a system to **share** or **limit** access to directories and files contained in those file systems.

When **directories** and **regular files** are created, they are assigned to an **owner** (typically the username of the creator).

To *allow* or *limit* **access** to those files and directories, those files and directories are assigned to an initial **group** referred to as a "**primary group**".

Users that own those *directories* and *regular files* are referred to as **users**, users that belong within that same primary group are referred to as **same group members**, and those users that do NOT belong to a particular group are referred to as **other group members**.



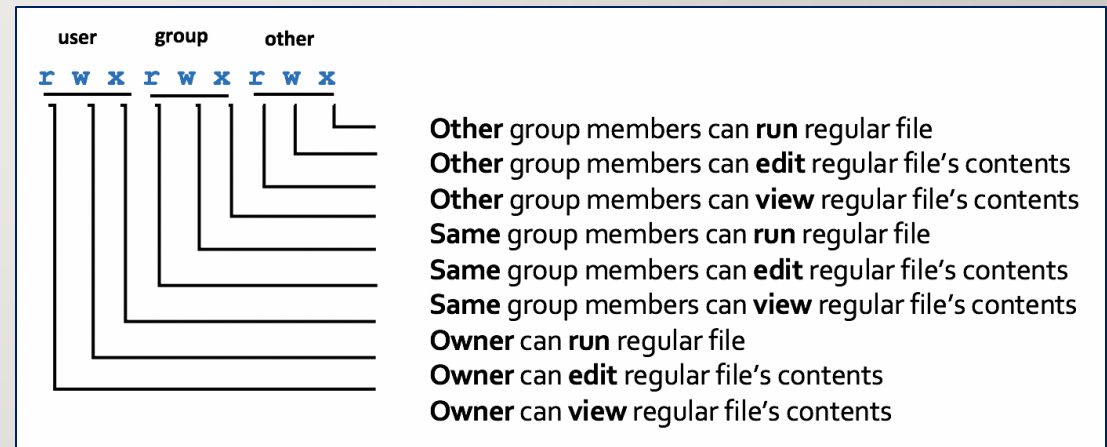
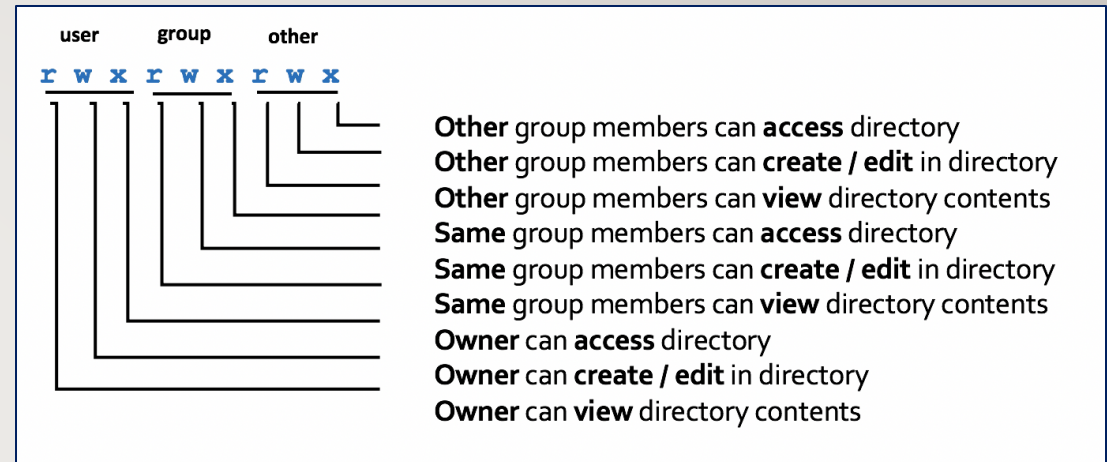
FILE PERMISSIONS

File permissions consist of **two-layers**:

- **First**, the permissions relating to a **directory**.
- **Second**, the permissions relating to the **regular files** contained within a directory.

NOTE: Permissions for **directories** have a different meaning than permissions for **regular files**.

NOTE: A symbol *dash* "-" indicates that the permission is **NOT** granted.



FILE PERMISSIONS

Changing File Permissions with `chmod` command - *Symbolic Method*:

The `chmod` command can use **symbols** to **add, remove,** and **set `rx`** permissions for **user, same group members, other group members** or **ALL** categories:

NOTE: You can use the **-R** option to set permissions for directory, subdirectory and directory contents **recursively**.

Command	Description
<code>chmod ugo+x script.bash</code>	Add execute permissions to the file script.bash so it can be run.
<code>chmod u=rwx,go=x ~</code>	Set " pass-thru " permissions of your home directory for same group members and other group members to navigate to other subdirectories (that may have access / view permissions).
<code>chmod go-w ~/shared</code>	Remove write permissions for same group members and other group members for the directory ~/shared
<code>chmod a=rx myfile.txt</code>	Set read and execute permissions for the directory myfile.txt

FILE PERMISSIONS

Instructor Demonstration

Your instructor will now demonstrate how to **add, remove** and **set** permissions with the **chmod** command the *Symbolic* method



FILE PERMISSIONS

Changing File Permissions with `chmod` command - *Absolute (Octal) Method:*

You can also use **octal numbers** to **set** permissions.

This method is a shortcut and may require less typing than using the *symbolic* method.

- **First**, write **permissions** for user, group and others that you want to set. **If permission is granted, write 1 and if not granted, write 0.**
- **Second**, perform a **binary to octal conversion**, for each group of three (user, group, other) and then issue the **chmod** command using the absolute (octal) method.

You can only use this method to **set** file permissions (as opposed to *adding* or *removing* permissions).

<u>r</u>	<u>w</u>	<u>x</u>	<u>r</u>	<u>-</u>	<u>x</u>	<u>-</u>	<u>-</u>	<u>x</u>
1	1	1	1	0	1	0	0	1
(4)	(2)	(1)	(4)	(2)	(1)	(4)	(2)	(1)
7	5	1						

FILE PERMISSIONS

Changing File Permissions with `chmod` command: *Absolute (Octal) Method*

Below is a table that displays common `chmod` commands (using the Absolute / Octal method) for common purposes.

Command	Description
<code>chmod 500 script.bash</code>	Set read and execute permissions for only the user for the file script.bash so it can be run.
<code>chmod 711 ~</code>	Set " pass-thru " permissions of your home directory.
<code>chmod 750 ~/shared</code>	Set full permissions for user, read and access permissions for some group members and no permissions for other group members for the directory ~/shared
<code>chmod 555 myfile.txt</code>	Set read and execute permissions for the directory myfile.txt

FILE PERMISSIONS

Instructor Demonstration

Your instructor will now demonstrate how to **set** permissions with the **chmod** command using the *Absolute / Octal* method.



FILE PERMISSIONS

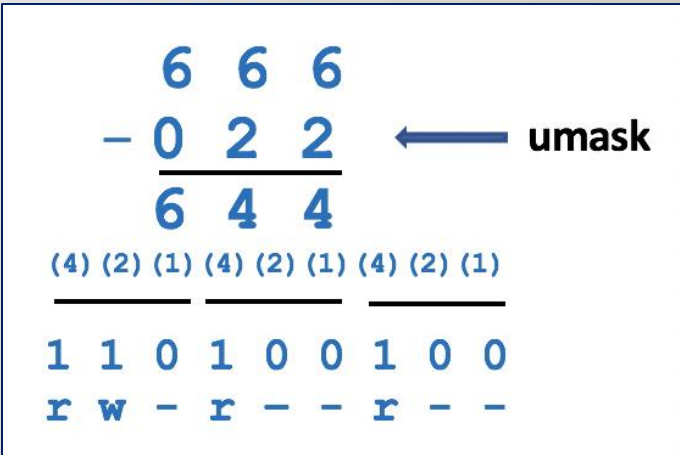
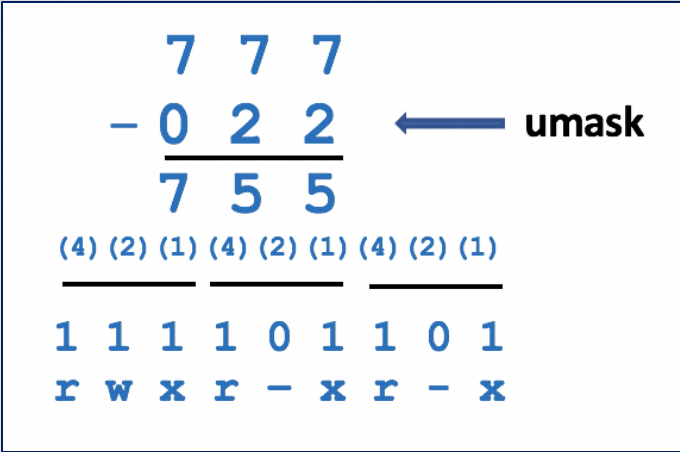
Setting Permissions for Newly-Created Directories and Regular Files (umask):

The `umask` command is used to set the permissions of **newly-created directories and regular files**. Issuing the `umask` command without arguments will display the current umask value.

The diagram on the above right shows how to calculate permissions for newly-created **directories** using the `umask` command.

The diagram on the below right shows how to calculate permissions for newly-created **regular files** using the `umask` command.

Setting the `umask` value works only in the current shell session unless the `umask` command is contained in a start-up file (e.g. `.profile`, `.bash_profile`, or `.bashrc`). Start-up files are discussed at the end of this course.



FILE PERMISSIONS

Instructor Demonstration

Your instructor will now demonstrate how to **set / confirm** permissions of newly-created directories and regular files using the **umask** command.



FILE PERMISSIONS

Getting Practice

To get practice to help perform **Assignment #2**, perform the online tutorial **Tutorial2: Unix / Linux File Management** (**ctrl-click** to open link):

- [INVESTIGATION 2: FILE PERMISSIONS](#)
- [LINUX PRACTICE QUESTIONS](#) (Questions 6 – 12)

Complete Assignment 1

Work on Assignment 2:

- **Section 2: Permissions**