

**Fundamentals of UNIX**  
**Lab 10.3.3 – Changing Permissions from the Command Line**  
**(Estimated time: 45 min.)**

**Objectives:**

- Work with file system to control security access
- Review **chmod** command modes
- Change file permissions using symbolic mode
- Change directory permissions using symbolic mode
- Determine octal mode permissions
- Change file permissions using octal mode
- Change directory permissions using octal mode
- Create a script file using the vi editor and make it executable

**Background:**

In this lab, you will analyze and change UNIX file system security permissions using command line utilities. File and directory permissions can be changed using the **chmod** (change mode) command. Normally the default permissions for a file or directory will be adequate for most security needs. There are times when you will want to change the permissions on a file or directory. By default, all files are created with permissions that allow the user category of **others** to read the file. This means anyone with a login id can see the contents of the file and copy it. For classified files and private information, you can modify the permission of the file to prevent **others** from accessing it.

Shell scripts are another example where you would want to change permissions. When you create a shell script file (or any file), the default permissions do not include execute, even for the owner/creator of the file. To run the shell script, you must change the permissions by adding the execute permission for the user (owner) category.

**Tools / Preparation:**

- a) Before starting this lab, review Chapter 10, Section 3 – Changing Permissions from the Command Line
- b) You will need the following:
  - 1. A login user ID (e.g. user2) and password assigned by your instructor.
  - 2. A computer running the UNIX operating system
  - 3. Networked computers in classroom with class file system installed

**Notes:**

---

---

---

---

---

---

---

---

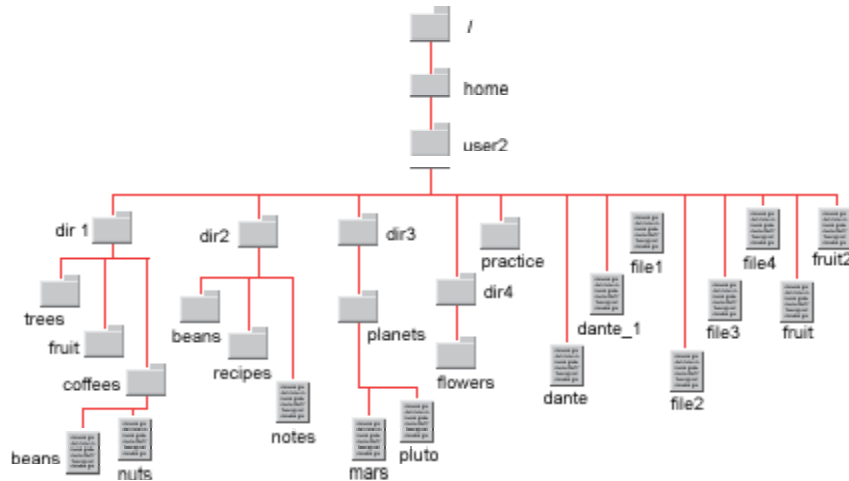
---

---

Fundamentals of UNIX  
Lab 10.3.3 – Changing Permissions from the Command Line  
Worksheet

Use the diagram of the sample Class File System directory tree to assist with this lab.

**Class File Tree Structure**



**Step 1. Log in to CDE**

Login with the user name and password assigned to you by your instructor in the CDE entry box.

**Step 2. Access the Command Line**

Right click on the **workspace** backdrop and click on **Tools**. Select **Terminal** from the menu to open a terminal window.

**Step 3. Review `chmod` Command Modes**

The `chmod` (change mode) command is used by a file's owner (or superuser) to change file permissions. The two **modes** of operation with the `chmod` command are **symbolic** (or *relative*) and **octal** (or *absolute*). The general format of the `chmod` command is shown below. The mode portion will change depending on whether you are using symbolic or octal mode.

**Command format:**            `chmod`            *mode*            *filename*

**Symbolic mode** - uses combinations of letters and symbols to add or remove permissions from various categories of users. Symbolic mode is also referred to as relative mode.

**Octal mode** - uses numbers to represent file permissions. Octal mode is also referred to as absolute or numeric mode.

- Which `chmod` mode uses **numbers** to represent file permissions? \_\_\_\_\_
- Which `chmod` mode uses **letters** (or symbols) to represent permissions? \_\_\_\_\_
- What is another term for **Octal** mode? \_\_\_\_\_
- What is another term for **Symbolic** mode? \_\_\_\_\_

**Fundamentals of UNIX**  
**Lab 10.3.3 – Changing Permissions from the Command Line**  
**Worksheet – Cont.**

**Step 4. Change File Permissions Using Symbolic Mode**

When using symbolic mode to set permission, you typically work with one category of users, although you can give all categories the same permissions simultaneously. The mode is referred to as **relative** since you are assigning or removing permissions relative to the ones that are already there. You can add one or more permissions to a specific category of users or take them away. The command format for symbolic mode uses letters and symbols.

**The mode portion of the command format is made up of three parts:**

**Who** - Category of users you are working with: **u** = user, **g** = group, **o** = other or **a** = all

**Op** - Operator or what you are going to do: set (**=**), remove (**-**), give (**+**)

**Permissions** - Permission(s) to be assigned – **r** = read, **w** = write or **x** = execute

The following example **removes** (-) the **read** permission (r) from the file **dante** for the **other** (o) category of users. **Note:** There should be NO spaces between the o, dash (-) and r.

```
chmod o-r dante
```

The next example **adds** (+) the **write** permission (w) to the file **dante** for the **group** (g) and **other** (o) categories of users.

```
chmod g o + w dante
```

a. From your **home** directory, create a new directory under your practice directory called **chmoddir** using a relative pathname. What command did you use to create the directory?

\_\_\_\_\_

b. Change to the **chmoddir** directory and create a new file called **symfile**. What command did you use to create the file? \_\_\_\_\_

c. Use the **ls -l** command to determine the permissions for the new **symfile** file. These are the default permission for a file. What are the permissions for **User**, **Group** and **Other**?

\_\_\_\_\_

d. You decide you do not want **other** users (other than yourself and members of your group) to be able to see the contents of or copy **symfile**. Use the **chmod** command, in **symbolic** mode, to **remove** the r (read) permission for **other** users for the file **symfile**. What command did you use?

\_\_\_\_\_

e. List the permission of the file again. What are the permission for the **others** user category now?

\_\_\_\_\_

f. What command would you use if you wanted to remove the read permission for **both** the **group** and **others** with a single command? \_\_\_\_\_

**Fundamentals of UNIX**  
**Lab 10.3.3 – Changing Permissions from the Command Line**  
**Worksheet – Cont.**

**Step 5. Change Directory Permissions Using Symbolic Mode**

- a. Change back to your **practice** directory. What command did you use?  
\_\_\_\_\_
- b. From your **practice** directory, list the permissions for the new **chmoddir** directory you created earlier. These are the default permissions for a directory. What are the permissions for **User**, **Group** and **Other**? \_\_\_\_\_
- c. Can users **other** than yourself or members of your group (staff) copy files from your **chmoddir** directory? \_\_\_\_\_ Why or why not? \_\_\_\_\_
- d. You do not want **other** users to be able to copy files from your **chmoddir** directory. Change to your **practice** directory and use the **chmod** command in **symbolic** mode to **remove** the **read** permission and the **execute** permission for the **others** category of users from the directory **chmoddir**. What command did you use? \_\_\_\_\_
- e. List the permissions of the directory again. What are the permission for the **others** user category now? \_\_\_\_\_
- f. Can the members of your primary **group** (staff) create new files in or copy files into your **chmoddir** directory? \_\_\_\_\_ Why or why not? \_\_\_\_\_
- g. Change to your **practice** directory and use the **chmod** command in **symbolic** mode to **add** the **write** permission for your primary **group** for the directory **chmoddir**. What command did you use?  
\_\_\_\_\_
- h. Change the permission back to the default permissions using symbolic mode. What command(s) did you use? (**Hint**: groups and permissions can be combined with one command or you can use two separate commands.) \_\_\_\_\_

**Step 6. Determine Octal Mode Permissions**

**Octal** mode provides a quick **numeric** means of changing permissions for all categories of users simultaneously while still allowing each set of permissions to be different. There are three possible permissions for each set (r, w, and x) for each type of user category (user, group, other). Each set of permissions can be assigned a numeric value (from 0 to 7) depending on which permissions are allowed.

The r (read) permission is assigned a value of 4, the w (write) permission a value of 2 and the x (execute) permission a value of 1. By adding up the numbers we can get a total of all three permissions for that category of user (User, Group or Other). For instance if the **Owner (user)** permission for a file is **r w x**, We add 4 (read) + 2 ( write) + 1 (execute) which equals **7**. If the **group** had **r w -** permissions they would have 4 + 2 + 0 (no execute) for a total of **6**. If **other** had only **r** they would have 4 + 0 + 0 (no write or execute) for a total of **4**. The octal\_mode for this file or directory is **764**.

7	6	4
4+2+1	4+2+0	4+0+0
r w x	r w -	r - -
User	Group	Other

**Fundamentals of UNIX**  
**Lab 10.3.3 – Changing Permissions from the Command Line**  
**Worksheet – Cont.**

**Step 6. Determine Octal Mode Permissions – Cont.**

a. Fill in the following table by converting the character permissions (r,w,x, -) to their **octal** equivalents. Convert each set of permissions first (User, Group or Other) and then enter the octal\_mode (three digit number) under Octal Mode permissions.

User Permissions	Octal Sum	Group Permissions	Octal Sum	Other Permissions	Octal Sum	Octal Mode Permissions
r w x		r w -		r w -		
r w -		r - -		r - -		
r - -		r - -		r - -		
r w x		r - x		r - x		

**Step 7. Change File Permissions Using Octal Mode**

With octal mode, it is not necessary to specify the category of users since the position of each number represents one of the three user categories. The *octal\_mode* is made up of three numbers, each of which is the sum for one of the user categories (User, Group, and Other). Octal values are combined to identify the octal\_mode that is used with the **chmod** command.

**Command Format:**    **chmod   octal\_mode   filename**

a. Change to the **chmoddir** directory and create a new file called **octfile**. What command did you use to create the file? \_\_\_\_\_

b. Use the **ls -l** command to determine the permissions for the new **octfile** file. These are the default permission for a file. What are the **alphanumeric** permissions for **User**, **Group** and **Other**?  
\_\_\_\_\_

c. What is the **octal** mode equivalent of the user, group, and other permission for this file.  
\_\_\_\_\_

d. You decide you do not want **other** users to be able to see the contents of or copy **octfile**. Use the **chmod** command in **octal** mode to **remove** the r (read) permission for **other** users for the file **octfile**. What command did you use? \_\_\_\_\_

e. List the permission of the file again. What are the permission for the **others** user category now?  
\_\_\_\_\_

f. What command would you use if you wanted to **remove all** permissions for **both** the **group** and **others** with a single command? \_\_\_\_\_

**Step 8. Change Directory Permissions Using Octal Mode**

The format below is used to change the permissions on a directory. The **-R** (recursive) option changes the permissions on the specified directory and on **all subdirectories and files** within it.

**Command Format:**    **chmod [-R]   octal\_mode   directoryname**

a. Change to your **practice** directory. What command did you use?  
\_\_\_\_\_

**Fundamentals of UNIX**  
**Lab 10.3.3 – Changing Permissions from the Command Line**  
**Worksheet – Cont.**

b. From your **practice** directory, list the permissions for the **chmoddir** directory. These are the default permissions for a directory. What are the **alphanumeric** permission for **User**, **Group** and **Other**? \_\_\_\_\_

c. What is the **octal** mode equivalent of the user, group and other permission for this directory?  
\_\_\_\_\_

d. Use the **chmod** command in **octal** mode to **remove** the **read** and the **execute** permission for the **others** category of users from the directory **chmoddir**. What command did you use? (Remember, you must always specify all 3 sets of permissions with octal mode even if they are not to be changed.)  
\_\_\_\_\_

e. List the permissions of the directory again. What are the permissions for the **others** user category now? \_\_\_\_\_ Did the permissions remain the same for the user and group? \_\_\_\_\_

f. Can the members of your primary **group** (staff) create new files in or copy files into your **chmoddir** directory? \_\_\_\_\_ Why or why not? \_\_\_\_\_

g. You decide you want members of your **group** to be able to copy files to your directory. Change to your **practice** directory and use the **chmod** command in **octal** mode to **add** the **write** permission for your primary **group** (staff) for the directory **chmoddir**. The **user** should have **rw**, the **group** should have **rw** and **other** should have **no permissions** to the directory. What command did you use?  
\_\_\_\_\_

h. Change the permissions back to the **default** permissions (**rxr-r-x**) using **octal** mode.

**Step 9. Create a Script File and Make it Executable**

In this step, you will create a simple text script file using the vi editor. You will then need to make it executable in order to run (execute) the script file. Script files can be very useful to help automate repetitive tasks.

a. Change to the **chmoddir** directory and start the vi editor using **vedit**. As you start the editor, specify or open a new file called **myscript**. Press **i** to go into Insert Entry mode and type the following commands as lower case text. Press **Enter** after each one.

```
clear
pwd
ls -l
banner "my script"
```

b. Press **Esc** to return to command mode and then type a colon to get to **last-line** mode. Press **wq** to **write** (save) the file and **quit** vi.

c. List the file to determine its permissions. What are they? \_\_\_\_\_  
\_\_\_\_\_

d. Type **myscript** as though it were a command and press **Enter**. What was the response?  
\_\_\_\_\_ Why did it not execute? \_\_\_\_\_

**Fundamentals of UNIX**  
**Lab 10.3.3 – Changing Permissions from the Command Line**  
**Worksheet – Cont.**

e. Change the permissions for the **myscript** file so that the **user** permissions include **x** (execute) so that you as the owner can execute or run the file. You can use either **symbolic** or **octal** mode. What command did you use to change the permissions? \_\_\_\_\_

\_\_\_\_\_

f. List the file to verify that the permissions changed. What are the permissions for the user (owner) now? \_\_\_\_\_

f. Type **myscript** as a command again and press **Enter**. What was the response?

\_\_\_\_\_

**Step 10 – Remove Files and Directories Created in this Lab**

**Remove all files** and **directories** created in your home directory during this lab.

**Step 11. Close the Terminal Window and Logout**

Double click on the dash button in the upper left corner of the screen, then click the **EXIT** icon on the front panel.