**COMP 2766: Introduction to Linux**

**Assignment 8: File Ownership, Default Permissions, and Special Permissions**

**INSTRUCTIONS**

***\*\*\* READ THIS \*\*\*:*** ***If a step in this lab requests a screenshot, insert it in the space provided. If your command line generates no output, your screenshot MUST show the cursor on the following line. Failure to do this will require you to redo that step and provide a correct screenshot.*** ***NO SCREENSHOT, NO MARKS***. ***You may only use commands shown in the NDG course or in the lecture slides. Do NOT search for an answer on the Internet.***

**PREPARATION FOR THE LAB**

1. **On your EduCloud CentOS 9 VM**, login user ***root*** (**do *NOT* use the su command!**). **You *MUST* do this on your EduCloud CentOS 9 VM**.
2. open a new terminal window
3. create group comp2766 and these new users, one of which is based on your first name:
   * # groupadd comp2766
   * # useradd -G comp2766 elton
   * # passwd elton (*use abc as the password and ignore the warning*)
   * # useradd cheryl
   * # passwd cheryl (*use abc as the password and ignore the warning*)

If you already have a user on your VM named after your first name, enter:

* + # usermod -G comp2766 *yourFirstName*
  + # passwd *yourFirstName* (*use abc as the password and ignore the warning*)

If you do **NOT** currently have a user on your VM named after your first name, enter:

* + # useradd -G comp2766 *yourFirstName*
  + # passwd *yourFirstName* (*use abc as the password and ignore the warning*)

1. in your terminal window, open a new tab
2. using your actual first name, enter this: # su – *yourFirstName*
3. in your terminal window, open another tab
4. in that new tab, enter: # su – elton
5. in your terminal window, open another tab
6. in that new tab, enter: # su – cheryl
7. confirm that you have four tabs open with users root, *yourFirstName*, elton, and cheryl in each
8. as you do this lab, recall that *~username* refers to a user’s home directory (ex: ~elton means user elton’s home directory)

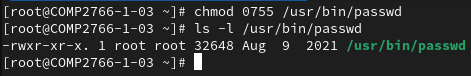
**THE SUID (a.k.a. set UID) PERMISSION BIT**

The SUID permission bit is set only on regular executable files such as programs, commands, and scripts. It has no effect when set on directory files. When set on a regular executable file, that file is ***run as*** the user who owns that file instead of the user who executed that file. This is necessary when the executable file (like the passwd command) must access or modify the contents of another file (like /etc/shadow) that a non-root user has no access to.

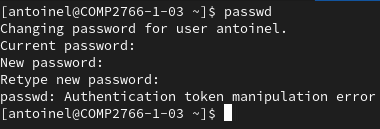
1. In your terminal window, click the tab for ***root***
2. (**1 mark**): As user *root*, enter a command line that outputs a long listing of the passwd command file:

# ls -l /usr/bin/passwd

Notice who the user owner is. The presence of a lowercase s in the user owner permissions section indicates that the SUID bit is set and that there is a lowercase x permission underneath. To verify that, as user *root*, enter a command line that removes only the SUID bit without changing any other permissions. ***Insert a screenshot*** showing the command line that you entered and its output, here:

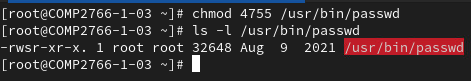


1. As user *root*, enter a command line that outputs a long listing of /usr/bin/passwd . Verify that the lowercase s has disappeared and that the permissions are now rw**x**r-xr-x. If they are different, then you must reverse the changes you made in the previous step and do it properly.
2. In your terminal window, click the tab for ***yourFirstName***.
3. (**1 mark**): As user ***yourFirstName***, enter the passwd command (without an argument) to change your password. You **MUST** enter a sufficiently complex password, so use #27-901Willingdon, but it should still fail. ***Insert a screenshot*** showing your attempt to change your password and the result, here:

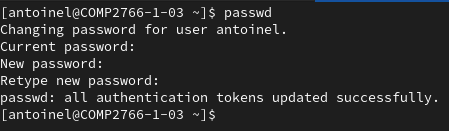


Your attempt to change your password failed, because the passwd command must update /etc/shadow with a new SALT and a new encrypted password hash. However, the passwd command was executed by you and you have no access to /etc/shadow. To change your password, you and other non-root users must execute the passwd command as the root user -- who happens to be the user owner of /usr/bin/passwd

1. In your terminal window, click the tab for ***root***
2. (**1 mark**) As user *root*, enter a command line that sets only the SUID bit without changing any other permissions. ***Insert a screenshot*** showing the command line that you entered and its output, here:



1. As user *root*, enter a command line that outputs a long listing of /usr/bin/passwd . Verify that the lowercase s has reappeared and that the permissions are now rw**s**r-xr-x (meaning that the SUID bit has been set). If they are different, then you must reverse the changes you made in the previous step and do it properly.
2. In your terminal window, click the tab for ***yourFirstName***.
3. (**1 mark**): As user ***yourFirstName***, enter the passwd command (without an argument) to change your own password. You **MUST** enter a sufficiently complex password, so use #4B-765CanadaWay. ***Insert a screenshot*** showing your attempt to change your password and the result, here:



You can now change your password, because the SUID bit is set. When you executed /usr/bin/passwd, it was executed as the user owner which is root who can update /etc/shadow with your new SALT and new encrypted password hash.

1. In your terminal window, click the tab for ***root***
2. As user *root*, enter a command line that assigns gedit to the environment variable named EDITOR. Remember that you **MUST** export an environment variable. Refer to the NDG course for help.
3. As user *root*, enter the visudo command and add this line immediately below line 100 using your first name:

*yourFirstName* ALL=/usr/bin/tail /var/log/messages

1. Click Save and close gedit.
2. In your terminal window, click the tab for ***yourFirstName***.
3. As user ***yourFirstName***, enter this command line which should fail:

$ tail /var/log/messages

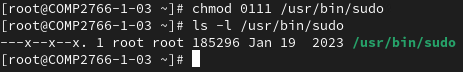
1. As user ***yourFirstName***, enter this command line by filling in the blank with the appropriate command that you learned in the NDG course to allow you to tail the messages log file:

$ sudo tail /var/log/messages

1. In your terminal window, click the tab for ***root***
2. (**1 mark**): As user *root*, enter this command line to output a long listing of the sudo command file:

# ls -l /usr/bin/sudo

Notice who the user owner is. The presence of a lowercase s in the user owner permissions section indicates that the SUID bit is set and that there is a lowercase x permission underneath. Enter a command line that removes only the SUID bit on /usr/bin/sudo without changing any other permissions. ***Insert a screenshot*** showing the command line that you entered and its output, here:



1. As user *root*, enter a command line that outputs a long listing of /usr/bin/sudo. Verify that the lowercase s has disappeared and that the permissions are now --**x**--x--x. If they are different, then you must reverse the changes you made in the previous step and do it properly.
2. In your terminal window, click the tab for ***yourFirstName***.
3. (**1 mark**): As user ***yourFirstName***, re-enter this command line that you entered in step #28 by filling in the blank with the appropriate command that you learned in the NDG course to allow you to tail the messages log file:

$ sudo tail /var/log/messages

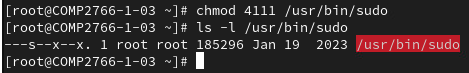
***Insert a screenshot*** showing the command line that you entered and its output, here:



This time, the command line fails because the SUID bit is no longer set on /usr/bin/sudo. That means when your user runs the sudo command, your user has no access to /etc/sudoers which contains this rule that you added in a previous step:

*yourFirstName* ALL=/usr/bin/tail /var/log/messages

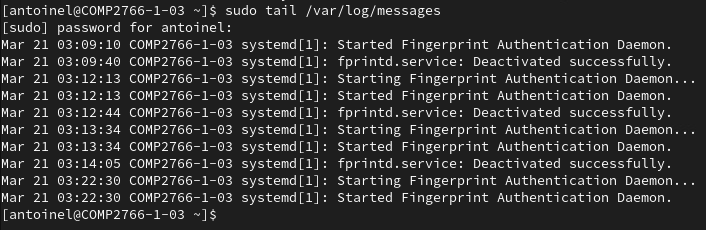
1. In your terminal window, click the tab for ***root***
2. As user *root*, enter a command line that sets only the SUID bit on /usr/bin/sudo without changing any other permissions. ***Insert a screenshot*** showing the command line that you entered and its output, here:



1. As user *root*, enter a command line that outputs a long listing of /usr/bin/sudo . Verify that the lowercase s has reappeared and that the permissions are now --**s**--x--x (meaning that the SUID bit has been set). If they are different, then you must reverse the changes you made in the previous step and do it properly.
2. In your terminal window, click the tab for ***yourFirstName***.
3. (**1 mark**): As user ***yourFirstName***, re-enter this command line that you entered in step #28 by filling in the blank with the appropriate command that you learned in the NDG course to allow you to tail the messages log file:

$ sudo tail /var/log/messages

***Insert a screenshot*** showing the command line that you entered and its output, here:



You can now tail the messages log file, because the SUID bit is set. When you executed /usr/bin/sudo, it was executed as the user owner which is root who can access /etc/sudoers which contains the rule that you entered in a previous step above:

*yourFirstName* ALL=/usr/bin/tail /var/log/messages

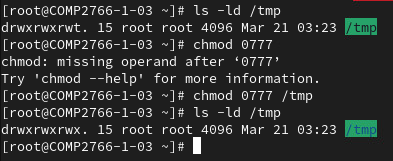
**THE STICKY PERMISSION BIT**

The sticky permission bit is set only on directory files. It has no effect when set on regular files. When set on a directory file, files in that directory can only be deleted by the file’s user owner, user root, and the user owner of that directory. This prevents accidental or intentional deletion of other people’s files.

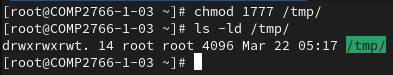
1. In your terminal window, click the tab for ***root***
2. (**1 mark**): As user *root*, enter a command line that outputs a long listing of the /tmp directory file:

# ls -ld /tmp

Notice who the user owner is. The presence of a lowercase t in the other access class permissions section indicates that the sticky bit is set and that there is a lowercase x permission underneath. To verify that, as user *root*, enter a command line that removes only the sticky bit without changing any other permissions. ***Insert a screenshot*** showing the command line that you entered and its output, here:



1. (**1 mark**): As user *root*, enter a command line that sets only the sticky bit without changing any other permissions. ***Insert a screenshot*** showing the command line that you entered and its output, here:



1. As user *root*, enter a command line that outputs a long listing of /tmp. Verify that the lowercase t has reappeared and that the permissions are now **rwxrwxrwt** (meaning that the sticky bit has been set). If they are different, then you must reverse the changes you made in the previous step and do it properly.
2. In your terminal window, click the tab for user ***elton***
3. As user ***elton***, enter the following command line to create a file in /tmp:

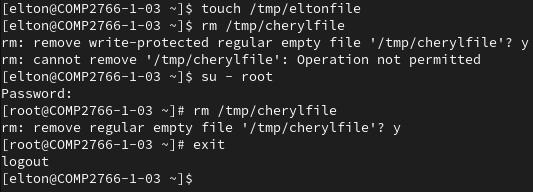
$ touch /tmp/eltonfile

1. In your terminal window, click the tab for user ***cheryl***
2. As user ***cheryl***, enter the following command line to create a file in /tmp:

$ touch /tmp/cherylfile

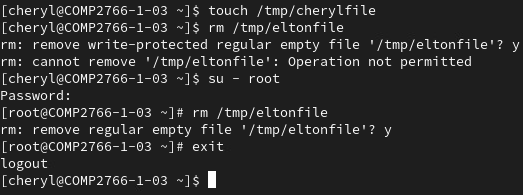
1. In your terminal window, click the tab for user ***elton***
2. (**1 mark**): As user ***elton***, enter a command line to try to remove /tmp/cherylfile

You will be asked to confirm whether you want to remove a write-protected file. Enter y to confirm. Then, you should see a permission denied error message. Your attempted removal failed because the sticky bit is set on the directory. User elton cannot delete other users’ files in /tmp, unless he is root, the user owner of that file, or the user owner of /tmp. ***Insert a screenshot*** showing the command line that you entered and its output, here:



Ne pas supprimer /tmp/cheryfile

1. In your terminal window, click the tab for user ***cheryl***
2. (**1 mark**): As user ***cheryl***, enter a command line to try to remove /tmp/eltonfile which should fail because the sticky bit is set on the directory. User cheryl cannot delete other users’ files in /tmp, unless she is root, the user owner of that file, or the user owner of /tmp. ***Insert a screenshot*** showing the command line that you entered and its output, here:



Ne pas supprimer /tmp/eltonfile

**THE SGID (a.k.a. set GID) PERMISSION BIT**

When set on a directory file, the SGID permission bit sets the group owner of newly created files in that directory to the group owner of that directory.

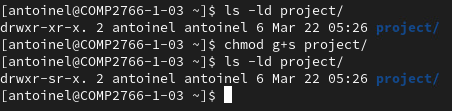
1. In your terminal window, click the tab for ***yourFirstName***
2. As user ***yourFirstName***, enter these command lines to create a project directory in your home and to let others access your home directory:

$ chmod o+x ~

$ mkdir ~/project

$ chmod g+w ~/project

1. (**1 mark**): As user ***yourFirstName***, enter a command line that sets only the SGID permission bit of ~/project without changing any other permissions. ***Insert a screenshot*** showing the command line that you entered and its output, here:

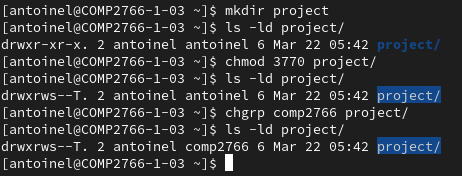


1. As user *yourFirstName*, enter a command line that outputs a long listing of the ~/project directory file:

$ ls -ld ~/project

The presence of a lowercase s in the group owner permissions section indicates that the SGID bit is set and that there is a lowercase x permission underneath.

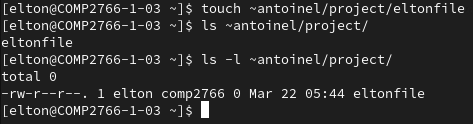
1. Enter a command line that removes all permissions from only the other access class for ~/project. Only you and group comp2766 will have any permissions to the project directory.
2. Enter a command line that sets only the sticky bit for ~/project to prevent members of group comp2766 from deleting each other’s files unless they are the user owner of ~/project or the user owner of the file being deleted.
3. (**1 mark**): Enter a command line to change the group owner of ~/project to group comp2766. ***Insert a screenshot*** showing the command line that you entered and its output, here:



1. In your terminal window, click the tab for ***elton***
2. (**1 mark**): As user ***elton***, enter these two command lines to create a file in ~/project and verify that the group owner is group comp2766, ***not*** group elton. ***Insert a screenshot*** showing both command lines that you entered and their output, here:

$ touch ~yourFirstName/project/eltonfile

$ ls -l ~yourFirstName/project/eltonfile



**EXPLORING DEFAULT PERMISSIONS AND THE UMASK**

1. (**1 mark**) As user *yourFirstName*, enter the following command lines:

touch mynewfile

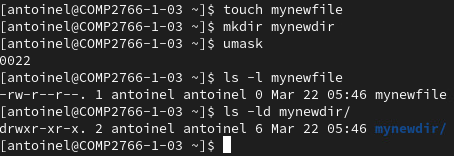
mkdir mynewdir

umask

ls -l mynewfile

ls -ld mynewdir

***Insert a screenshot*** showing all five of the above command lines and their output, here:



(**2 marks**): Fill in the empty cells in the following table **using both symbolic and octal notation** by referring to the output of the above command lines. Note that the NDG course uses different terms than those in the table below and in the lecture slides: *Initial* (instead of Kernel default permissions) and *Result* (instead of Effective default permissions).

|  |  |  |
| --- | --- | --- |
|  | **mynewdir**  **symbolic octal** | **mynewfile**  **symbolic octal** |
| Kernel default permissions | rwxrwxrwx 777 | rw-rw-rw- 666 |
| Your user’s default umask | ----w--w- 022 | ----w--w- 022 |
| Effective default permissions | rwxr-xr-x 755 | rw-r--r-- 644 |

1. (**1 mark**) As user *yourFirstName*, enter a command line that **immediately** changes your user’s umask in your user’s current shell to a value that results in the other access class having no permissions to either a newly created directory file or a newly created regular file. Note that a user has only one umask at a time and that the umask applies to both the creation of directory and regular files.

Then, enter the following command lines to confirm that your user’s new umask has masked out all the permissions for the other access class whenever you create a new regular or directory file:

touch mynewfile2

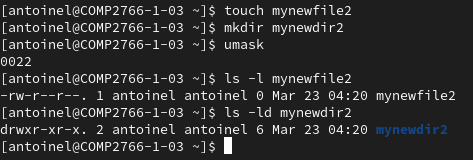
mkdir mynewdir2

umask

ls -l mynewfile2

ls -ld mynewdir2

***Insert a screenshot*** showing all five of the above command lines and their output, here:



Le umask n’a pas été modifié correctement

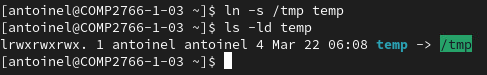
1. (**1 mark**) As user *yourFirstName*, modify the appropriate file so that every time a new shell is opened for your user, your user’s umask will be what you calculated in the previous step. ***Insert a screenshot*** showing the changes that you made to the appropriate file and the name of that file, here:



1. Remove the line that you added to the appropriate file in the previous step. Then, close your user’s current shell (ex: use the exit command) and open a new shell by using the su command with its login option (ex: su - yourFirstName). Confirm that your user’s umask has reverted to the default value of 0022

**CREATING LINKS**

1. In your terminal window, click the tab for ***yourFirstName***
2. (**1 mark**): You often forget that there is no letter e in /tmp. In your home directory, enter a command line that creates a link named temp to the /tmp directory. ***Insert a screenshot*** showing both command lines that you entered and their output, here:



1. Enter these two command lines and compare their output:

* ls /tmp
* ls temp

Notice that you can refer to /tmp using temp, because the latter points to the former.

Enter this command line and look at the details about temp:

**ls -l**

Notice this about symbolic links (a.k.a. symlinks or soft links), in general:

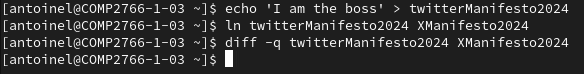
* a symlink involves two files: the symlink, which is a pointer, and the file that it points to
* the file type character found in the first column of a long listing is **l** for a symlink

1. (**1 mark**): Cheryl Musk has hired you to help him with the transition of his company’s name from Twitter to X. Enter this command line:

echo ‘I am the boss’ > twitterManifseto2024

To help employees during this transition period, he would like you to enable them to refer to that file using either twitterManifseto2024 or XManifseto2024 , but without using a symlink.

***Insert a screenshot*** showing the command line that you entered and its output, if any, here:



1. Enter these two command lines and compare their output:

* cat twitterManifseto2024
* cat XManifseto2024

Notice that you can refer to twitterManifseto2024 using XManifseto2024, because both refer to the same single file.

Enter these two command lines to append data:

* echo ‘I love myself’ >> twitterManifseto2024
* echo ‘I am the greatest’ >> XManifseto2024

Enter these two command lines and compare their output:

* cat twitterManifseto2024
* cat XManifseto2024

Notice that whatever you do to twitterManifseto2024, you do to XManifseto2024, because they both refer to the same, single file.

1. Enter these two command lines and compare their output:

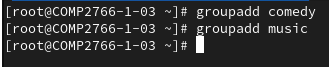
* ls -li twitterManifesto2024
* ls -li Xmanifesto2024

Notice that:

* the inode for both files is identical, because there is only one file (with two names)
* the file type character found in the first column of a long listing is **-** for a hard link
* the integer that appears immediately after the permission columns in a long listing indicates the number of links (that is, ways to refer to that file by name) and it is the same for twitterManifesto2024 and Xmanifesto2024 (it should be 2, because you can refer to the file using either name).

**USERS AND GROUPS**

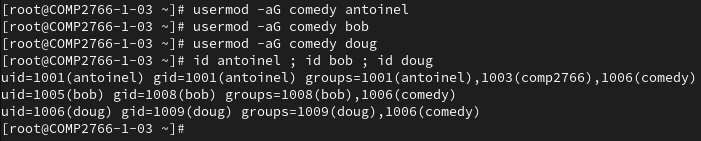
1. In your terminal window, click the tab for ***root***
2. (**1 mark**) Create two secondary groups named comedy and music (without any members). Insert a screenshot, here, that shows the command lines you entered and their output if any:



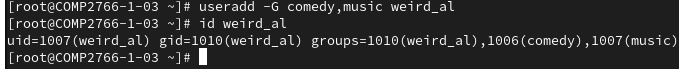
1. (**1 mark**) Create two users named bob and doug (***without*** adding them to any secondary groups, yet). Insert a screenshot, here, that shows the command lines you entered and their output, if any, here: (**NOTE: you can only create one user at a time**)



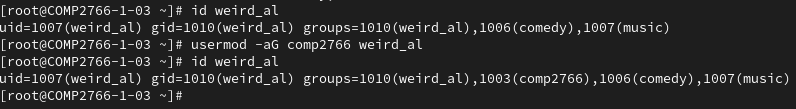
1. (**1 mark**) Now, add users bob, doug, and ***yourFirstName*** to group comedy. Insert a screenshot, here, that shows the command lines you entered and its output. (**NOTE: you must add each user individually)**



1. (**1 mark**) Using ***ONE*** command line containing just one command, create a user named weird\_al and, at the time that you create him, make him a member of groups comedy and music. Do **NOT** create user weird\_al and, then, add him to those groups. Insert a screenshot, here, that shows the single command line you entered and its output, if any, here:



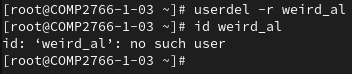
1. (**1 mark**) Enter ***ONE*** command line containing just one command that adds user weird\_al to group comp2766 while maintaining his existing group memberships. (**NOTE: Ensure that, when you add him to this group, he is not removed from other groups that he already bcherylgs to.**) Insert a screenshot, here, that shows the command line you entered and its output, if any, here:



1. (**1 mark**) User doug was caught stealing a two-four, so your manager says you must lock his account so that he cannot login. Insert a screenshot, here, that shows the command line you entered and its output, if any, here:



1. (**1 mark**) User weird\_al never figured out how to login, so just delete his account and his home directory using a single command line and the appropriate switch. He will never notice. Insert a screenshot, here, that shows the command line you entered and its output, if any, here:



1. (**2 marks**) Enter a command line using switches and arguments that changes the password aging information for user bob as follows (NOTE: You may ***not*** change the password aging information interactively):
   1. Last date password changed: March 17, 2024
   2. Minimum password age: 3 days
   3. Maximum password age: 90 days
   4. Warning days: 7 days
   5. Inactivity days: 5 days

Then, enter a command line that lists the password aging information for user bob.

Insert a screenshot showing the TWO command lines that you used above, here:

