**LAB PROJECT 03**

Oregon Tech

MIS 240: Linux Fundamentals

Ethan Dunzer

7/14/20

**Instructions:** In this week’s lab, we are focusing on users, groups, ownership, and permissions. Part of administering any system includes the ability to control permissions, meaning who has access to what on the shared system.

At some point in this course, you have probably received the ‘permission denied’ output when attempting various commands that require administrative privileges or when trying to view contents of files that you do not have permissions to access. In this lab, you will perform a mix of commands used to manage users, groups, and permissions.

It is important to fully understand permissions and how they can affect certain processes, services, and the overall behavior of the operating system. From a security standpoint, Linux’s approach to permissions is one of the features that make Linux more secure than other operating systems. It is important that we do not give full permissions to the entire filesystem, but rather pick and choose what permissions we give and to whom.

As you work through the lab, I will ask for screenshots of your output or answers that you’ll receive based on your output. This document allows you to enter your answers and submit it with your Word document containing your screenshots. Paste your screenshots into a Word document with a brief explanation of each screenshot. Make sure you have a cover page with your full name.

**Note:** This week I am providing even less guided instructions. My expectation is you will use your resources to learn how to use the commands to complete the following tasks.

\*\*The following can be completed on either VM, Ubuntu, or CentOS.

1. Viewing users, groups, and passwords
   1. From the command prompt, use the tail command to view the last 2 entries of the /etc/passwd file.
   2. Issue the same command, but this time for the /etc/shadow file.
   3. Along with your screenshot, answer the following questions. Page 154 of your textbook will help here.
      1. What is your user account name?

edunzer

* + 1. What character is in the former password filed?

x

* + 1. What is your user’s unique user ID (UID)?

1000

* + 1. What is your primary group ID (GID)?

1000

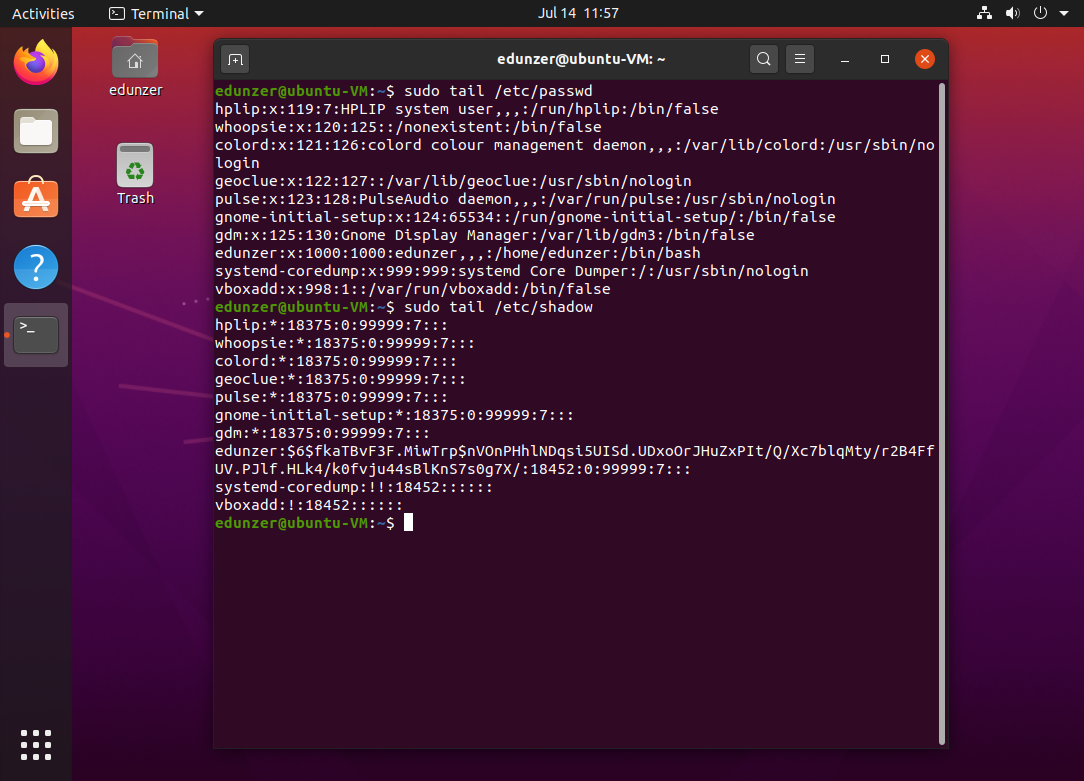
* + 1. What shows in the comment filed?
    2. What is your home directory?

/home/edunzer

* + 1. What does the system use for your default login shell?

/bin/bash

**\*\*\*Screenshot #1, output of your tail commands along with answers to the above questions**



1. Add a user to the system via the command line
   1. Using the useradd command, add a user ‘frank’ (without the quotes)
   2. Use the tail command to view your newly added user in both the passwd and shadow files and answer the following questions.
      1. What is frank’s UID?

1001

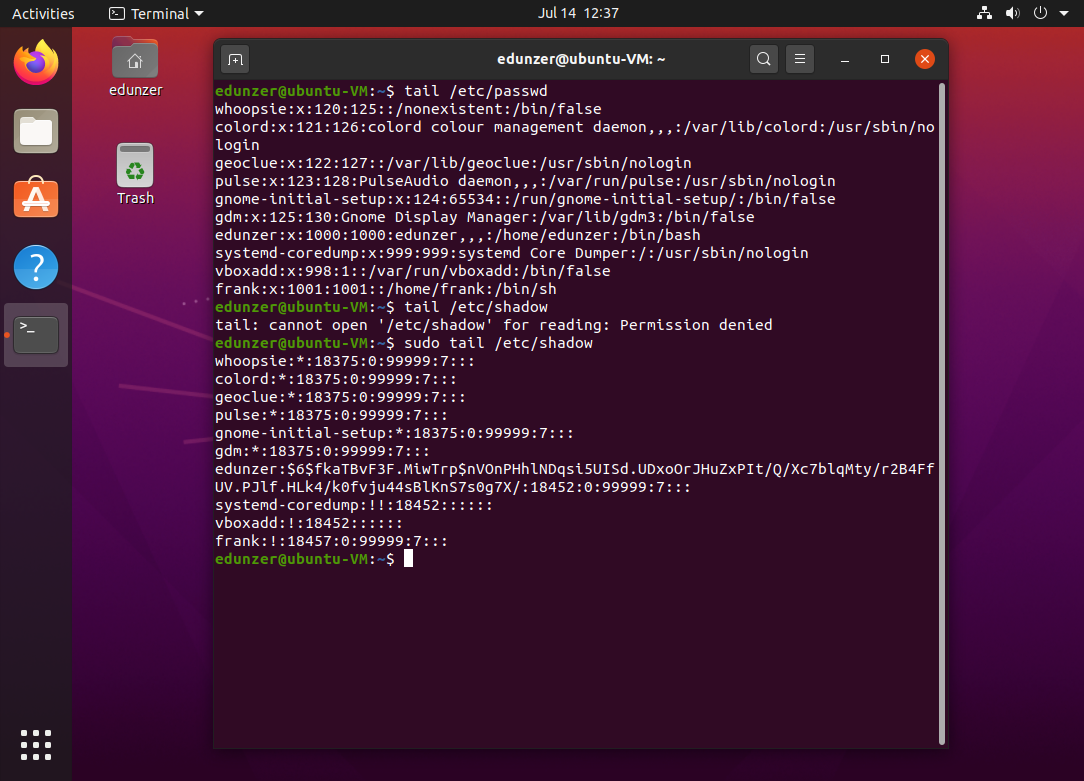
* + 1. What is frank’s GID?

1001

* + 1. Is there a password hash for frank? Explain.

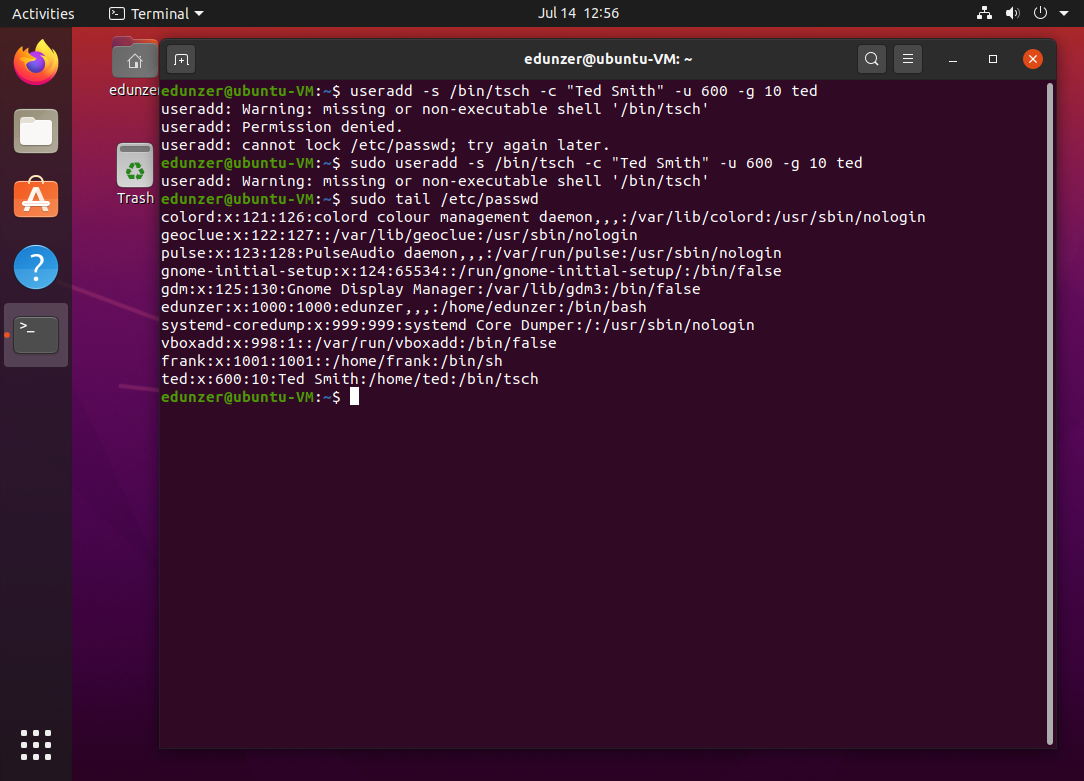
No there is not a hashed password for fank yet

**\*\*\*Screenshot #2, output of the above commands with answers to the questions**



* 1. Now, add another user to the system called, ted. This time, use the different options available with the useradd command to assign a full name of “Ted Smith”, a GID of 10, a UID of 600, and a default shell of /bin/tsch to your new user.
     1. **Hint:** this is a more complicated user add. Use the manual pages to view the different options for the useradd command, man useradd
  2. Use the tail command to view your changes.

**\*\*\*Screenshot #3, output of your new user with manual configurations, be sure to show the command used to add the new user.**

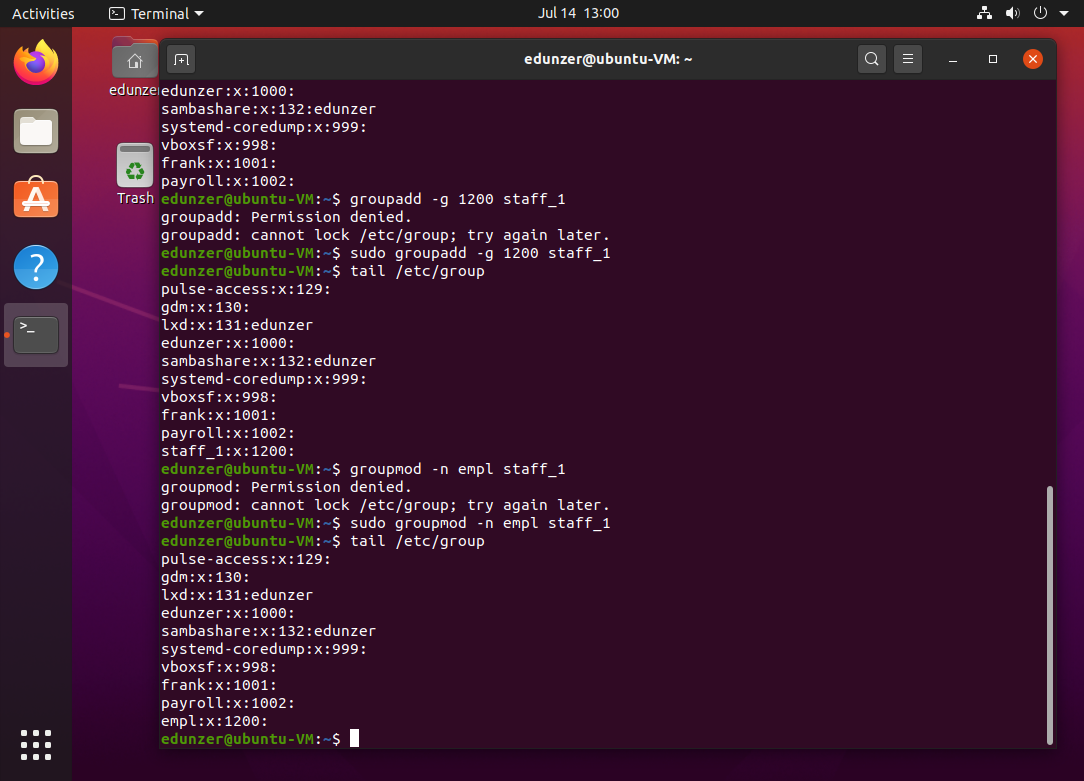


1. Creating a group in Linux
   1. Use the groupadd command to add a group called ‘payroll’ (without the quotes)
   2. View the end of the /etc/group file to confirm your changes
      1. What is the GID of the new group?

1002

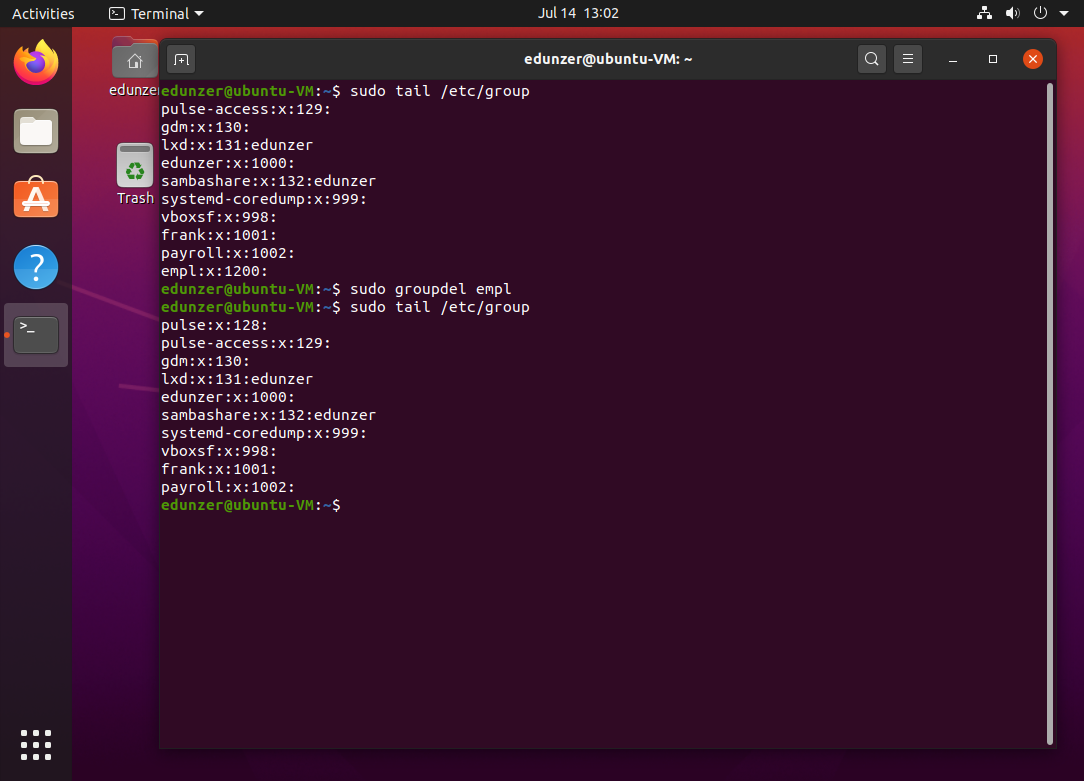
* 1. Add a new group called ‘staff’ and manually assign the GID of 1200
  2. View the end of /etc/group to confirm your changes
  3. Modify the staff group and rename it to ‘empl’
     1. **Hint:** use the groupmod command, you may need to use man groupmod to see how you can change group names
  4. Use the grep command to find the empl group in the /etc/group file

**\*\*\*Screenshot #4, show your output of the above commands along with any answers**



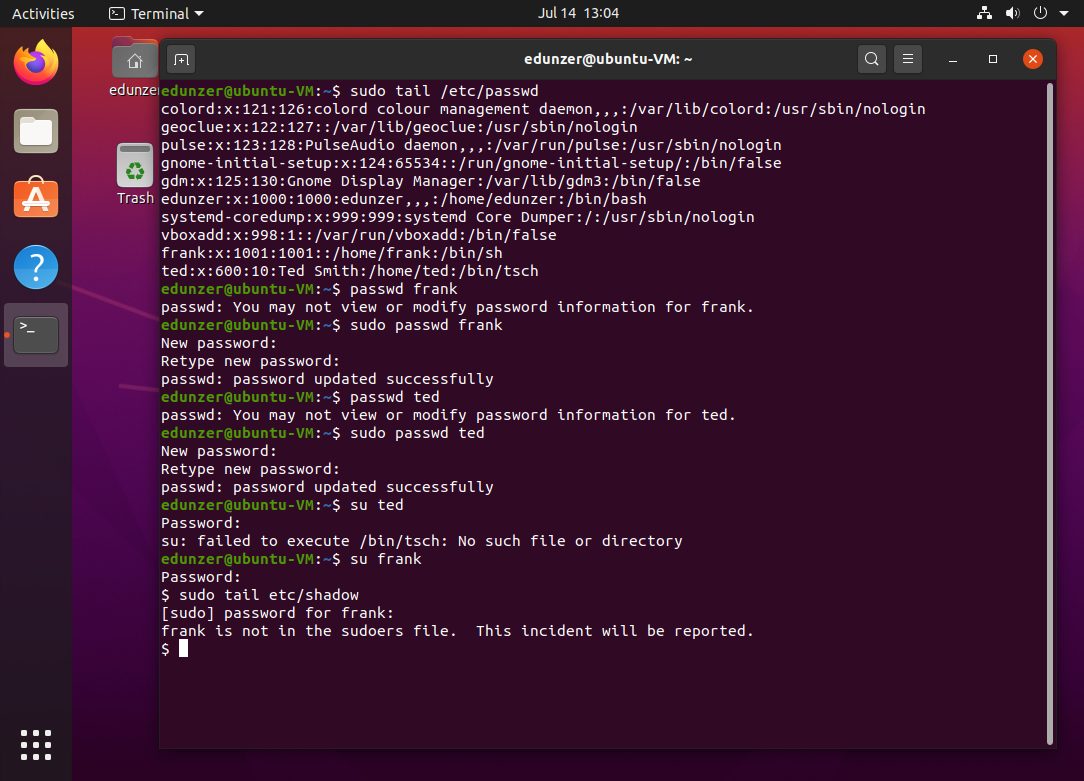
1. Delete a group
   1. Find the command needed to delete a group and delete the empl group.
   2. Use the grep command to attempt to find the empl group to verify your changes

**\*\*\*Screenshot #5, show your commands and output showing the deleted group**



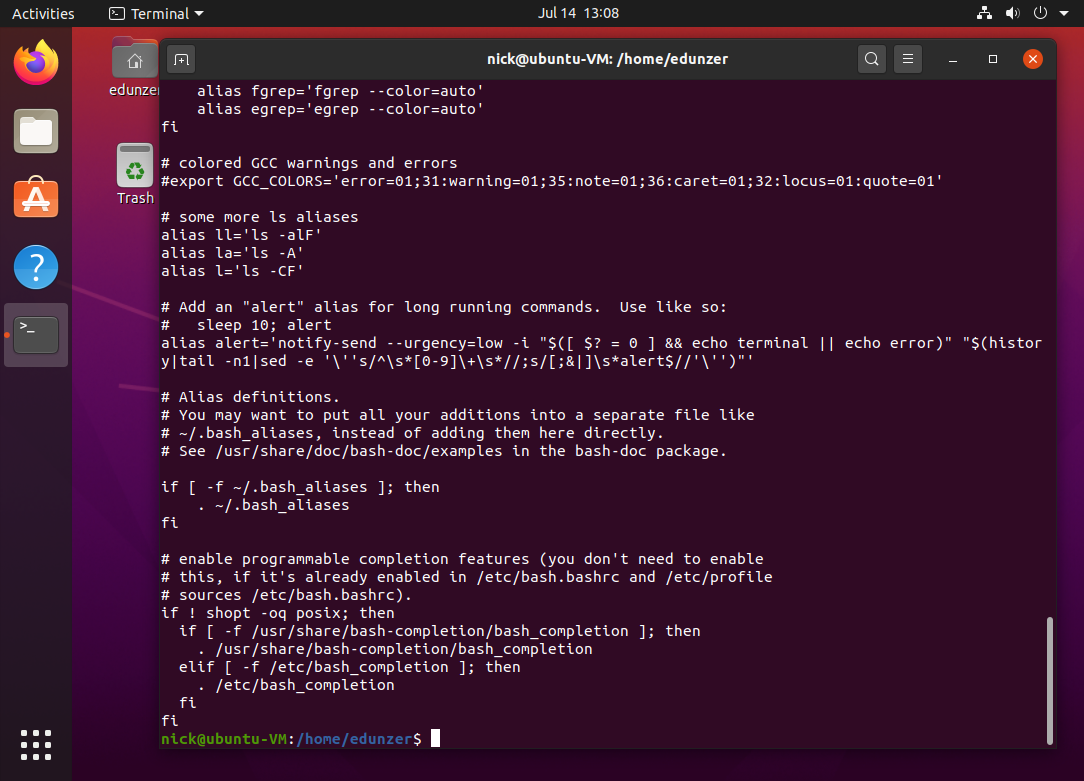
1. Create / modify passwords
   1. You noticed above when you added a new user, a password was not created immediately.
   2. Using the passwd command, give both frank and ted a password of your choice
   3. After creating passwords, switch users to either frank or ted using the su command

**\*\*\*Screenshot #6, show your outputs and you logged in as one of the new users**



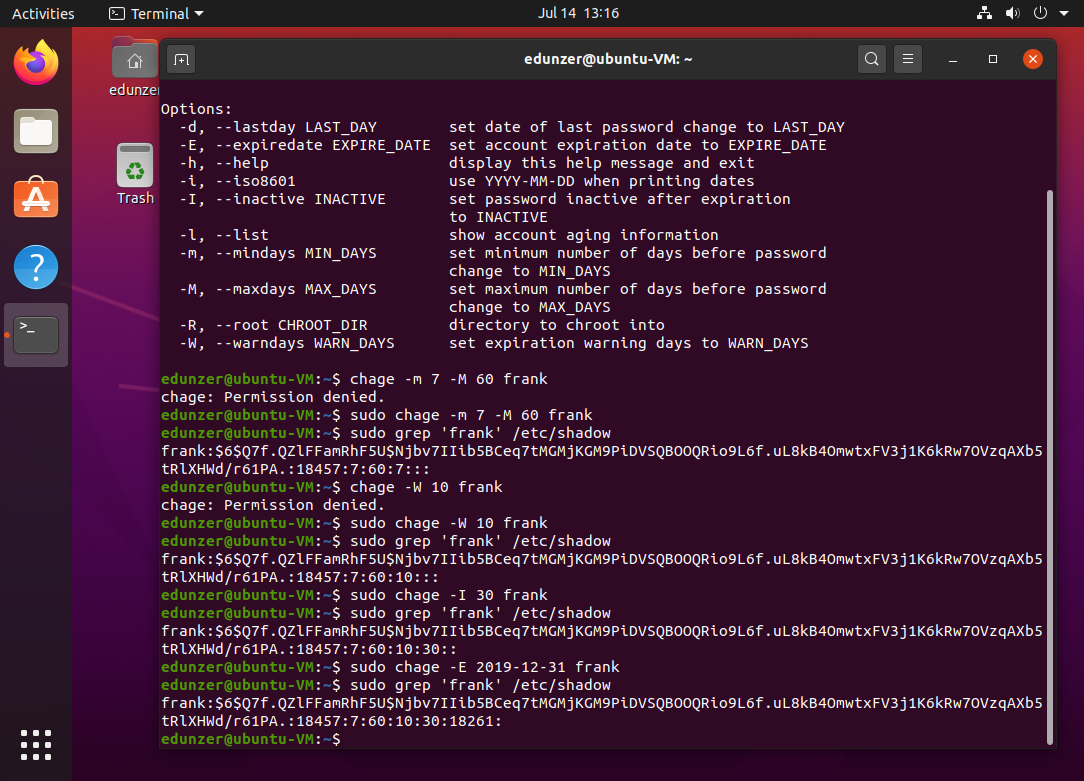
1. Changing default profiles
   1. Before adding new users to your system, you have a chance to change the environment for all new users by editing the default .bashrc file. However, this will not change the environment for existing users.
   2. Use nano to edit this file: /etc/skel/.bashrc
      1. Enter a new alias ls=”ls –l” on the last line of the file
      2. Save and close the file
   3. Now, create a new user called nick and give nick a password
   4. Logout of your current user account and login as nick
   5. Concatenate the .bashrc file in the home directory of nick
      1. What is the last line in the .bashrc file?

**\*\*\*Screenshot #7, show the output of the end of the .bashrc file for your new user and be sure to answer any questions**



1. Modify password requirements
   1. Make sure you are back to your regular user account (not nick).
   2. Use the grep command to see any entries containing frank in the shadow file
   3. Use the chage command (notice the missing n in chage), and set the minimum days to 7 and the maximum days to 60 for the user frank.
      1. Hint: you’ll need to reference the manual page for chage to get your options and login set properly.
   4. Use the grep command again to view the frank entry in the shadow file. You should see your changes at the end of the password hash.
   5. When you set min/max days for passwords, it’s a common courtesy to warn your users their password is about to expire.
      1. Use the chage command to change the number of warning days to 10 for frank.
      2. Again, verify with grep.
   6. Inactivity is another useful account control entry to lock a user’s account due to inactivity (security reasons). Use the chage command to change frank’s inactivity setting to 30 days.
      1. Again, grep to verify your changes.
   7. One more use of the chage command. This time, set the expiration date of frank’s account to December 31, 2019.
      1. Notice, in the manual there is a specific format for the date
   8. Finally, use the grep command to verify your changes

**\*\*\*Screenshot #8, shadow output showing password settings for frank**



1. Special user accounts (system user)
   1. Sometimes you need special user accounts to run services, but you do not need them to be able to login to the system. All services need user permissions to run, if you use the top command, you’ll see the root user runs a lot of services.
   2. Create a system user called test
      1. Hint: you’ll need to look through the useradd manual to get the right option to create a system user
   3. Use the grep command to view the test user in the passwd and shadow files.
      1. What is the UID of the test user?

1002

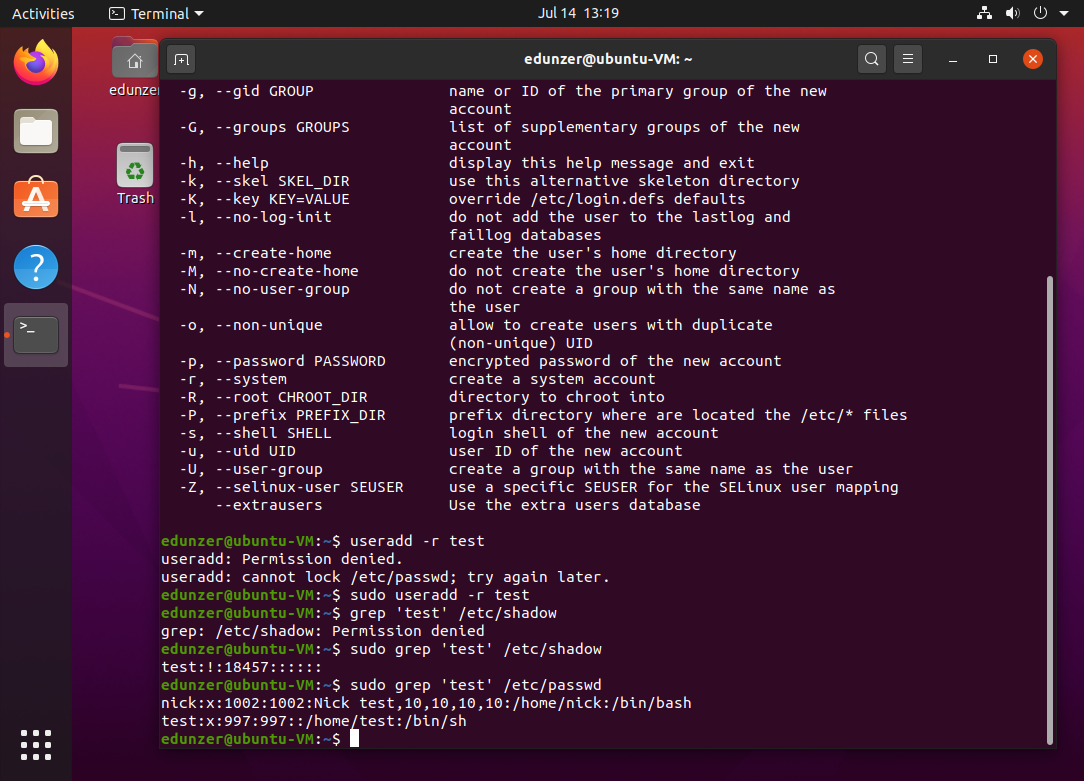
* + 1. What is the GID of the test user?

1002

* + 1. What do you notice in the shadow output when compared to standard user accounts?

There is an ! instead of an X

**\*\*\*Screenshot #9, show your output of the newly created special user**



1. Setting Ownership
   1. From your home directory (not root), create 2 new directories, one called dogs, the other called cats
   2. Run the list command with the –al options
   3. What permissions are set for your cats and dogs directories? (Hint: if you need help determining permissions, reference pages 33 & 34 in your textbook)
      1. What are the permissions for users?

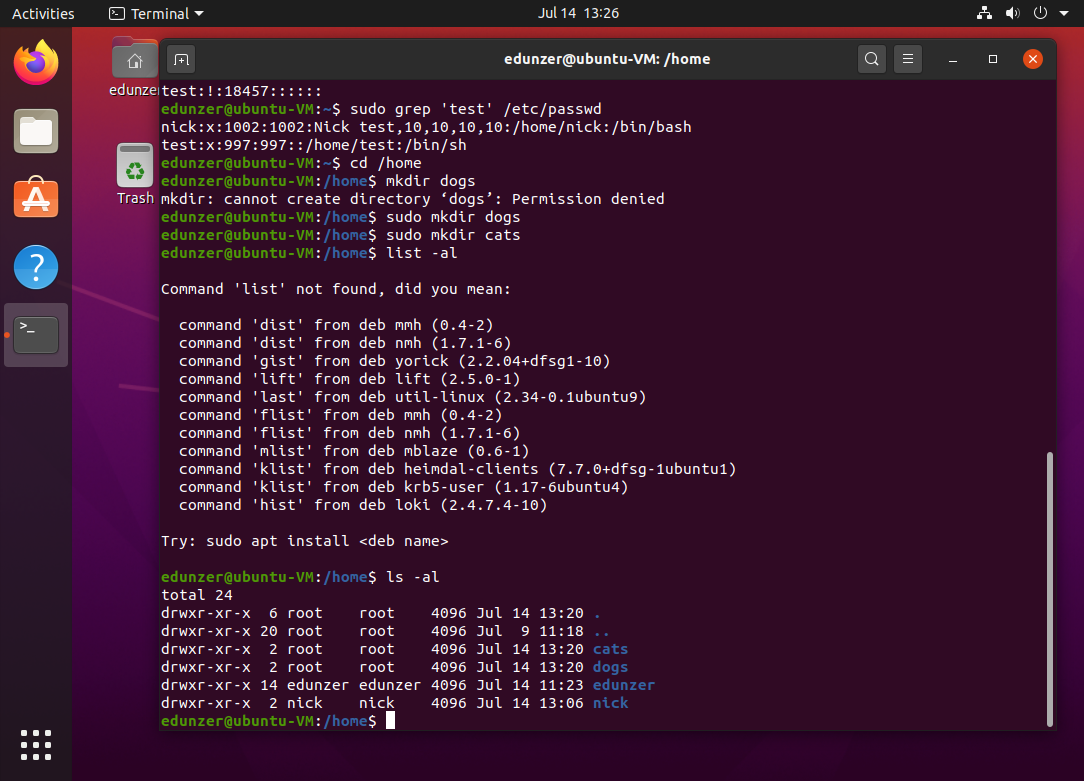
Read, write and excute

* + 1. What are the permissions for the group?
    2. What are the permissions for others?

Read, write and excute

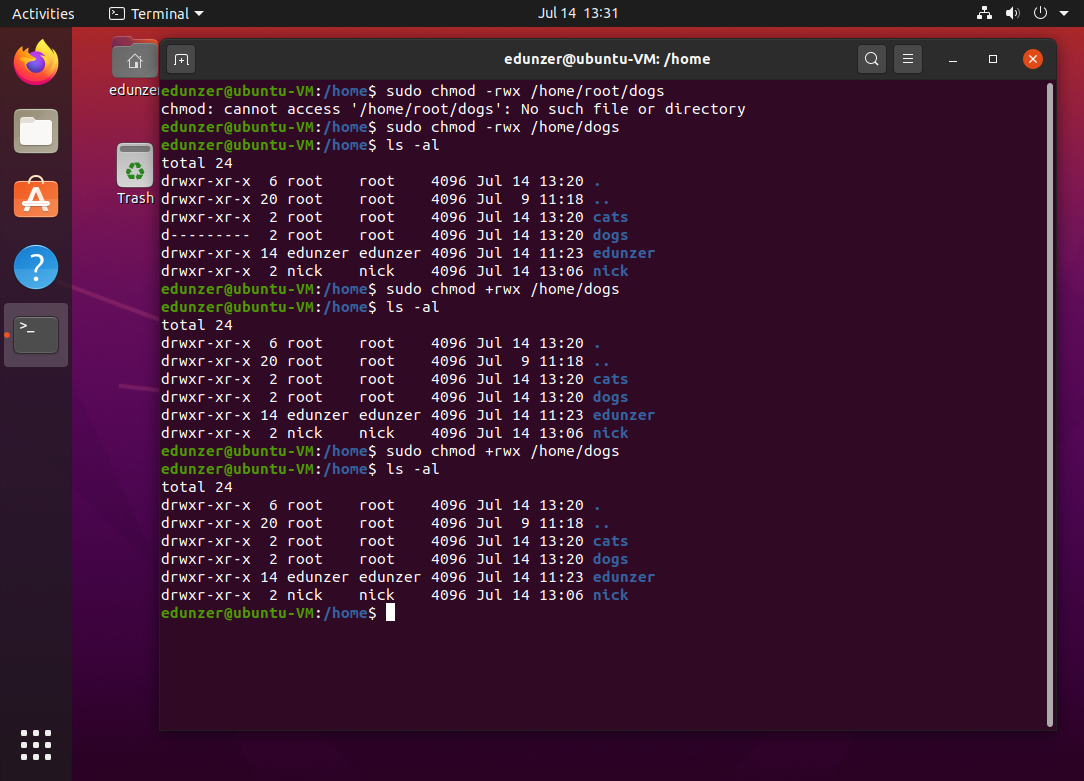
* 1. Use the chgrp command to change the group owner to root for the /home/<username>/dogs directory
     1. Remember, use your help commands to get the command right
     2. Also, <username> means put in the username you used to create your 2 directories (not root)
  2. Use the ls -al command to view your ownership changes to the /dogs directory
  3. Now, use the chown command to change the user owner to root for the /home/<username>/dogs directory.
  4. Use your ls -al command again to view your changes
     1. At this point, you should see root root, instead of <username> <username> as the owners for your /dog directory

**\*\*\*Screenshot #10, showing root ownership for both user and group**



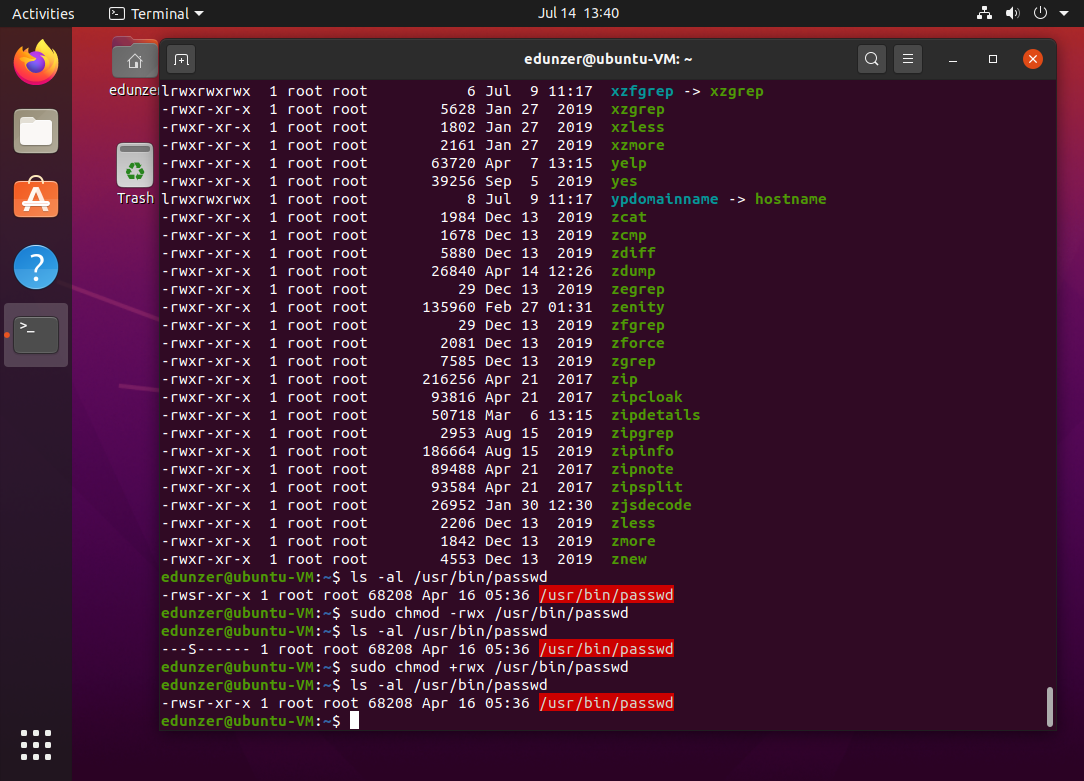
1. Setting Permissions
   1. Use the chmod command and symbolic notation to add write permissions to others for the /home/<username>/dogs directory
      1. Again, reference Chapter 2 for help. Symbolic notation is letters; octal notation is numbers. Spend some time for review to help you.
   2. Use the ls -al command to verify your changes
   3. Now, use octal notation with the chmod command to give users and groups rwx and others r-x permissions
   4. Use the ls -al command to verify your changes

**\*\*\*Screenshot #11, show permission changes to /home/<username>/dogs directory**



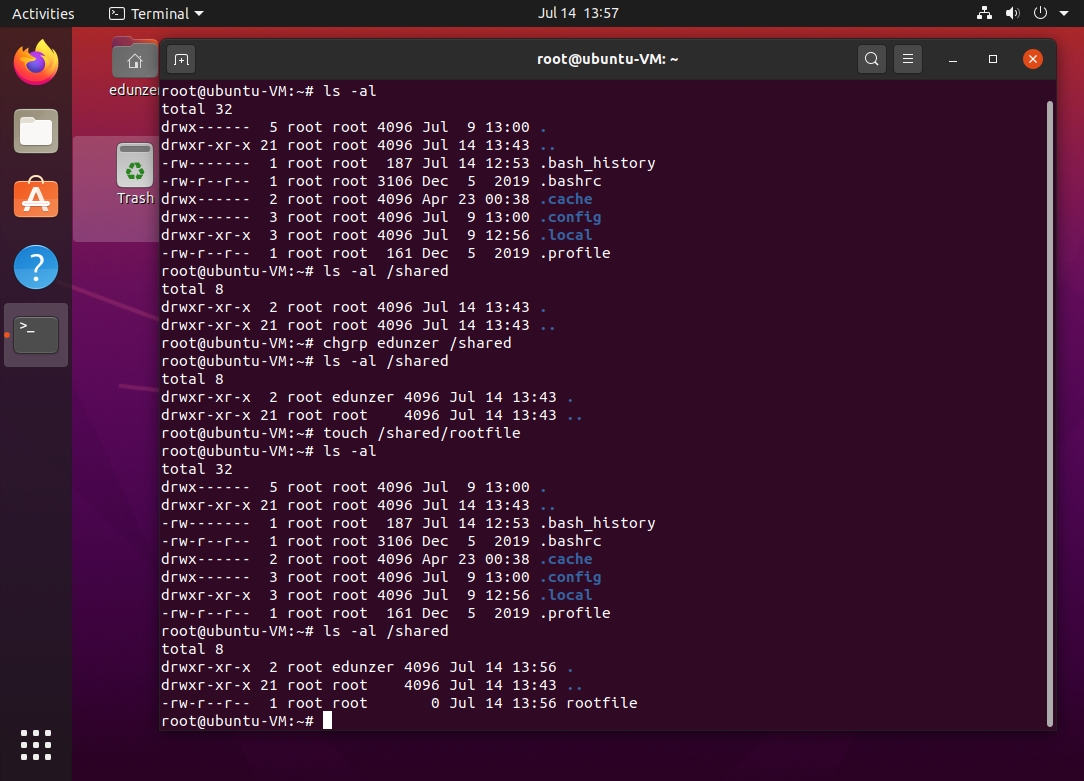
* 1. Now we’re going to set some special permissions on /usr/bin/passwd
     1. Use the ls -al command to view current permissions for /usr/bin/passwd
        1. You should see the user has rws permissions. Please read the middle paragraph on page 34 on suid.
     2. Use the chmod command to remove the suid permission from the /usr/bin/passwd file
        1. The easiest way is to use symbolic notation
     3. Use the ls -al command to view your changes
     4. Use the chmod command to add the suid permission back to the /usr/bin/passwd file
     5. Again, use the ls -al command to view your changes

**\*\*\*Screenshot #12, output showing removal and addition of special permission**

**clear**

1. Permission Inheritance
   1. Permission inheritance allows us to keep permissions consistent as we create new files and directories from a main directory. This way, you can set permissions once and not have to worry about who has access to what.
   2. As root, make a new directory /shared
   3. Use the chgrp command to change group owner to users on the /shared directory
      1. Use the ls -al command followed by a pipe and grep command to narrow your list command down to just shared
         1. Hint: you should only have one output
      2. You should see the root user and the group users own the /shared directory
   4. Use the chmod command to add group permissions
      1. add write and sgid permissions to the /shared directory
      2. Use the ls -al directory command to see your changes
         1. Hint: you should see groups with rws permissions
   5. Use the touch command to create /shared/rootfile
      1. Use the ls -al command to view the permissions of the newly created rootfile file
         1. You should see the users group automatically has ownership (set by the rws from the directory)
         2. This means anyone in the users group automatically has access to our new rootfile file

**\*\*\*Screenshot #13, output of permission inheritance commands**



1. Save your work and upload your document to Canvas for grading.