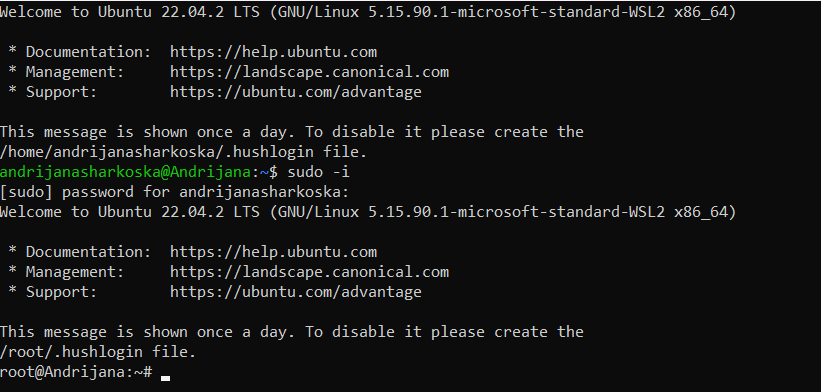
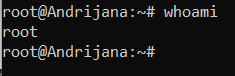
# Linux Commands - Part II

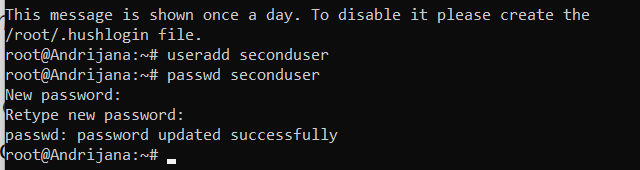
*1. Elevate your user access to root;*





***sudo -i*** –> to elevate access to the root user or also known as the superuser

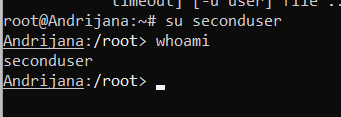
*2. Add a new user to your Linux OS and set a password for it;*



With the command ***useradd <user name>*** we can add a new user

***passwd <user name>*** → to set up the password for the new use

*3. Test if you can log in using that user;*



***su <user name>***→ to switch account and log in with another user account

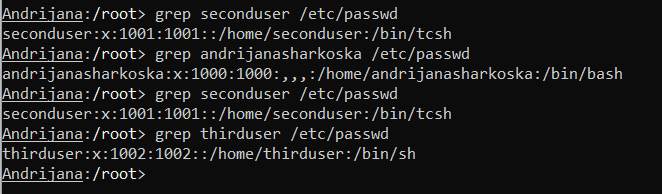
***whoami*** → to check if I’ve switched to the other user account

*4. Using grep command check if the user is created;*



* From what I’ve researched, each created user is stored in the ***etc/passwd*** text file, which stores essential information for login.
* Thus, I used the ***grep <user name>/etc/passwd*** command to search for the user

*5. grep the UID of each user;*



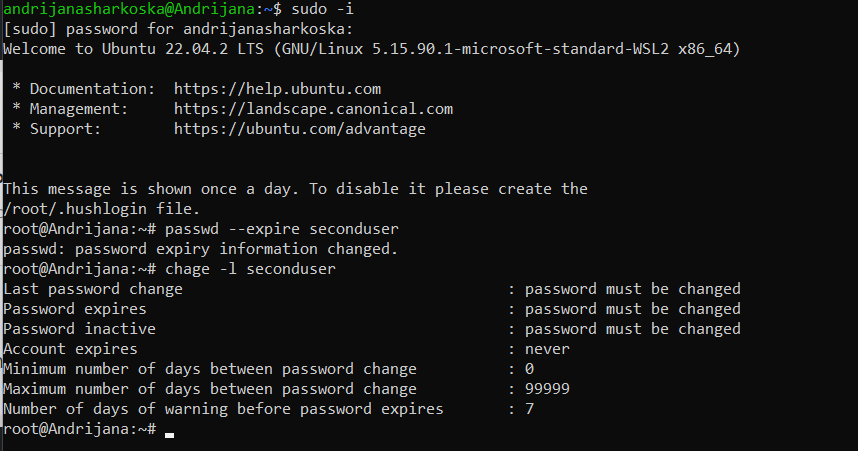
* And I tried this approach, using only the **grep** command with the *username* and the **/etc/passwd** command (not sure if it’s the right approach)

*6. Find out the GID of the created user;*



***id -g <user name>***→ finds the GID of the created user

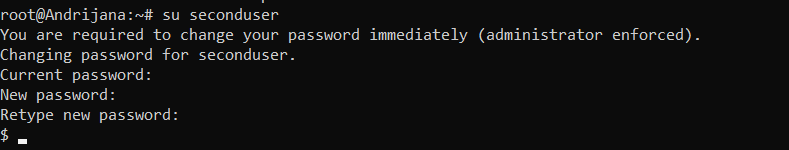
*7. Change the password of the user and force it to change the pass on his next login;*



* I had to elevate the access to root so I could change the password for the created user. First I had to set its expiration, so the next time the user logs in, will be forced to change the password.

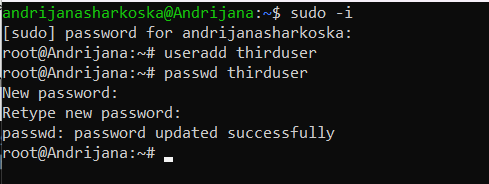
***passwd –expire <user name>*** → to set expiration on the password

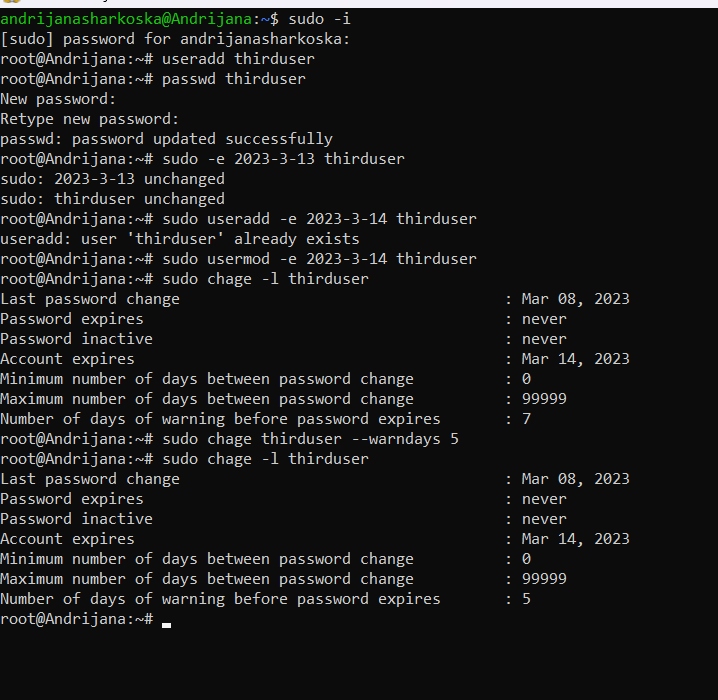
***chage -l <user name>*** → to check the password expiry information



* The next time the user tries to log in, a prompt will appear that they have to change the password immediately

*8. Add a new user and set an expiration date for it, with a five-day warning period;*





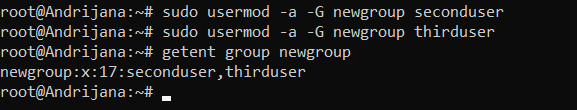
* I was not sure whether the expiration was pertaining to the user account or the password, so I set up both. You can see with the ***sudo chage -l <user name>*** command the user information, moreover, the expiration policy, and when the account and the password will expire.

*9. Create a new group;*



***groupadd -g <ID of the group> <name of the group>*** → adding a new group

*10. Assign the two new users to that group;*



***sudo usermod -a -G <group name> <user name>*** → to add an existing user to another group

***getent group <group name>*** → to check whether the users have been added to the group

*11. Lock one of the user accounts;*

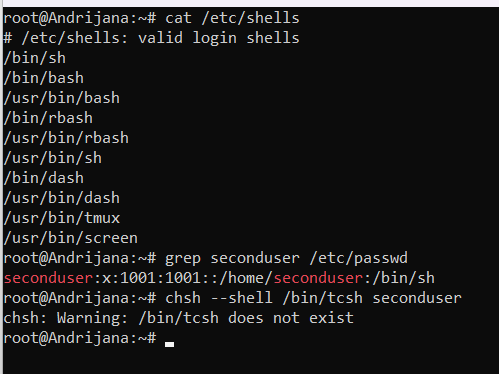




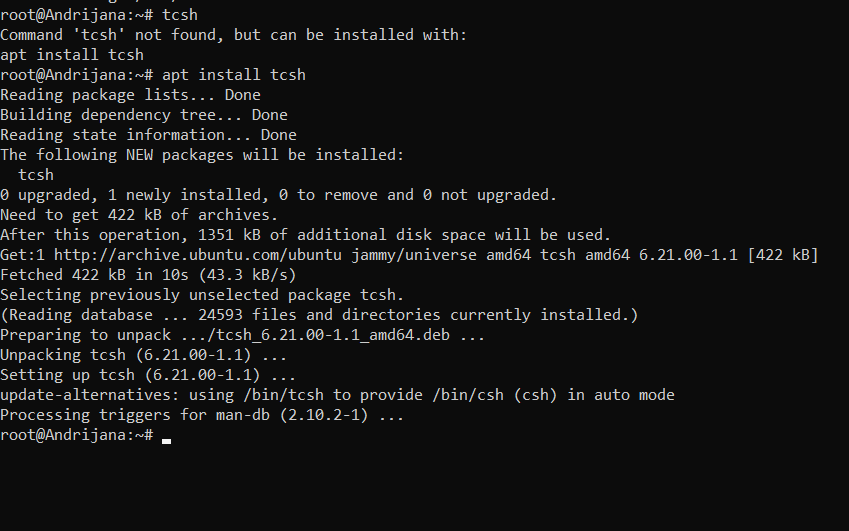
***passwd -l <user name>*** → to lock user’s account

***passwd –status <user name>*** → check whether the account was locked --> L is showing that the locking was successful.

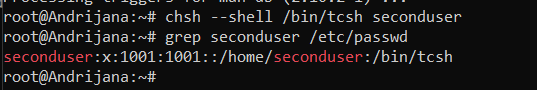
*12. Change the shell of one user to tcsh;*



* First, I am listing the available shells, and I need to have root access to be able to see the information
* Then I tried to change the shell to tcsh to one of the users, and I got a warning that it does not exist. The command I used for changing the shell is ***chsh –shell /bin/tcsh***
* By doing a bit of a research, I found out that I can install the **tcsh** shell in my Linux environment by typing tcsh into the terminal. However, that wasn’t the case, and I had to install it via the ***apt install tcsh*** command. Obviously, I had to do more extensive research, or maybe not in this case, since I got the answer right in the terminal (in front of my eyes)

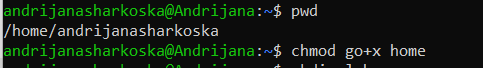


* ***chsh –shell /bin/<name of shell> <name of user>*** → to change the shell
* Finally, I was able to change the shell and check whether it was really changed



========================================================================

*13. Make sure your home directory has “execute” access enabled for group and other.*

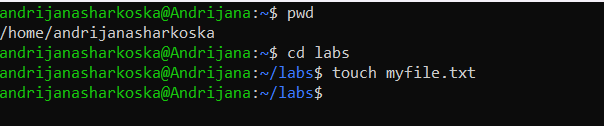


***chmod go+x home*** → created a home directory and set up the “execute” access with the ***x*** and ***go*** is used for specifying that the access is for the group and others

*14. Change to your home directory, and create a directory called labs;*



*15. Create an empty file in labs directory*

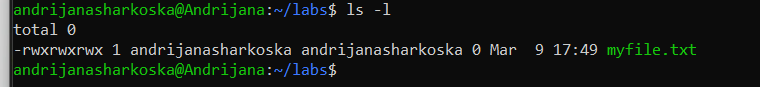


*16. Change permissions of file to rwx-rwx-rwx*



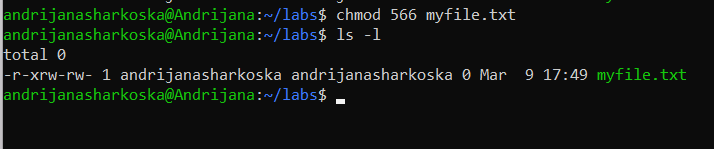
***chmod 777 <filename>*** → changes the file permissions to rwx-rwx-rwx

*17. List the file. What color is the file?*

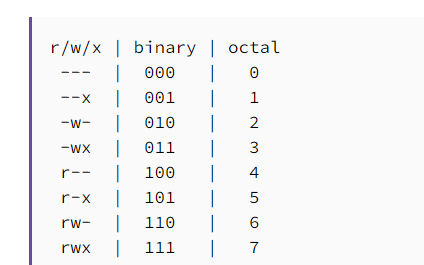


* The color of the file is green

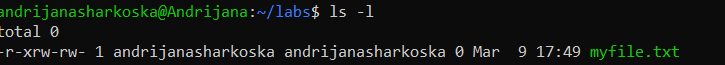
*18. Change the permissions back to rx-rw-rw*



* I used this table as a reference when typing the numbers for changing the permissions

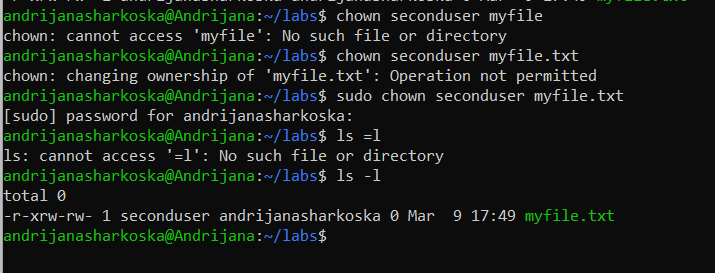


*19. Check what owners does the file have.*



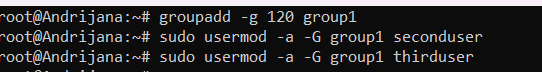
* My username shows as the owner of the file (andrijanasharkoska – the first name is the username)

*20. Change the user ownership of the file to another user;*



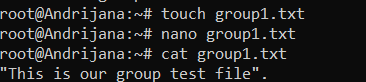
* First, I used the ***chown <user name> <file name>*** command to change the user, however, the operation was not permitted
* Thus, upon further research, I found out I can use the keyword **sudo** before since the **sudo/root** user has the permission to change system settings like changing ownership, adding or removing users, etc.
* **sudo chown <user name> <file name>** → to change the ownership of the file
* **ls -l** to list the files in order to verify the owner was actually changed

*21. Create a group called group1 and assign two users to the group;*

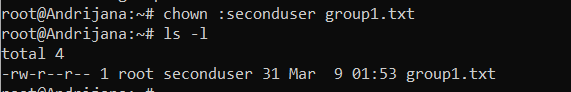


* I’ve added the existing group members for the sake of not creating additional users

*22. Create a file called group1.txt and redirect below input into the file: “This is our group test file”.*



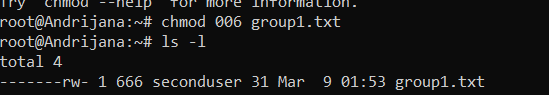
*23. Change the group of the file to one of your users;*

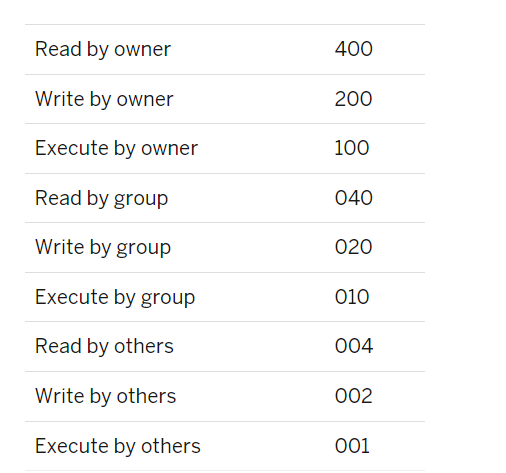


***chown :<user name> <file name>*** → to change the group of the file to one of the existing users

* Then I am listing the file information to confirm the changes have taken place

*24. Give members of the group group1 read/write access to this file?*





Sum all the accesses you wish to permit. For example, to give write and execute privileges to the owner of grop1.txt(200+100=300)

* In this case, we sum up the read by others and write by others (002 + 004 - 006) and get the **rw** permissions for the users.

0Other is **everyone that is not the owner or in the group**. For example, if you have a file that is root:root then root is the owner, users/processes in the root group have group permissions, and you are treated as other.