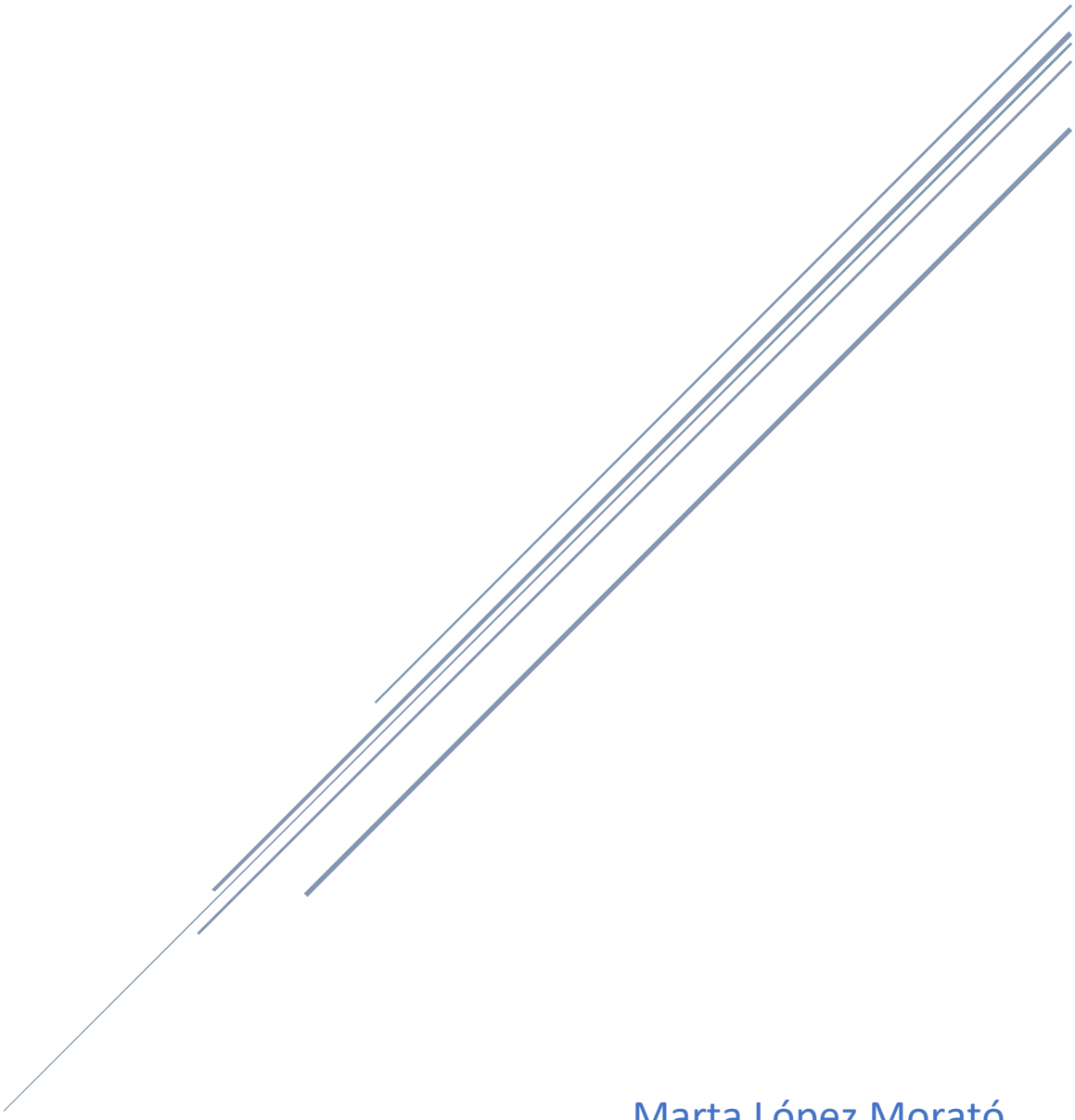
COMPUTER SYSTEMS

1st Term Assessable Activity



Marta López Morató

** Note: All the linux commands are written in italics and in green.*

EXERCISE 1

Section a

The commands to create the specified tree of folders, starting from the home directory are:

mkdir /tmp/"My Files"

mkdir /tmp/"My Files"/"Cook Recipes" /tmp/"My Files"/Comics /tmp/"My Files"/Movies

cd /tmp/"My Files"/"Cook Recipes"

mkdir /tmp/"My Files"/"Cook Recipes"/Salty /tmp/"My Files"/"Cook Recipes"/Sweet

touch /tmp/"My Files"/"Cook Recipes"/Sweet/Doughnut.jpg

touch /tmp/"My Files"/Comics/"Man of Steel.txt" /tmp/"My Files"/Comics/"Wonder Woman.txt"

mkdir /tmp/"My Files"/Movies/Superheroes /tmp/"My Files"/Movies/Comedy

Section b

1. **Create a file and add a text**

Commands to create the "Mac and cheese.txt file inside the /Cook Recipes/Salty folder:

touch /tmp/"My Files"/"Cook Recipes"/Salty/"Mac and cheese.txt"

To add the text using pipes or redirection:

echo "To make macaroni and cheese, you must first buy macaroni and cheese"> /tmp/"My Files"/"Cook Recipes"/Salty/"Mac and cheese.txt"

2. **Delete the Sweet directory**

The command is:

rm -R /tmp/"My Files"/"Cook Recipes"/Sweet

3. **Copy a file to another directory and add some text**

The commands are:

cp /tmp/"My Files"/Comics/"Man of Steel.txt" /tmp/"My Files"/Movies/Superheroes/"Man of Steel_copy.txt"

To add text:

echo "Up up and away!"> /tmp/"My Files"/Movies/Superheroes/"Man of Steel_copy.txt"

Section c

1. **Create a hard link of “Man of Steel.txt” and a symbolic link of “Wonder Woman.txt”.**

Using absolute paths:

ln /tmp/”My Files”/Comics/”Man of Steel.txt” /tmp/”My Files”/Comics/”Man of Steel_hard.txt”

ln /tmp/”My Files”/Comics/”Wonder Woman.txt” /tmp/”My Files”/Movies/Superheroes/”Wonder Woman_soft.txt”

2. **Modify the contents of “Man of Steel_copy.txt” and check if the contents of “Man of Steel.txt” and “Man of Steel_hard.txt are modified. Why?**

The contents of “Man of Steel.txt” and “Man of steel_hard.txt” are not modified. “Man of Steel_copy.txt” is a copy of “Man of Steel.txt”, but they don’t point to the same inode, they are independent files, each of them being stored in different blocks. So, when “Man od Steel_copy.txt” is modified, the blocks where the other “Man of Steel” files are stored are not modified.

3. **If deleting “Man of Steel.txt”, what will happen to “Man of Steel_hard.txt” and Man of Steel_copy.txt”?**

Nothing. “Man of Steel_hard.txt” will have the same content that “Man of Steel.txt” had, and “Man of Steel_copy.txt” will have the text previously modified.

4. **Delete the Comics directory. What happens to “Wonder Woman_soft.txt”?**

The soft link is broken. Its name appears in red, and when trying to show its content, there is an error message saying: “the file or the directory does not exist”.

This happens because the soft link is a path to the original file. If the original file is deleted, the soft link is broken, because it is not a copy pointing to the same inode as the hard link, it just contains the path to the inode. If the inode does not contain information anymore, the soft link is useless and it’s broken.

EXERCISE 2

Section a

Create the users specified with the proper requirements:

The next list of commands will create all the users, with a home directory (specified by the "*-m*") and the shell /bin/sh (specified with the *"-s /bin/sh*" command).

sudo useradd -m -s /bin/sh gru

sudo useradd -m -s /bin/sh kevin

sudo useradd -m -s /bin/sh stuart

sudo useradd -m -s /bin/sh nefario

sudo useradd -m -s /bin/sh agnes

sudo useradd -m -s /bin/sh supermegavillain

To create a password for each user, these commands are used:

sudo passwd gru

* It asks for the new password: gru_password

sudo passwd kevin

* It asks for the new password: kevin_password

sudo passwd stuart

* It asks for the new password: stuart_password

sudo passwd nefario

* It asks for the new password: nefario_password

sudo passwd agnes

* It asks for the new password: agnes_password

sudo passwd supermegavillain

* It asks for the new password: supermegavillain_password

Since the exercise specified to create the users in a non-interactive way, the command useradd was selected to create the users. It allows us to create the users quicker, by specifying the options (home directory, shell, groups, others) in a single command, and we don't need to fill in extra information for each user. With this command, we only need to create the user with "useradd" and then use an extra command, "passwd user_name", to assign a password to the user. Using "adduser" would take more time and would require more input from us. The password would be created, but we should manually change the shell for each user using another command.

Section b

To create the groups, these **commands** are used:

sudo groupadd masteroftheuniverse

sudo groupadd minions

sudo groupadd kids

sudo groupadd researchanddevelopment

The association of users to groups could be:

Stuart and kevin → minions

agnes → kids

nefario → researchanddevelopment, kids

Gru → researchanddevelopment, minions, masteroftheuniverse

supermegavillain → masteroftheuniverse

The **commands** to add these users to each group are:

sudo adduser stuart minions

sudo adduser kevin minions

sudo adduser agnes kids

sudo adduser nefario kids

sudo adduser nefario researchanddevelopment

sudo adduser gru researchanddevelopment

sudo adduser gru minions

sudo adduser gru masteroftheuniverse

sudo adduser supermegavillain masteroftheuniverse

The persmissions to assign are these ones:

rwx r-x --- agnes kids operation_birthday/

rwx r-x --- nefario researchanddevelopment Science/

rwx r-x --- kevin minions /science/bananas

rwx r-x --- gru masteroftheuniverse evilplans/

The **commands** to assign the ownership and the permissions are:

sudo chown agnes operation_birthday/

sudo chown nefario science/

sudo chown kevin /science/bananas

sudo chown gru evilplans/

sudo chgrp -R kids operation_birthday/

sudo chgrp -R researchanddevelopment science/

sudo chgrp -R minions /science/bananas

sudo chgrp -R masteroftheuniverse evilplans/

chmod u+rw, g+rx, o- operation_birthday/

chmod u+rw, g+rx, o- science/

chmod u+rwx, g+rx, o- /science/bananas

chmod u+rwx, g+rx, o- evilplans/

These commands assign a user to each folder, that will have certain permissions, and then the users are added to different groups, to be able to have different permissions to other folders, as specified in the exercise.

Exercise 3

The backup will need to be done by a user who has, at least, read and execute permissions on the /user/important_data folder, so the user can access the folder and copy its content.

Assuming that the user is not the owner of the folder, and that the user does not have the right permissions, these commands should be run:

sudo chown user_name /user/important_data

sudo chmod u+rw -R /user/important_data

To copy the folder into different hard disks, whether one hard disk or more than one, these first need to be mounted on the system. To have them mounted at boot time every day, the *fstab* file needs to be modified.

Now let's assume that we have two SATA disks, and each day we want to copy the folder in one of these, alternating the disks everyday. First, the hard disks would need to be included in the fstab file with these commands/steps:

1. Create the folder where the hard disk will be mounted (/home/disk1 and /home/disk2):

cd *(to go to the home directory)

mkdir disk1 disk2

2. Get the UUID of the hard disks:

blkid

3. Edit the fstab file:

sudo nano /etc/fstab

4. For hard disk 1:

Include the next sentence in the fstab content:

UUID (disk 1) */home/disk1 ext4 auto 0 0*

5. For hard disk 2:

Include the next sentence in the fstab content:

UUID (disk 2) */home/disk2 ext4 auto 0 0*

6. Save the modifications in the fstab file and close it.

To copy the folder into the hard disks preserving the permissions, timestamps, ownership and without creating historical data, we could use the "rsync" command. The next steps will copy the folder in the hard disk 1, but the next day these steps need to be repeated for the hard disk 2, and the same everyday alternatively between disk 1 and disk 2.

1. Install rsync in our system:

sudo apt install rsync

2. Copy the folder "important_data" into disk1:

rsync -a --delete /user/important_data /home/disk1

The *-a* attribute in rsync preserves the timestamp, ownership and permissions. The *--delete* attribute keeps the source folder and the destination folder in synch, so no historical data is created. The content of "important_data" is copied exactly as it is on the day of the backup. If there are extra files in the destination file (hard disk) that are no longer in the "important_data" folder, these files will be deleted in the hard disk, to keep both folders synchronized.

Everyday, to create the backup, the user will need to run the rsync command:

rsync -a --delete /user/important_data /home/disk1 *(or disk2)