CHAPTER III: OPENING AND CLOSING SESSION

I/ Introduction:

In this chapter, we'll see how the Linux system manages groups and users, and the commands needed to operate on both concepts.

Unix user = accounts + groups

An account allows you to use a computer → Log in, work, log out

There are 3 types:

Administrator: only one possible, qualified as superuser, his account name is root. Some systems prevent direct connection to this account, as it has full rights.

System accounts: associated with certain services, e.g., printer, network... Generally, not connectable

Ordinary users

Note*:* the Unix system is case-sensitive. Most Unix commands are written in lower case.

II/ sudo command

sudo (short for substitute user do) is a command allowing the system administrator to grant certain users (or groups of users) the ability to run a command as administrator.

sudo is used on the command line, in a terminal. It gives root access to execute a command.

The password requested is that of the user who entered **sudo**. The command will be executed if the password entered is correct and the current user can perform administrative tasks.

```
sudo ls /root
[sudo] password for user
```

III/ Groups management

- ✓ Accounts are placed in groups
  - Example: students, teachers...
  - Example: system-related groups: bluetooth
- ✓ An account can belong to several groups
- ✓ Belonging to a group can give rights to files and directories.
  - For example, if you're in the cdrom group, you can use /dev/cdrom

1/ Files containing user information:

Files	Description
`/etc/passwd`	Users accounts information
`/etc/shadow`	hidden user account information
`/etc/group`	Defines the groups to which users belong
`/etc/gshadow`	Hidden groups infrmation
`/etc/sudoers`	List of who can run what with **sudo**
`/home/*`	Users directories

- Display users list:

```
cat /etc/passwd
```

- Display groups list:

```
cat /etc/group
```

Groups are listed in /etc/group, format: **name:x:no:users**

- Group name
- Group password: x = none
- Group number called GID
- GID > 999 for normal users
- List of participating users

2/ Creating a new group:

addgroup is a command that add a new group

```
sudo addgroup nom_gr
```

- The **--gid GID** option is used to force the group identifier to a certain value.
- Adding the **--system** option creates a system group

```
sudo addgroup –gid 1001 engineer
```

Note that there is also the less complete **groupadd** command

3/ group modification:

We can change the group name:

groupmod -g GID group

```
sudo groupmod -g 1002 section
```

Note that changing the GID does not update the files and folders assigned to the group => there will be an anomaly, as these files will have no known group.

4/ group deleting:

delgroup is a command that delete a specific group

delgroup *nom*

```
sudo delgroup section
```

The command is refused if there are users for whom this is the main group.

IV/ Users management

An account is characterized by :

- A logname, e.g. Ing
- A password
- A unique number called UID, e.g. 1001
- A home dir, e.g. /home/Ing, which contains directories.
- A main group identified by the GID, e.g. admin, and secondary groups, e.g. cdrom, lpr, sudoer...
- A shell to launch, e.g. /bin/bash
- Other information called GECOS: plain name, e-mail address...

1/ users' information:

Accounts are defined in the **/etc/passwd** file

- Format: **login:x:UID:GID:infos:homedir:shell**

Passwords are stored in **/etc/shadow**

```
cat /etc/shadow
```

- This file is protected, only root can see it
- Format: **login:mdp:date:min:max**

PAM authentication

"Pluggable Authentication Modules" store passwords in encrypted form in **/etc/shadow**

Ex: 6n4kRE2Hc$e09e6TO0xUCgHzqgv0Zm1c

They cannot be decrypted:

- use of non-reversible algorithms (a kind of projection)

When you log on, the PAM system encrypts the word you type and compares it with what's in **/etc/shadow: equality => OK**

Password hacking

To "crack" a password, we use brute force:

- Review of all dictionaries of words, first names, dates (and their anagrams)

2/ create a new account:

adduser is a command that ass a new user

adduser --ingroup groupe nom

```
sudo adduser --ingroup section ing
```

It requests additional information, some features can be forced

- --uid UID
- --shell shell, ex : --shell /bin/false
- --home homedir

3/ Viewing account information:

The id command displays various information

id option user

its options are

- **-g**: Displays only the actual group ID.
- **-G**: Prints all group IDs.
- **-n** : Displays the name instead of the number.
- **-r** : Displays actual ID instead of numbers.
- **-u**: Prints only the actual user ID.
- **-help**: Displays help messages and quits.
- **-version**: Displays version information and exits.

```
id -g ing
```

4/ Account modification:

The **usermod** command changes what you want with the options :

- **-C** We can add a comment field for the UserAccount

- **-d** Modify the directory for any existing user account
- **-e** Make the account expire within a specific time period
- **-g** Change the main group for a user
- **-G** Add an additional group
- **-a** Add anyone from the group to a secondary group
- **-l** Lock a user's password.
- **-L** Lock user account. This will lock the password so we can't use the account
- **-m** Move contents of home directory
- **-p** Use an unencrypted password for the new password (insecure).
- **-s** Create a specified shell for new accounts.
- **-u** Use UID assigned to user account between 0 and 999.
- **-U** Unlock user accounts.

Every user has a main group, the one listed in **/etc/passwd**, it can be added to other, so-called secondary, groups:

usermod -G grpe2,grpe3... utilisateur

```
sudo usermod -G section1 section2 ing
```

5/ Password change:

The **passwd** command is used to change the password

- **passwd** (no parameters): changes for the user issuing the command
- **sudo passwd nomuti**: changes this user's password

```
sudo passwd ing
```

6/ Deleting account:

The **deluser** command is used to delete an account:

deluser --remove-all-files nom

Without the option, the command does not delete files owned by this user

```
sudo deluser ing
```

7/ Account login:

You can log in to any account (except root on Debian) during system startup.

Once logged in, you can change account using the **su** command

su : (without parameters) logs in as root

- **su user**: log in as this user, but remain in the current folder

- **su - user**: log in and go to this user's account

8/ Find out which account:

- The **whoami** command displays the name of the account in which you are working.
- The **who** command lists users connected to the same system

V/ Users and files

A user = { a logname and groups }

A file = { an owner (creator) and a group (the owner's group, but not always) }

Operating system have an algorithm to determine who is this user % this file:

- If the user = the owner, then it's in category **U** (misnamed, user instead of owner)
- Otherwise, if one of the user's groups = that of the file, then it's in category **G** (group)
- Otherwise, if neither the owner nor the user's group **O**

Rights:

- If the user wishes in any way to consult what's inside the file, he needs the **R (read)** right.
- If they wish to modify the contents in any way, they need the **W (write)** right
- If it attempts to execute the contents as a program, it needs the **X (eXecute)** right.

Each file has an array of 3x3 Booleans that say:

- **U** has such and such rights **{ R, W, X }**
- **G** has such and such rights **{ R, W, X }**
- **O** has these rights among **{ R, W, X }**

	R	**W**	**X**
U	Yes	Yes	Yes
G	Yes	No	Yes
O	No	No	Yes

Example:

The file exp2.txt belongs to ing, and is placed in the section1 group.

Exp2.txt	**R**	**W**	**X**
U	Yes	Yes	Yes
G	Yes	No	Yes

O	No	No	Yes

Rights viewing

The **ls -l** command displays the rights table just after the file type

- 3 triplets of 3 letters, for **U**, for **G** and for **O**

1 letter = right granted**,** **-** = right denied

- File owner and group name

Example:

```
rwxrwxr-x   5 leila admin  4096 nov. 25 bin
-rw-r--r--  1 leila admin  223  août 15 infos
-rwxr-x--x  2 leila Ing    4096 nov. 13 chk
```

VI/ Change the owner of a file

The main command for changing file ownership is **chown**. It allows users to change user and group ownership for both files and directories.

chown [OPTION] owner[:GROUPE] FICHIER(s)

```
sudo chown -R ing:section /cours/chap3.txt
```