Lab 4

User Management

4.1 Objectives

- Create new user accounts and change their properties
- Disable and safely remove user accounts

4.2 Before you start

Answer the following questions:

In which files are defined users, passwords, and group databases?

How UID (user identifiers) can be mapped for new users?

Which commands can be used to change the owners and permissions of a file? And of all the files in a directory?

4.3 Introduction

At the system each user has an account. An account consist of all the files, resources and information pertaining to each user. User accounts allow the system to differentiate each user's data and processes and allow users to protect their information. For the kernel, users are identified by an integer known as user ID (user identifier or UID). Apart, there is a database associating the UID with a textual name: username. This is the username that the user employs to login. The database includes other information concerning the user like the directory path, the user's full name and the command interpreter (shell).

Creating a new user includes assigning a UID and the modification of the database users to assign their own user settings. Apart, you need at least one group associated with the user and finally to copy configuration file and customization to each user's home directory. Optionally the user can be assigned to multiple groups, allowing system administrator to divide users into groups with different permissions and privileges. This way we can maintain better control over what users can do.

4.4 Profile and user environment

After an interactive login, the shell automatically executes one or more predefined files. Each shell uses different files. The BASH shell runs first the general configuration file */etc/bash_profile* and then run the user configuration files *~/.bash_profile*. The */etc/bash_profile* allows the system administrator define a common environment for all users, particularly by defining the variable *PATH*. On the other hand *~/.bash_profile* allows each user to define its own environment adapting the PATH variable, prompt, etc.

When you create the home directory of a user you must copy the files in */etc/skel*. The system administrator can put files in */etc/skel* to provide a basic environment for users. For example, the administrator can create a */etc/skel/bash_profile* with some basic definitions that, later on, the user may change.

To make life easier for users to modify the file *~/.bash_profile* that there are at */etc/skel*. Verify that the PATH of all users contains the */usr/local/bin* directory, if not, add it at the end of the file. Exporting variables is done through `export` built-in bash command in the following form:

```
export PATH=$PATH:/usr/local/bin
```

? *Analyze the above command and explain here how it works*

Also change the prompt of the system in such a way that for the user `aso` it looks like:

```
aso:CWD - date $
```

Where `CWD` stands for Current Working Directory, and `date` stands for the current date in the form of: *day/month hour:minute*.

? *What are the changes you applied to the PATH variable?*

? *In which environment variable the prompt is defined?*

? *What are the changes that you applied to the prompt variable?*

4.5 Creating users by hand

Now create an user account for each member of your laboratory group. Before doing it define the parameters for each account. Those users should be in the `admin` group.

Parameters	**User 1**	**User 2**
UID		
Username		
HOME directory		
Shell		
Groups		

Table 4.1: Parameters for new users

Edit the user database in order to add the new users. User the `vipw` command to edit this file.

What is the difference between using the `vipw` command and edit directly the database with a text editor like `vi`?

Suggestion: Open two `vipw` sessions at the same time and see what happens!

In a similar way, use the `vigr` to edit the group database in order to create and modify groups. Anothher variant is `vipw -s`, which allows editing the */etc/shadow* file.

What groups you have created?

Which users are part of that groups?

For security reasons it is better to deactivate the user account until all the creating process has finished.

How you can deactivate an user account, in such a way that the user can not make login?

Deactivate the new accounts that you are creating until you finish all the process.

Create the `home` directory for each user. Copy all the files from the */etc/skel* directory, and assign the correct owner, group and permissions to the new `home` directories.

What commands and parameters you have used to change the owner of the `home` directory

What commands you have used to change the file permissions?

Now, assign a password to the new users. For security reasons the password is not written directly in the user database- Instead there is another file, called shadow */etc/shadow*, for putting the password and some parameters related to it.

What are the security risks associated by putting the password in the user database?

Which command can be used to edit safely the shadow file?

What is the meaning of the password parameters defined in the /etc/shadow file?

what command can be used to modify those password parameters?

To edit the user account and other parameters you can use the commands `chfn` and `chsh`. Use these commands to assign appropriate values to the accounts you have created.

4.6 Automatic creation of users

Most Linux distributions include software to automate the task of user account creation and modification. Some of these tools are `useradd` and `adduser`. They allow to create and assign various parameters required for new user accounts.

Use these commands to register the following accounts:

- Professors: emorancho, rserral
- Students: student
- Other users: four accounts for four different groups for the labs. The username of these accounts will be: asoXX. Where XX is a two digit identifier of a lab group.

Choose and justify the most appropriate username for all users. Analyze that some users may require usernames and directories.

The permissions for each of these user groups (teachers, students, administrators, and other groups as necessary) are defined as follows:

- Teachers will have access control at group level to all files of all defined users.
- Students will have access control at group level to all files of all users, except of teachers.
- Administrators have access control at group level only to the files of their own group.
- Other users (asoXX) will have not access control at group level to any file of any group.

Note that the above conditions specify only access levels. So, you don't need to modify the protections of the directories or files once set.

How can you configure the directory permissions to inherit to the created files within the directory automatically?

Hint: this implies directory permissions at group level, and also setting the proper `umask` variable.

4.7 Removing and disabling users

In order to remove a user account it is necessary to delete all files in the mailboxes, print jobs, `cron` and `at` jobs and all references to the user. After that you can delete the lines associated with the user and group databases. As a user may have files outside their home directory you need to search the entire directory tree for files belonging to the user and remove them. Before removing the files a good practice is to make a backup of them.

Now we want to delete one user account, but first we need to make backup of all the files and then delete them.

Which command(s) you can use to make the backup of all the files of a user?

Hint: `xargs` may be useful.
`xargs` allows you to use the standard output of a command as parameters for the command specified as parameter. This is specially useful in pipes and when having really long list of parameters.

What's wrong with the files that have spaces in their name? How you can resolve this? (Hint: see options of `xargs` command or the `-exec` option of `find`)

Which command you can use for searching all files of a user and delete them?

It's a good security practice to disable the user account before deleting files and other steps during the account removal.

One way to disable an account is to invalidate the password by changing the shell of the user for a simple program. This program just writes a message giving information to the user of the reasons that its user account has been deactivated. This program is called a tail script and looks like:

```
#!/usr/bin/tail -n 2
This account has been closed due to a security problem. Please
contact the system administrator.
```

Try to understand how this tail script works. It'll help to understand the UNIX way of achieving things

This script can be assigned as a shell to a user employing the command `chsh` and it can be stored in a separate directory like `/usr/local/lib/no-login`.

Use the `chsh` command to assign a tail script in order to deactivate the account of one of the users asoXX.

How you can check that the account has been deactivated?

Now create a script that given a username makes a backup pf all the files of the user, remove all the files of that user, and finally changes the shell for a tail script.

What is the content of this script

4.8 Special users

Some UNIX commands, like `shutdown` used to turn off the machine, can only be executed by the `root` user. In many cases, however, it is necessary that other users can also turn off the machine, but without having access to root privileges. To achieve this, you are asked to create an account used to run a special simplified shell that will allow to run `shutdown` and other special commands that require superuser permissions.

Create a new user with `asosh` as username. Set an appropriate password.

When someone makes a login for this account, it is going to run the `asosh` shell that you installed in laboratory 2 about application management. For security reasons you should make sure when this user make login it does not execute any shell, apart from the restricted shell `asosh`.

What are the required set of permissions for this application in order to avoid a direct execution by any user?

What is the appropriate values for the entry in the user database for `asosh`?

4.9 Sudo and control of application execution

Like `shutdown`, there are other management commands that can only be executed by the root user. It is a bad security practice to use the root account to execute these commands. To solve this is problem you can use the `sudo` command. `sudo` allows a user to execute a command with root permissions. The configuration of what applications can execute a particular user is defined in the */etc/sudoers* (See `man sudoers`. This file can be edited safely using the command `visudo`.

Make the required changes in the *sudoers* configuration in order to allow the member of the admin group can execute all commands as superusers. Additionally make the necessary changes so that users in the teachers group can execute the script to delete users created in a previous section and all the binaries in the */usr/local/teachers/bin*. Verify that it works by running with `vipw` command.

What changes are required in the /etc/sudoers file to enable the configuration described above?

Finally disable the root account so you can not login as root. The administration commands should be executed only from the users in the admin group using the `sudo` command

Make sure you can execute administrative commands before disabling the root account

What are the steps required to disable the root account?

More information about user management can be found in [1] and [2].

Bibliography

[1] L. Wirzenius, J. Oja, S. Stafford, and A. Weeks, *The Linux System Administrators' Guide, version 0.9*. The Linux Documentation Project. TLDP. [Online]. Available: http://www.tldp.org/LDP/sag/sag.pdf

[2] E. Nemeth, G. Snyder, T. R. Hein, B. Whaley, and D. Mackin, *UNIX and Linux System Administration Handbook (5th Edition)*, 5th ed. Addison-Wesley Professional, 2017.