Managing users and Groups

Managing User Accounts

- Managing user accounts involves adding, modifying and deleting user accounts and account's information
- To add user accounts we use the `useradd` or `adduser` command.

  In Ubuntu, the `adduser` program is recommended over `useradd` due to useradd being a low-level utility.
- To modify user's information we use the `usermod` program.
- To delete a users we use the `userdel` program.
- The following files are involved in the user creation process:
  - `/etc/login.defs` `/etc/default/useradd` `/etc/skel/`
  - `/etc/passwd` `/etc/shadow` `/etc/group`

Managing User Accounts

- How do I add a user in Ubuntu?
  - Run the command `adduser` followed by the username.

The adduser command must be run with superuser privileges

During the account process, you will be asked to choose a password.

During the account process, you can add other details about the user. This is optional and can be modified in the future.

```
adrian@server-inspiron:~$ sudo adduser ralberto
[sudo] password for adrian:
Adding user `ralberto' ...
Adding new group `ralberto' (1001) ...
Adding new user `ralberto' (1001) with group `ralberto' ...
Creating home directory `/home/ralberto' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for ralberto
Enter the new value, or press ENTER for the default
        Full Name []: rob
        Room Number []: 101
        Work Phone []: 999-999-9900
        Home Phone []: 999-999-9900
        Other []:
Is the information correct? [Y/n] Y
adrian@server-inspiron:~$
```

Managing User Accounts

- How do I delete a user in Ubuntu?
  - To delete a user use the userdel command followed by the username.
  - By default the userdel command does not delete the user's home directory.
  - You need to pass the -r option for the command to delete the user and its home directory.

Notice that if ran without superuser privileges, the userdel command will return an error.

```
adrian@server-inspiron:~$ userdel -r ralberto
userdel: Permission denied.
userdel: cannot lock /etc/passwd; try again later.
adrian@server-inspiron:~$ sudo userdel -r ralberto
[sudo] password for adrian:
userdel: ralberto mail spool (/var/mail/ralberto) not found
adrian@server-inspiron:~$
```

This mail spool error is irrelevant because at the time of creating the user, this directive was never created.

Managing User Accounts

- The management of users in all linux system is governed by multiple files. These files dictate how the users will be created and what configuration applies to these users.
- Understanding the purpose of these files is canonical to the understanding of how users and groups work on in Linux.

```
/etc/login.defs
/etc/default/useradd
/etc/skel/
/etc/passwd
/etc/shadow
/etc/group
```

The /etc/login.defs file

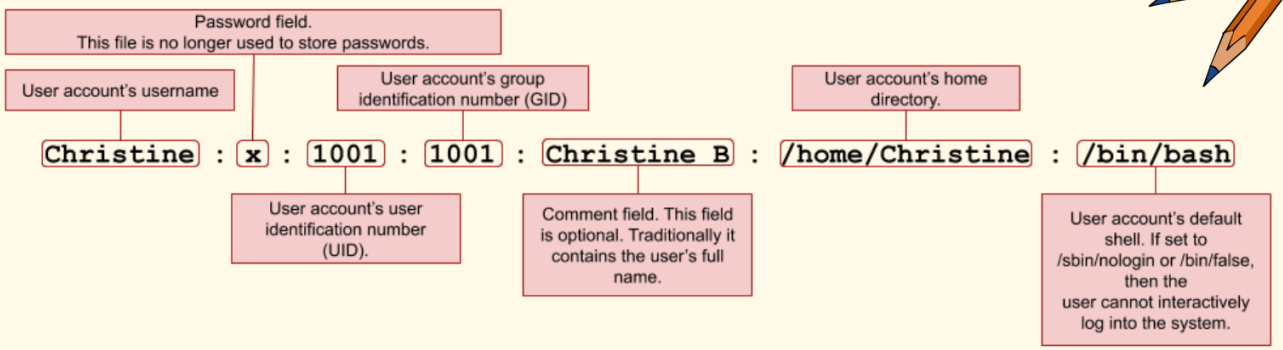
- Let's break down that grep command:

```
grep -ve ^$ /etc/login.defs | grep -v ^#
```

This grep command will suppress all empty lines. Notice that per the man page options v and e will allow us first to invert the string we are looking for ($ representing empty lines) and then use a pattern to search.

This grep command will suppress all comments which are lines that start with the # symbol

The /etc/passwd file



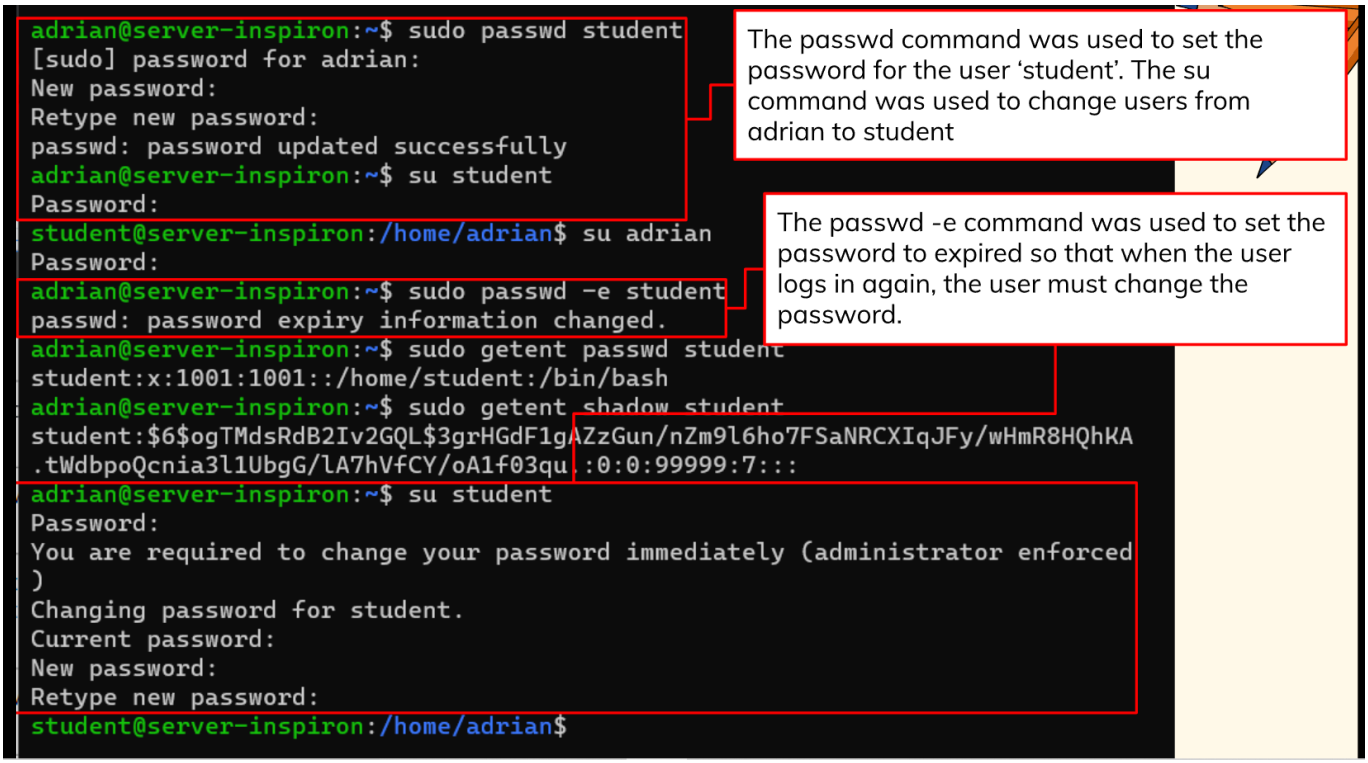
- Field #7 of the passwd file is the user's login shell. There are 3 possible values for this field:
  - A login shell like: /bin/bash, /bin/sh, /bin/zsh
  - /bin/false
  - /sbin/nologin
- When a user with the /bin/false shell tries to log in, they are kicked out of the system.
- When a user with the /sbin/nologin shell tries tog in, a message is displayed and the user is kicked out.
- The message displayed by the /sbin/nologin is stored in `/etc/nologin.txt`

![](../Imgs/Managing%20User%20Accounts7.png

Maintaining Passwords

- The **useradd** utility does not create a password for users. For this case, we use the **passwd** utility.
- The passwd utility can update passwords for any user as well as update the password of the current user.
- To change the password of another user use: **passwd + username**
- To change the password of the current user, use **passwd** with no argument
- The passwd utility can also lock and unlock accounts with the **-l** and **-u** options.

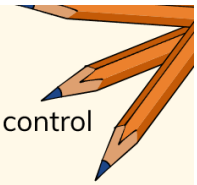
Short	Long	Descriptions
-d	--delete	Removes the account's password.
-e	--expire	Sets an account's password as expired. User is required to change account password at next login.
-i	--inactive	Sets the number of days after a password has expired and has not been changed until the account will be deactivated.
-l	--lock	Places an exclamation point (!) in front of the account's password within the /etc/shadow file, effectively preventing the user from logging into the system via using the account's password.
-n	--minimum	Sets the number of days after a password is changed until the password may be changed again.
-S	--status	Displays the account's password status.
-u	--unlock	Removes a placed exclamation point (!) from the account's password within the /etc/shadow file.
-w	--warning or --warndays	Sets the number of days a warning is issued to the user prior to a password's expiration.
-x	--maximum or --maxdays	Sets the number of days until a password change is required. This is the password's expiration date.



Modifying User Accounts

- Where usermod comes in really handy is in a situation where you've created an account with the useradd utility and need to modify some information about the user that was created.
- Lets see an example:
  - Create a user with useradd in Ubuntu: **`sudo useradd sampleuser`**
  - Let's give the user a home directory: **`sudo usermod -md /home/sampleuser sampleuser`**
  - Let's give the user a password: **`sudo passwd sampleuser`**
  - Login with the new user: **`su sampleuser`**
  - What shell did you login with?
  - Let's change the default login shell.
    - Log out: **`exit`**
    - Change the default shell: **`sudo usermod -s /bin/bash sampleuser`**

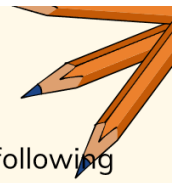
Managing Groups



- Groups are organizational structures that are part of Linux's discretionary access control (DAC).
- DAC is the traditional Linux security control, where access to a file, or any object, is based upon the user's identity and current group membership.
- When a user account is created, it is given membership to a particular group, called the account's default group.

```
adrian@server-inspiron:~$ cat /etc/passwd | grep "adrian"
adrian:x:1000:1000:adrian:/home/adrian:/bin/bash
adrian@server-inspiron:~$ cat /etc/group | grep ^"adrian"
adrian:x:1000:
adrian@server-inspiron:~$
```

Setting Up the Environment



- There are four potential files found in the user's home directory, $HOME, that are environmental files. For a default login or interactive shell, the first file found in the following order is run, and the rest are ignored:
  - `.bash_profile`
  - `.bash_login`
  - `.profile`
- Typically, the fourth file, `.bashrc`, is run from the file found in the preceding list. However, anytime **a noninteractive shell** is started, the .bashrc file is run.
- These individual user environment files are typically populated from the **`/etc/skel/`** directory, depending on your account creation configuration settings.
- For future accounts, you can make changes to the skeleton environment files.

Querying users

There are several utilities used for querying users:

- **Whoami:** Display what user account you are currently using.

```
adrian@server-inspiron:~$ whoami
adrian
adrian@server-inspiron:~$
```

- **Who:** provides a little more data than the whoami utility. You can view information concerning your own account or look at every current user on the system.

```
adrian@server-inspiron:~$ who -aH
NAME       LINE         TIME             IDLE          PID COMMENT  EXIT
           system boot  2020-11-18 00:54
LOGIN      tty1         2020-11-18 00:54               724 id=tty1
           run-level 5  2020-11-18 00:54
adrian   + pts/0        2020-11-18 00:54   .           920 (192.168.1.177)
adrian@server-inspiron:~$
```

Querying users

- **W:** shows who is logged on and what they are doing.

```
adrian@server-inspiron:~$ w
 15:07:51 up  8:27,  1 user,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
adrian   pts/0    192.168.1.177    00:54    2.00s  0.67s  0.00s w
adrian@server-inspiron:~$ w adrian
 15:08:06 up  8:27,  1 user,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
adrian   pts/0    192.168.1.177    00:54    3.00s  0.67s  0.00s w adrian
adrian@server-inspiron:~$
```

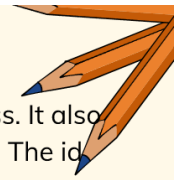
The first line shows:

- The current time
- How long the system has been up
- How many users are currently using the system
- The CPU load

The follow up lines contain:

- The user account name
- The users TTY
- When the user logged in
- For how long has the user being IDLE
- The total CPU time that the account has used
- How much CPU time the current command has the account used
- What command is the account running

Querying users



- **Id:** The id utility allows you to pull out various data concerning the current user process. It also displays information for any account whose identification you pass to id as an argument. The id command provides a nice one-line summary as shown in Listing.

```
adrian@server-inspiron:~$ id adrian
uid=1000(adrian) gid=1000(adrian) groups=1000(adrian),4(adm),24(cdrom),27(su
do),30(dip),46(plugdev),116(lxd),1003(devs)
adrian@server-inspiron:~$
```

- **Last:** The last command pulls information from the /var/log/wtmp file and displays a list of accounts showing the last time they logged in/out of the system or if they are still logged on. It also shows when system reboots occur and when the wtmp file was started.

```
adrian@server-inspiron:~$ last
adrian   tty1                      Wed Nov 18 20:17   still logged in
adrian   pts/1        192.168.1.177    Wed Nov 18 19:35   still logged in
adrian   pts/0        192.168.1.177    Wed Nov 18 00:54    gone - no logout
reboot   system boot  5.4.0-53-generic Wed Nov 18 00:54   still running
adrian   pts/0        192.168.1.177    Sat Nov 14 15:15 - crash (3+09:38)
adrian   pts/0        192.168.1.177    Thu Nov 12 17:10 - 23:00 (1+05:49)
adrian   tty2                      Thu Nov 12 17:04 - crash (5+07:49)
reboot   system boot  5.4.0-53-generic Thu Nov 12 17:04   still running
adrian   tty3                      Thu Nov 12 15:27 - down   (01:36)
adrian   tty1                      Thu Nov 12 15:26 - down   (01:37)
reboot   system boot  5.4.0-53-generic Thu Nov 12 15:25 - 17:04  (01:38)

wtmp begins Thu Nov 12 15:25:49 2020
adrian@server-inspiron:~$
```