

Actividad 6. Ambiente de trabajo y Comandos Linux

Inicio

El propósito de este taller es realizar la instalación y configuración del ambiente de trabajo y conocer los comandos básicos de Linux.

PARTE 1. Ambiente de trabajo (tomado de las notas de clase del profesor Oscar Mondragón)

El ambiente de trabajo consta de las siguientes herramientas: **Vagrant** + **VirtualBox** + **Ubuntu**. Veremos a continuación como manejar cada una de ellas.

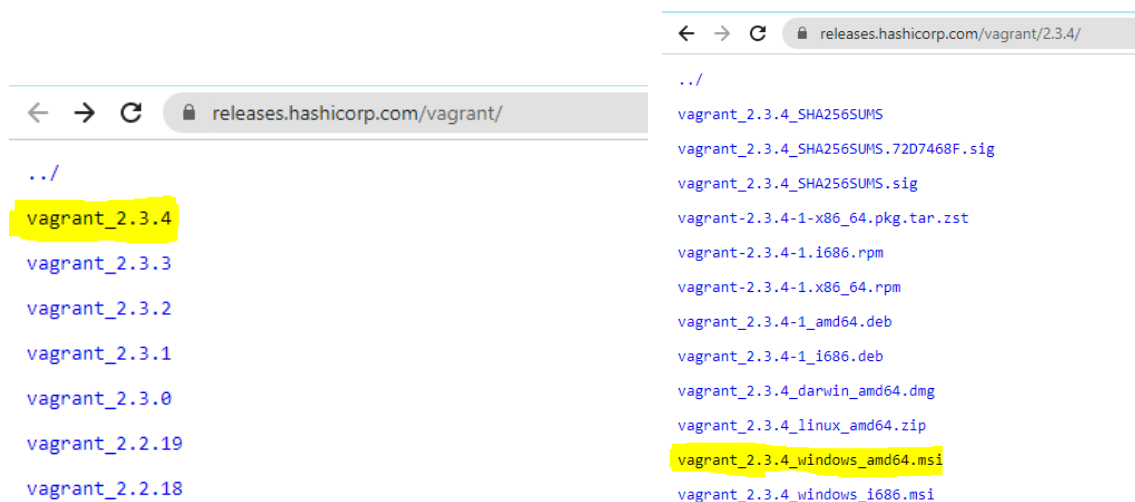
Para no tener problemas con este ambiente de trabajo se recomienda desactivar las actualizaciones automáticas de Windows.

I. Instalación de VirtualBox.

Descargar e instalar la última versión de VirtualBox. La descarga la puede hacer desde el siguiente enlace: <https://www.virtualbox.org/wiki/Downloads>. La instalación no requiere ninguna configuración especial así que se puede hacer con todos los valores por defecto.

II. Instalación de Vagrant

Descargar e instalar la última versión de Vagrant, la descarga la puede hacer desde el siguiente enlace: <https://releases.hashicorp.com/vagrant/>



En la consola de Windows se puede verificar la versión de Vagrant que se instaló:

```
Símbolo del sistema
Microsoft Windows [Versión 10.0.15063]
(c) 2017 Microsoft Corporation. Todos los derechos reservados.

U:\>vagrant version
Installed Version: 2.3.4
Latest Version: 2.3.4

You're running an up-to-date version of Vagrant!

U:\>
```

Instalar el plugin **vbguest** para vagrant, con el fin de mantener las adiciones de los guest de VirtualBox actualizados. Estas adiciones (**Guest Additions**) son un paquete de software que forma parte de VirtualBox y añade funcionalidades a la instalación básica de VirtualBox que mejoran su rendimiento y consiguen un mejor nivel de integración entre la máquina huésped y la máquina anfitriona.

```
U:\>vagrant plugin install vagrant-vbguest
Installing the 'vagrant-vbguest' plugin. This can take a few minutes...
Installed the plugin 'vagrant-vbguest (0.31.0)'!

U:\>
```

III. Configuración y creación de las máquinas virtuales

Muchos de los servicios que trabajaremos en este módulo usan el modelo cliente/servidor, por tanto configuraremos 2 máquinas virtuales, una se comportará como servidor y la otra como cliente.

El proceso de configuración es el siguiente:

1. Cree un directorio y le da un nombre, en este ejemplo se llamará prueba.
2. Ingrese al directorio y desde la consola de Windows ejecute el siguiente comando **vagrant init** con lo cual se crea un archivo de configuración llamado **Vagrantfile**.

```
U:\>c:
C:\>mkdir prueba
C:\>cd prueba
C:\prueba>vagrant init
A `Vagrantfile` has been placed in this directory. You are now
ready to `vagrant up` your first virtual environment! Please read
the comments in the Vagrantfile as well as documentation on
`vagrantup.com` for more information on using Vagrant.

C:\prueba>
```

El archivo Vagrantfile contiene la información básica para la creación de las dos máquinas virtuales.

El contenido de Vagrantfile debe ser el siguiente:

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do |config|

  if Vagrant.has_plugin? "vagrant-vbguest"
    config.vbguest.no_install = true
    config.vbguest.auto_update = false
    config.vbguest.no_remote = true
  end

  config.vm.define :clienteUbuntu do |clienteUbuntu|
    clienteUbuntu.vm.box = "bento/ubuntu-22.04"
    clienteUbuntu.vm.network :private_network, ip: "192.168.100.3"
    clienteUbuntu.vm.hostname = "clienteUbuntu"
    clienteUbuntu.vm.box_download_insecure=true
  end

  config.vm.define :servidorUbuntu do |servidorUbuntu|
    servidorUbuntu.vm.box = "bento/ubuntu-22.04"
    servidorUbuntu.vm.network :private_network, ip: "192.168.100.2"
    servidorUbuntu.vm.hostname = "servidorUbuntu"
    servidorUbuntu.vm.box_download_insecure=true
  end
end
```

Este Vagrantfile define dos máquinas virtuales, una llamada **servidorUbuntu** con dirección ip 192.168.100.2 y la otra llamada **clienteUbuntu** con dirección ip 192.168.100.3, ambas instanciadas desde un box en el repositorio de bento llamado bento/Ubuntu-20.04.

3. Crear las máquinas virtuales mediante el comando `vagrant up` desde la consola de Windows.

```
C:\prueba>vagrant up
Bringing machine 'clienteUbuntu' up with 'virtualbox' provider...
Bringing machine 'servidorUbuntu' up with 'virtualbox' provider...
==> clienteUbuntu: Importing base box 'bento/ubuntu-22.04'...
==> clienteUbuntu: Matching MAC address for NAT networking...
==> clienteUbuntu: Checking if box 'bento/ubuntu-22.04' version '202212.11.0' is up to date...
==> clienteUbuntu: Setting the name of the VM: prueba_clienteUbuntu_1675890604175_35397
==> clienteUbuntu: Vagrant has detected a configuration issue which exposes a
==> clienteUbuntu: vulnerability with the installed version of VirtualBox. The
==> clienteUbuntu: current guest is configured to use an E1000 NIC type for a
==> clienteUbuntu: network adapter which is vulnerable in this version of VirtualBox.
==> clienteUbuntu: Ensure the guest is trusted to use this configuration or update
==> clienteUbuntu: the NIC type using one of the methods below:
==> clienteUbuntu:   https://www.vagrantup.com/docs/virtualbox/configuration.html#default-nic-type
==> clienteUbuntu:   https://www.vagrantup.com/docs/virtualbox/networking.html#virtualbox-nic-type
==> clienteUbuntu: Clearing any previously set network interfaces...
==> clienteUbuntu: Preparing network interfaces based on configuration...
clienteUbuntu: Adapter 1: nat
clienteUbuntu: Adapter 2: hostonly
==> clienteUbuntu: Forwarding ports...
clienteUbuntu: 22 (guest) => 2222 (host) (adapter 1)
==> clienteUbuntu: Booting VM...
==> clienteUbuntu: Waiting for machine to boot. This may take a few minutes...
clienteUbuntu: SSH address: 127.0.0.1:2222
clienteUbuntu: SSH username: vagrant
clienteUbuntu: SSH auth method: private key
```

Verificar el estado de las máquinas con el comando `vagrant status`

```
C:\prueba>vagrant status
Current machine states:

clienteUbuntu          running (virtualbox)
servidorUbuntu         running (virtualbox)

This environment represents multiple VMs. The VMs are all listed
above with their current state. For more information about a specific
VM, run `vagrant status NAME`.
```

4. Conexión a las máquinas virtuales

A través del servicio de **SSH** establecer una conexión con las máquinas virtuales.

Conexión con `servidorUbuntu`:

```
C:\prueba>vagrant ssh servidorUbuntu
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-56-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Thu Feb  9 12:57:29 PM UTC 2023

System load:  0.08544921875      Processes:            102
Usage of /:   11.5% of 30.34GB   Users logged in:     0
Memory usage: 18%               IPv4 address for eth0: 10.0.2.15
Swap usage:   0%                IPv4 address for eth1: 192.168.100.2

This system is built by the Bento project by Chef Software
More information can be found at https://github.com/chef/bento
vagrant@servidorUbuntu:~$
```

Autenticarse como super usuario e instalar herramientas para configuración de red:

```
vagrant@servidorUbuntu:~$ sudo -i
root@servidorUbuntu:~# apt-get install net-tools
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
 net-tools
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 204 kB of archives.
After this operation, 819 kB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu jammy/main amd64 net-tools amd64 1.60+git20181103.0eebece-1ubuntu5 [204 kB]
Fetched 204 kB in 1s (161 kB/s)
Selecting previously unselected package net-tools.
(Reading database ... 44068 files and directories currently installed.)
Preparing to unpack .../net-tools_1.60+git20181103.0eebece-1ubuntu5_amd64.deb ...
Unpacking net-tools (1.60+git20181103.0eebece-1ubuntu5) ...
Setting up net-tools (1.60+git20181103.0eebece-1ubuntu5) ...
Processing triggers for man-db (2.10.2-1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@servidorUbuntu:~#
```

Instalar el editor Vim

```
root@servidorUbuntu:~# apt-get install vim
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
vim is already the newest version (2:8.2.3995-1ubuntu2.1).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
root@servidorUbuntu:~#
```

Repetir los pasos para la maquina clienteUbuntu.

5. Verificación de conectividad entre las máquinas

Con el comando `ifconfig` se consulta la dirección de cada máquina y con el comando `ping` se prueba la conectividad.

The image displays four terminal windows arranged in a 2x2 grid, showing the process of verifying network connectivity between two virtual machines: 'servidorUbuntu' and 'clienteUbuntu'.

- Top Left (servidorUbuntu):** Shows the installation of Vim and the output of the `ifconfig` command. It details the configuration for the `eth0` (ethernet) and `lo` (loopback) interfaces, including IP addresses, netmasks, and broadcast addresses.
- Top Right (clienteUbuntu):** Shows the output of the `ifconfig` command on the client machine, displaying similar interface details for `eth0` and `lo`.
- Bottom Left (servidorUbuntu):** Shows the output of the `ping` command from the server to the client's IP address (192.168.100.3). It displays the results of four successful ping attempts with response times and packet statistics.
- Bottom Right (clienteUbuntu):** Shows the output of the `ping` command from the client to the server's IP address (192.168.100.2). It displays the results of four successful ping attempts with response times and packet statistics.

6. Detener o suspender una máquina virtual.

`vagrant suspend` guarda el estado actual de la maquina y la detiene. Para volver a la máquina desde el punto en que la suspendió puede ejecutar `vagrant up`.

`Vagrant halt` apaga la máquina virtual de manera segura conservando los contenidos del disco y permitiendo un inicio seguro de nuevo. Para levantar la máquina de nuevo puede usar `vagrant up`.

Vagrant destroy remueve la máquina del sistema completamente. Para levantar la máquina de nuevo puede usar vagrant up.

IV. Publicar boxes en Vagrant Cloud

Una vez modificada una máquina virtual se puede empaquetar en un box y subirla a **Vagrant Cloud** para usarla posteriormente. Para esto se deben seguir los siguientes pasos.

1. Empaquetar la máquina virtual en un box de vagrant.

```
C:\prueba>vagrant package servidorUbuntu --output mynew.box
==> servidorUbuntu: Attempting graceful shutdown of VM...
==> servidorUbuntu: Clearing any previously set forwarded ports...
==> servidorUbuntu: Exporting VM...
==> servidorUbuntu: Compressing package to: C:/prueba/mynew.box
```

2. Agregar el box creado a la instalación de vagrant.

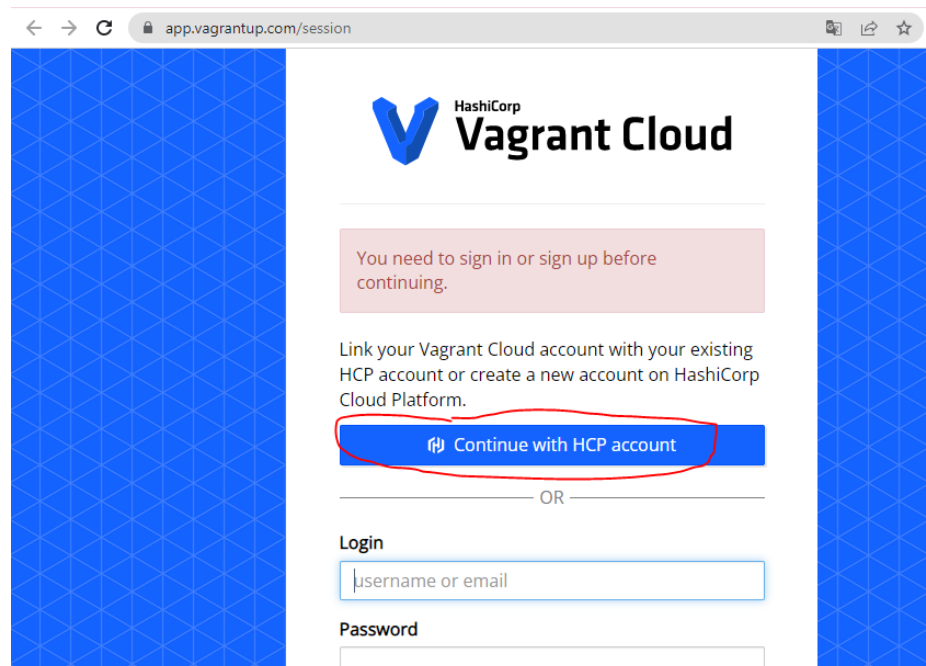
El comando anterior creara un archivo **mynew.box**. Con el siguiente comando agregaremos ese box a nuestra instalación de Vagrant:

```
C:\prueba>vagrant box add mynewbox mynew.box
==> box: Box file was not detected as metadata. Adding it directly...
==> box: Adding box 'mynewbox' (v0) for provider:
    box: Unpacking necessary files from: file://C:/prueba/mynew.box
    box:
==> box: Successfully added box 'mynewbox' (v0) for 'virtualbox'!
```

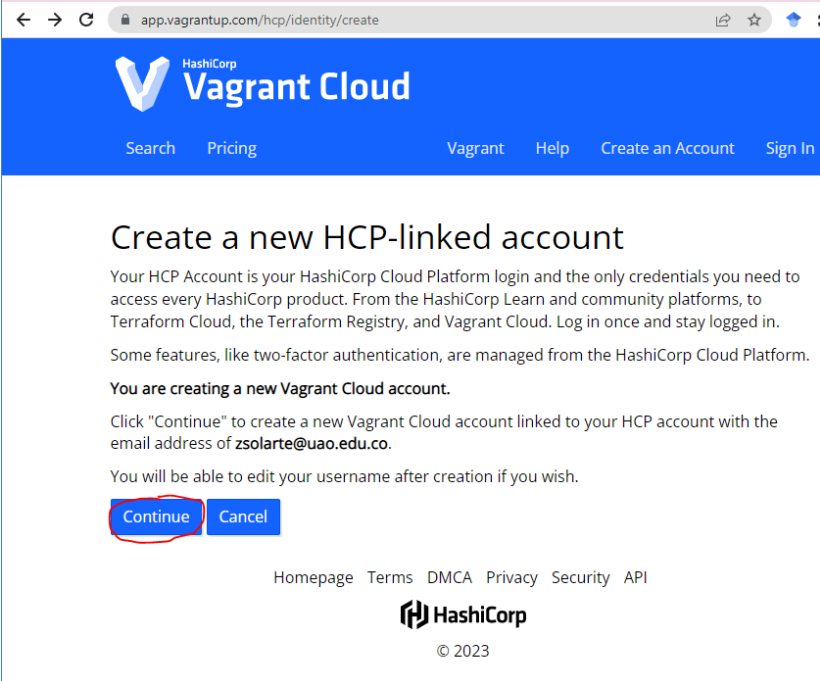
Esto permitirá usar el box desde cualquier ubicación en su computador.

3. Publicar el box en vagrant cloud

Lo primero es ir al link: <https://app.vagrantup.com/boxes/new> si es la primera vez que lo hace y no tiene una cuenta le aparecerá la siguiente página:

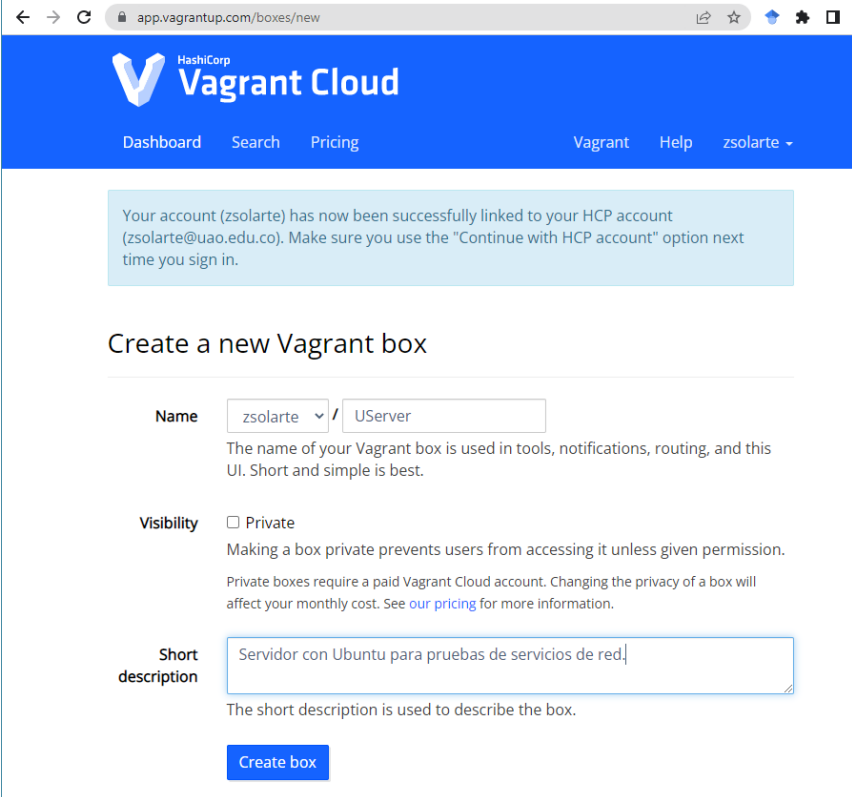


En este punto creamos la cuenta, validamos el correo e ingresamos al link de nuevo. Aquí nos pedirá que vinculemos vagrant cloud con la cuenta creada y le decimos que si.



The screenshot shows the HashiCorp Vagrant Cloud website. The header is blue with the Vagrant Cloud logo and navigation links: Search, Pricing, Vagrant, Help, Create an Account, and Sign In. The main content area is white and titled 'Create a new HCP-linked account'. It explains that an HCP account is needed to access HashiCorp products and that the user is creating a new Vagrant Cloud account. It instructs the user to click 'Continue' to create the account linked to the email address 'zsolarte@uao.edu.co'. A red circle highlights the 'Continue' button. At the bottom, there are links for Homepage, Terms, DMCA, Privacy, Security, and API, followed by the HashiCorp logo and copyright notice for 2023.


Y llegamos a este punto en donde le ingresamos un nombre y una descripción al box que vamos a crear:



The screenshot shows the HashiCorp Vagrant Cloud website. The header is blue with the Vagrant Cloud logo and navigation links: Dashboard, Search, Pricing, Vagrant, Help, and a user profile dropdown for 'zsolarte'. A light blue notification box states: 'Your account (zsolarte) has now been successfully linked to your HCP account (zsolarte@uao.edu.co). Make sure you use the "Continue with HCP account" option next time you sign in.' The main content area is white and titled 'Create a new Vagrant box'. It contains a form with three sections: 'Name' with a dropdown menu set to 'zsolarte' and a text input field containing 'USever'; 'Visibility' with a checkbox for 'Private' and explanatory text; and 'Short description' with a text area containing 'Servidor con Ubuntu para pruebas de servicios de red.'. A 'Create box' button is at the bottom.

Cuando le damos crer el box debemos asignarle una versión al box, esta versión debe cumplir con el formato [0-9].[0-9].[0-9]. Por ejemplo 0.0.1.

← → ↻ app.vagrantup.com/zsolarte/boxes/UServer/versions/new

 HashiCorp **Vagrant Cloud**

Dashboard Search Pricing Vagrant Help zsolarte ▾

Versions **New Version** Settings Access

zsolarte / UServer Vagrant box

New Box Version

Version

The version should be compatible with [RubyGems versioning](#).

Description


The version description functions as release notes. Include any important changes here.

Create version

Después se debe crear un provider para el box (**Virtualbox** es el provider más común), luego se carga el archivo .box que corresponde al provider creado

Una vez cargado el box lo puede encontrar en la sección de boxes de <https://app.vagrantup.com/>. Tenga en cuenta que antes de usar una versión del box, debe liberarlo **“release”**, ya una vez creado y liberado un box, puede liberar nuevas versiones dando click en **“create new version”** en el menú de versiones de la página del box.

← → ↻ app.vagrantup.com/zsolarte/boxes/UServer/versions/0.0.1

 HashiCorp **Vagrant Cloud**

Dashboard Search Pricing Vagrant Help zsolarte ▾

Successfully released

Versions **New Version** Settings Access

zsolarte / UServer Vagrant box

How to use this box with [Vagrant](#):

Vagrantfile [New](#)

```
Vagrant.configure("2") do |config|
  config.vm.box = "zsolarte/UServer"
  config.vm.box_version = "0.0.1"
end
```

v0.0.1 currently released version [Edit](#)

This version was created 12 minutes ago.

Esta versión solo tiene instalada las funciones de red basicas

1 provider for this version. [Add a provider](#)

virtualbox Hosted by Vagrant Cloud (1.11 GB) [Edit](#) ⓘ

Ya con el box libreado en la nube, se puede descargar o acceder a él desde cualquier parte, simplemente configurando el archivo **Vagrantfile**.

NOTA: Investigue en qué consisten los **directorios sincronizados** de Vagrant. Demuestre su funcionamiento.

PARTE 2. Comandos básicos de Linux

Los grupos de comandos que nos interesa recordar son los siguientes: comandos para manejo de archivos y directorios, comandos para gestión de usuarios, comandos para gestión de permisos de archivos.

a) Comandos para el manejo de archivos y directorios

A continuación, se listan algunos comandos básicos de Linux:

cd :	Cambia el directorio
pwd:	print working directory (muestra el directorio actual)
ls:	Lista el contenido de directorios (opción -la para ver archivos ocultos y propiedades)
cp:	Copiar archivos
mv:	Mover archivos o directorios
rm:	Borrar archivos o directorios
mkdir:	Crear directorios
rmdir:	Borrar directorios vacíos
logout:	Salir de un logueo de sesión
cat:	Visualización del contenido de archivos
less:	Visualización del contenido de archivos
more:	Visualización del contenido de archivos
touch:	Crea un archivo vacío

Los comandos en Linux usan el siguiente formato:

nombre-comando -opciones lista de parámetros

nombre-comando: el nombre del comando actual, generalmente es el nombre del programa ejecutable que implementa el comando.

-opciones: el símbolo **-** se usa para indicar una opción. La opción modifica el modo de operar del comando.

lista de parámetros: La lista de parámetros (o argumentos) con los que operará el comando podrán ser 0 1 o más parámetros. Los parámetros son separados por espacios en blanco.

Desarrolle los siguientes ejercicios:

1. Visualice el tamaño de los archivos que se encuentran en el directorio **/root** utilizando el comando **ls**. Nota: debe listar también los archivos ocultos.
2. Ejecutar los siguientes comandos:

ls

ls /

¿Cuál fue la diferencia?

3. Ejecutar los siguientes comandos e indicar cuál fue el resultado en cada caso.

```
cd
cd..
cd ..
cd /etc
cd
```

4. Crear el directorio **prueba1** en /root

5. Copiar algún archivo de configuración del sistema (/etc) en el directorio de usuario.
Renombrarlo a **datos**.

6. Copiar el archivo en el mismo directorio con el nombre **datos2** y verificar que el contenido sea el mismo.

7. Crear el directorio **prueba2** y copiar dentro el archivo **datos**.

8. Mover el archivo **datos2** al directorio **prueba2**.

9. Mover el archivo **datos** al directorio **prueba2** con el nombre **datos3**.

10. Cambiarle el nombre al archivo **datos3** por **datos**.

11. Mover todos los archivos (de una sola vez) que empiecen con el nombre **datos** del directorio **prueba2** al directorio **prueba3**.

12. Borrar el archivo **datos3**.

b) Comandos para la creación de Usuarios y Grupos

La administración de usuarios y grupos solamente la puede hacer el usuario root. Los comandos asociados son los siguientes:

useradd:	creación de usuarios
usermod:	modificación de usuarios
userdel:	eliminación de usuarios
groupadd:	creación de grupos
groupmod:	modificación de grupos
groupdel:	eliminación de grupos
adduser:	añadir usuarios a un grupo
deluser:	quitar usuarios de un grupo

La sintaxis del comando useradd es la siguiente:

useradd [opciones] nombre-usuario

en donde las opciones pueden ser las siguientes:

- g:** Grupo principal al que pertenecerá el usuario (el grupo debe existir)
- d:** Carpeta home del usuario. Suele ser /home/nombre-usuario
- m:** Crear carpeta home si no existe
- s:** interprete de comandos (Shell) del usuario. Suele ser /bin/bash

Si no se especifica ninguna opción, automáticamente se crea un grupo con el mismo nombre del usuario y este pasa a pertenecer a dicho grupo, también se crea el home /home/nombre-usuario y el Shell se asigna /bin/bash por defecto.

Desarrolle los siguientes ejercicios:

1. Cree un grupo denominado **profesores** y otro denominado **estudiantes**
2. Cree los usuarios **profe1**, **profe2** y asígneles al grupo **profesores**
3. Cree los usuarios **user1**, **user2** y asígneles al grupo **estudiantes**
4. Cree el usuario **especial** y asígnelo al grupo **profesores** y **estudiantes**.

- c) Comandos para el manejo de permisos de archivos

A continuación se presentan algunos comandos básicos:

- chmod:** cambia los permisos de acceso de los archivos.
- chown:** cambia el usuario y grupo propietarios de archivos
- chgrp:** cambia el grupo al que pertenecen los archivos

Los permisos se definen para el propietario del archivo (**user**), los miembros del grupo a que pertenecen los usuarios (**group**) y el resto de usuarios (**others**). Los permisos pueden ser de lectura (**read**), de escritura (**write**) o de ejecución (**execute**). Existen dos maneras de representar los permisos: con caracteres alfanuméricos o con números octales.

Las letras **u**, **g** y **o** representan a user, group y others, el signo **=** significa fijar los permisos, y las letras **r**, **w** y **x** representan los permisos de read, write y execute respectivamente.

Ejemplo:

chmod u=rwx,g=rx,o=r nombre-archivo

En números octales el mismo ejemplo quedaría de la siguiente forma:

chmod 754 nombre-archivo

En este caso cada número representa los permisos para el usuario, grupo y otros en ese orden. La representación en binario de 3 bits de cada uno de ellos indica los permisos de lectura, escritura y ejecución (1 si el permiso existe y 0 si no existe) en ese orden. Por ejemplo, 5 en binario sería 101 lo que significa que el archivo tendrá permisos de lectura y ejecución, pero no de escritura.

Desarrolle los siguientes ejercicios:

1. En un directorio vacío (nuevo), crear 9 archivos (archiv1,archiv2,etc.) utilizando el comando **touch**.

2. Modificar los permisos usando el comando **chmod**, para que queden de la siguiente manera:

```
archiv1 -rwx-----  
archiv2 -rw-----  
archiv3 -rwxrwxrwx  
archiv4 -rwxrw-r--  
archiv5 -rwxr-----  
archiv6 -r-xrw-r--  
archiv7 -r-----x  
archiv8 -rw-r--r--  
archiv9 -rw-rw-r--
```

3. Modificar el dueño y grupo de los archivos para que quede de la siguiente manera:

```
archiv1 profe1 profesores  
archiv2 profe2 profesores  
archiv3 user1 estudiantes  
archiv4 user2 estudiantes  
archiv5 especial profesores  
archiv6 especial estudiantes
```