

Laboratorio de Sistemas Operativos



Práctica 4: Administración

Objetivo

Las prácticas anteriores se han enfocado exclusivamente en el manejo básico de comandos de consola. En cuanto a la administración de un sistema UNIX vimos lo de los permisos de UGO (User-Group-Others) y sus modificaciones. Esta práctica se centra en:

- El acceso y administración de usuarios y de recursos locales y remotos
- Montaje de dispositivos
- Configuración de interfaces y otras tareas administrativas.

Índice

[Comandos](#)

[Administración de usuarios](#)

[Root](#)

[Crear usuarios y grupos.](#)

[Actividad 1:](#)

[Cambiar la contraseña](#)

[Eliminar usuarios](#)

[Cambiar permisos](#)

[Instalar aplicacion con manejador de paquetes.](#)

[Administración de aplicaciones en ejecución](#)

[Interfaces de red](#)

[Alámbricas \(Wired\)](#)

[Inalámbricas](#)

[Montar dispositivos de almacenamiento](#)

[Memorias Flash USB.](#)

[Archivos ISO](#)

[Discos duros](#)

[Ver usuarios conectados](#)

[Enviando mensajes a usuarios.](#)

[Programando la ejecución de tareas](#)
[RespalDOS.](#)
[Conexiones remotas.](#)
[Información adicional](#)
[Comandos de administración y más allá...](#)

[Formulario laboratorio](#)

Comandos

chmod	Cambia los permisos de los archivos.
su	Permite darse de alta con otra cuenta de usuario.
groupadd	Añade un nuevo grupo al sistema
useradd	Añade un nuevo usuario al sistema
who	Despliega listado de usuarios conectados
whoami	Despliega el usuario actual
whos	Despliega lista de usuarios más detallada (origen y actividad)
chown	Permite cambiar a los dueños de los archivos

Administración de usuarios

El manejo de usuarios y grupos en sistemas GNU/Linux ha ido cambiando en las últimas dos décadas. Se distinguen dos tipos de usuarios - los **usuarios comunes** y los **administrativos**.

Usuario administrativo: es el que tiene la capacidad de alterar los parámetros del sistema (*settings*) y administrar a otros usuarios con operaciones tales como darlos de alta o de baja, o bien cambiar los privilegios de algún usuario. El usuario administrativo con mayores privilegios tiene la cuenta llamada **Root**, y el password de esta cuenta se define una sola vez al instalar Unix. Root puede dar suficientes privilegios a otros usuarios comunes para convertirlos en usuarios administrativos que puedan tener algunos de los privilegios del mismo Root.

Un **usuario** puede pertenecer a un grupo de usuarios, y opcionalmente a varios más. Los sistemas GNU/Linux por defecto añaden a los usuarios comunes a varios grupos para permitirles cierto control sobre el equipo. Como ejemplo tenemos el siguiente usuario:

```
User1@Muud-Linux:~$ groups User1
User1 : User1 adm cdrom sudo dip plugdev lpadmin sambashare wireshark
```

Se puede observar que el usuario **User1** forma parte de ciertos grupos por defecto (“adm”, “cdrom” etc); de hecho sin ellos no podría montar y desmontar unidades de Cd-ROM o memorias USB sin utilizar los privilegios del administrador. Está también en el grupo “**adm**”, porque se trata de un usuario administrativo que puede utilizar su propia contraseña en vez de la de Root; es decir, es

un usuario al que **Root** ha otorgado privilegios administrativos

Además de los usuarios humanos (comunes y administrativos), ciertas aplicaciones requieren tener permisos para acceder diversos archivos diferentes a los archivos propios de cualquier usuario común - por ejemplo, un manejador de bases de datos necesita tener sus propios archivos para sus tablas, que son accesadas por muchos usuarios comunes. Entonces al instalar ese dbms, Linux crea usuarios-aplicaciones que no son otra cosa que un usuario digamos “artificial” por aplicación y les otorga los permisos requeridos. Se requieren este tipo de usuarios-aplicación también para los “daemons” o programas que corren siempre en background y que igualmente pueden dar servicio a múltiples usuarios comunes.

Root

Root es el super-usuario administrativo que tiene control total del sistema. Root puede dar y quitar privilegios y por lo mismo el acceso a esta cuenta debe estar fuertemente protegido en el sistema. En los sistemas GNU/Linux modernos hay dos variantes para acceder a los privilegios de Root:

1. Directamente entrando a la cuenta de Root por medio de login y password de root
2. Mediante el comando SUDO.

Para acceder a una cuenta Root se requiere la contraseña de dicha cuenta (usualmente configurada al inicio de la instalación del sistema) y se puede hacer mediante el comando **su**

```
user1@Localhost:~$ su [root]
password: <Password de Root>
root@Localhost:/home/user1#
```

Un usuario común puede hacer uso del comando SUDO para acceder a root como se muestra a continuación:

```
user1@Localhost:~$ sudo su [root]
[sudo] password for user1: <password>
root@Localhost:/home/user1#
```

Curiosidad: Los sistemas operativos Android están basados en Linux y la cuenta de Root queda sin un password así que por default un usuario común no puede tener control total del teléfono.

Un sistema GNU/Linux que hace uso de SUDO tiene por defecto a los usuarios configurados como “administrativos” con capacidad de utilizar SUDO como si fuesen root (En la siguiente práctica hablaremos más a fondo de esta herramienta).

Crear usuarios y grupos.

Una de las actividades más importantes en la administración de un sistema Unix es la creación de nuevos usuarios y de grupos de trabajo. Como mencionamos anteriormente, en los sistemas Unix existen dos tipos de usuarios que se diferencian por los permisos que tienen para ejecutar ciertos programas o acceder a ciertos archivos. El usuario que tiene acceso a todos los archivos y es capaz de ejecutar cualquier acción o programa se llama **super usuario (root)**; los demás usuarios tienen restricciones fijadas por root.

Linux guarda la información de cada usuario en el archivo **/etc/passwd** y la información de grupos en el archivo **/etc/group** en el formato: **group_name:passwd:GroupID**.

Revisa el contenido de estos archivos en tu sistema, y anota los datos de tu grupo (puedes identificar a qué grupos perteneces con el comando **groups**):

Group_name: _____
Passwd: _____
GroupID (GID): _____

Ahora corteja esa información con **/etc/passwd** el cual tiene el formato:

username:password:User ID:Group ID:User ID Info:Home:Shell ,

Deberá de aparecer el mismo GID, de hecho pueden utilizar el siguiente comando para hallar su registro en **/etc/passwd** con más facilidad: **"grep <GID> /etc/passwd"**.

Notarán que existen IDs para usuarios e IDs para grupos. Esto se debe a que los nombres nos facilitan la vida a los humanos pero para las máquinas un número único para cada uno les es mejor para la administración.

Password: El password de todos los usuarios por razones de seguridad no se halla en este archivo si no que se halla en **/etc/shadow** encriptado y protegido de mirones (es un archivo que solo Root puede leer).

Actividad 1:

Se desean crear los siguientes grupos con sus respectivos usuarios:

Grupo:	Mexico	EU	Britain
Usuarios:	PeñaNieto Fox	Trump Melania Ivanka	QueenElizabeth TheresaMay

Recuerda que para poder hacer cambios en el sistema debes autenticarte como **super usuario**, para ello:

Fedora: teclea el comando **su** y el password cuando se te pida.

Ubuntu: por el diseño que se le dió, la instrucción **SUDO** sustituye a **su**. Es decir, para pasar a la cuenta root en Ubuntu es **sudo su**.

Para **crear un grupo** utiliza el comando **groupadd <groupName>**. Crea cada uno de los grupos mostrados en la tabla anterior y reporta el **GroupID** que se asignó a cada uno.

Parents: _____
Children: _____
Soho: _____

Si el sistema te muestra el **error**: "bash: groupadd: command not found", quiere decir que el comando **groupadd** no se encuentra en la ruta de directorios que utiliza el sistema para localizar comandos; por lo que deberás ejecutar el siguiente comando:

export PATH=\$PATH:/usr/sbin

para agregar a la ruta del sistema el directorio donde se encuentra el groupadd y otros comandos.

Después procede a **crear los usuarios** y asignar cada uno a los grupos que corresponda. Para ello usarás el comando:

useradd -m -g <groupName> <userName>.

Si no especificas la opción “-g <groupName>” se crea automáticamente un nuevo grupo con el mismo nombre del usuario. La opción '-m' le indica al comando que cree el directorio del usuario en /home. Ejecuta el comando: **ls -l /home** y observa las columnas con los nombres de usuario y grupo.

***TIP:** el directorio /home/<user> puede ser opcional, porque no necesariamente el usuario que accede al sistema necesita almacenar archivos. En particular, un administrador puede no requerir almacenar sus propios archivos.*

Cambiar la contraseña

Hay que crear una contraseña para cada usuario. Aunque, cuando inicias sesión como un **super usuario** puedes iniciar sesión en la cuenta de cualquier usuario sin la necesidad de conocer su contraseña. Para que lo compruebes, puedes intentar iniciar sesión como si fueras algún usuario en particular con el comando: **su <userName>**, y cerrar esa sesión con el comando **exit**.

Para crear la contraseña de cada usuario debes, además de ser **super usuario**, teclear el comando: **passwd <userName>**.

También, el propio usuario puede cambiar su contraseña al teclear el comando **passwd**. Fija como password para cada cuenta el nombre de usuario invertido, por ejemplo el password de jane es “enaj”.

***TIP:** Si se crean las cuentas de usuario con la sintaxis del ejemplo no harán uso de BASH sino de SH, por lo tanto cuando entran a esas cuentas seguramente solo les desplegará un caracter (\$ ó # dependiendo del tipo de usuario). Pueden utilizar lo siguiente para entrar con BASH: **su -s /bin/bash <Usuario>** .*

Actividad 2:

Eliminar usuarios

Para **eliminar usuarios y sus archivos** en el directorio /home se utiliza el comando: **userdel -r <userName>**.

Elimina todos los usuarios que creaste anteriormente. Para la **eliminación de grupos** el comando es : **groupdel <Group>**

Actividad 3:

Cambiar permisos

Ya en la práctica dos manejamos el control de permisos de un archivo (UGO) mediante chmod pero siempre fue con archivos de un mismo usuario. En ocasiones nos toparemos con que es necesario hacerle modificaciones al archivo o bien cambiar el dueño del mismo; una razón típica

es cuando estamos utilizando la cuenta Root pero deseamos crear archivos en nuestro propio /home. Por ejemplo, sigue los siguientes pasos:

1. Siendo **super usuario** (para verificarlo utiliza el comando **whoami**) crea un archivo de texto con el siguiente comando: **echo "hola" > prueba**
2. Sal de la cuenta de **super usuario** con el comando **exit**, verifícalo con el comando **whoami**.
3. Trata de concatenar la palabra "mundo" al contenido del archivo **prueba** usando el comando **echo**. Escribe cómo lo hiciste.
4. Notarás que aparece el error: **"Permission denied"**.
5. Observa el **username** y **grupo** del propietario de dicho archivo con el comando: **ls -l prueba**.
6. Para corregir este error inicia de nuevo sesión como **super usuario** y modifica el propietario de dicho archivo, con el comando:
chown <username>:<groupname> <filename>
usando los valores correctos (los de tu cuenta).
7. Sal de la cuenta **super usuario**.
8. De nuevo trata de concatenar la palabra "mundo" al contenido del archivo **prueba**, y reporta una captura de pantalla de la ejecución de los comandos: **"cat prueba"** y **"ls -l prueba"**.

Instalar una aplicación con el manejador de paquetes.

Un manejador de paquetes es una aplicación que sirve para automatizar el proceso de **actualización, configuración, instalación y eliminación** de software en sistemas Linux. Algunos de los más conocidos son:

- **yum** de la distribución Fedora
- **apt-get** (APT) de Debian.

Puedes usar esta herramienta para instalar casi cualquier programa disponible en código abierto. A continuación instalarás la aplicación **htop** (Figura 1) utilizando el manejador de paquetes; **htop** es un visor de procesos interactivo para Linux, similar al Task Manager de Windows pero que se ejecuta en terminal.

Imagen 1 - Htop

CPU[|||||] 2.0%

Mem[|||||] 13/123MB

Swap[] 0/109MB

Tasks: 16 total, 1 running

Load average: 0.37 0.12 0.04

Uptime: 00:00:50

PID	USER	PRI	NI	UIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
3692	per	15	0	2424	1204	980	R	2.0	1.0	0:00.24	htop
1	root	16	0	2952	1852	532	S	0.0	1.5	0:00.77	/sbin/init
2236	root	20	-1	2316	728	472	S	0.0	0.6	0:01.06	/sbin/udevd --daemon
3224	dhcpc	18	-2	2412	552	244	S	0.0	0.4	0:00.00	dhclient3 -c IF_ME
3488	root	18	0	1692	516	448	S	0.0	0.4	0:00.00	/sbin/getty 38400
3491	root	18	0	1696	520	448	S	0.0	0.4	0:00.01	/sbin/getty 38400
3497	root	18	0	1696	516	448	S	0.0	0.4	0:00.00	/sbin/getty 38400
3500	root	18	0	1692	516	448	S	0.0	0.4	0:00.00	/sbin/getty 38400
3501	root	16	0	2772	1196	936	S	0.0	0.9	0:00.04	/bin/login --
3504	root	18	0	1696	516	448	S	0.0	0.4	0:00.00	/sbin/getty 38400
3539	syslog	15	0	1916	704	564	S	0.0	0.6	0:00.12	/sbin/syslogd -u s
3561	root	18	0	1840	536	444	S	0.0	0.4	0:00.79	/bin/dd bs 1 if /p
3563	klog	18	0	2472	1376	408	S	0.0	1.1	0:00.37	/sbin/klogd -P /va
3590	daemon	25	0	1960	428	308	S	0.0	0.3	0:00.00	/usr/sbin/atd
3604	root	18	0	2336	792	632	S	0.0	0.6	0:00.00	/usr/sbin/cron
3645	per	15	0	5524	2924	1428	S	0.0	2.3	0:00.45	-bash

Ubuntu:

Abre una terminal y ejecuta **sudo apt-get install htop** . O si lo prefieres consulta **man apt** para ver las opciones de instalación. **sudo** ejecuta esa línea como usuario root.

Fedora:

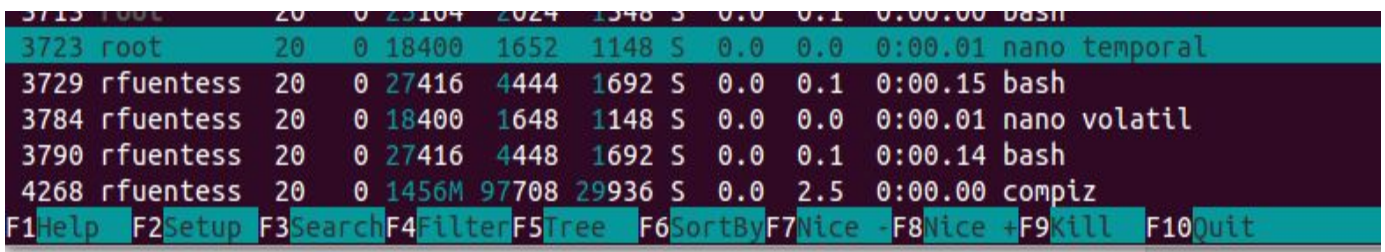
Abre una terminal y ejecuta el comando **su (usr@domain#su)** para pasar a ser usuario root, a continuación utiliza el comando **yum install htop** . O si lo prefieres consulta **man yum** para ver las opciones.

Administración de aplicaciones en ejecución

Un usuario es dueño de las aplicaciones que él está ejecutando y por lo tanto las puede controlar, pero no así las aplicaciones de otros usuarios. **Sólo root tiene control total sobre todas las aplicaciones.**

Para un ejemplo de control de aplicaciones en ejecución sigue los siguientes pasos:

1. Abre una segunda terminal y cámbiate a Superusuario. Ejecuta **nano temporal**. **nano** es un editor de textos
2. Abre una tercera terminal y utilizando tu cuenta de usuario, ejecuta **nano volatil**
3. En la terminal original, estando como usuario estándar ejecuta el programa **"htop"**
4. Con la terminal mostrando la pantalla de htop, presione F3 para buscar NANO
5. Selecciona con las flechas el programa NANO de tu cuenta de usuario
6. Presiona F9 (Kill). Se abrirá un menú en la parte izquierda. Selecciona la opción que viene por defecto (SIGTERM) y vuelve a presionar ENTER.
7. Ahora revisa la terminal donde tenías **"nano volatil"** en ejecución.
8. Intenta lo mismo: buscar nano y matarlo
9. Si el programa sigue apareciendo anota el numero que esta en la primera columna (Que es el Process ID).



3713	root	20	0	23104	2024	1348	S	0.0	0.1	0:00.00	bash
3723	root	20	0	18400	1652	1148	S	0.0	0.0	0:00.01	nano temporal
3729	rfuentess	20	0	27416	4444	1692	S	0.0	0.1	0:00.15	bash
3784	rfuentess	20	0	18400	1648	1148	S	0.0	0.0	0:00.01	nano volatil
3790	rfuentess	20	0	27416	4448	1692	S	0.0	0.1	0:00.14	bash
4268	rfuentess	20	0	1456M	97708	29936	S	0.0	2.5	0:00.00	compiz

F1 Help F2 Setup F3 Search F4 Filter F5 Tree F6 SortBy F7 Nice - F8 Nice + F9 Kill F10 Quit

Abre una nueva terminal y entra a Root o bien utiliza sudo con el comando **"kill -9 <process id>"**. Por ejemplo, en la figura anterior el proceso de Root que queremos matar es el 3723, el -9 es la señal "SIGTERM" que seleccionamos en htop.

Busca ahora si existe todavía dicho proceso o si este ha sido eliminado. Como verás, es un modo sencillo de matar aplicaciones; existe otro comando **"ps -A e"** que nos imprimirá el listado de procesos en ese instante (a diferencia de htop que es en tiempo real). Si utilizamos grep podremos lograr algo similar.

Actividades 4-7:

Interfaces de red

Algunas aplicaciones realizan el trabajo de conectar una computadora con Linux a una conexión de red que se encuentre disponible; sin embargo algunas ocasiones es necesario hacerlo de forma manual. A continuación aprenderás a configurar interfaces ethernet wired y wireless.

Las configuraciones manuales de las interfaces involucran entender cómo funciona el protocolo de red (IP) y el protocolo de la capa de datos (Ethernet/802.3 o los de 802.11x), por lo tanto en ocasiones puede resultar un poco complejo cuando no se tienen estos conocimientos.

Alámbricas (Wired)

Para configurar **interfaces alámbricas** se requiere primero levantar la interfaz y después solicitar una dirección de red. Los pasos son los siguientes:

1.-Ver cuáles interfaces tiene instaladas el sistema, usando el comando: **ifconfig -a**

Aquí buscamos el nombre de la interface **ethernet**, que generalmente se llama **eth0** (si tienes muchas interfaces es posible que cambie la numeración). Si tienes duda acerca del uso del comando **ifconfig** puedes ver una ayuda rápida del mismo con la bandera **--help**, o la ayuda completa con el comando **man ifconfig**

2.- Encender la interface con el comando: **ifconfig <Interfaz> up**

Podemos ver que está encendida si buscamos la palabra **UP** en la salida que nos entrega el comando: **ifconfig <Interfaz>**

3.-Después pedimos una dirección ip al servidor DHCP con el comando: **dhclient <Interfaz>**

Algunas veces, esta aplicación no está instalada y en su lugar usamos: **dhcpcd <Interfaz>**

En caso de que encuentres un error que te indique que no se puede renovar la dirección IP, deberás ejecutar cualquiera de los comandos anteriores con la bandera **“-r”**, por ejemplo: **dhclient -r**

4.- Verificamos que se nos haya asignado una dirección IP con el comando **ifconfig** en el campo **inet addr**:

ifconfig <Interfaz>

Inalámbricas

Para configurar **interfaces wireless**, se requiere levantar la interfaz, conectarnos a un punto de acceso en particular y luego solicitar dirección de red. Los pasos son los siguientes:

- Ver cuáles interfaces con capacidades wireless tiene instaladas el sistema: **iwconfig**

Las interfaces que muestran un valor diferente a **“no wireless extension”** son las interfaces que son de tipo wireless. Algunos nombres comunes son: **“at0, eth1, wlan0, ...”**.

- Identificamos el nombre de la interface y apagamos la interface para configurarla, ya que algunos drivers no soportan el cambio de la configuración mientras la interface está

encendida: **ifconfig <Interfaz> down**

- Procedemos a buscar los Access Points (antenas) que tenemos al alcance y tomamos los datos del que nos interesa: **iwlist <Interfaz> scan**

Estos datos son: **essid** y **channel**.

- Configuramos la interface wireless: **iwconfig <Interfaz> essid <essidDelAp> channel <canalDelAP>**

Si el AP tiene seguridad WEP o WAP se especifica la llave que queremos usar con el comando:
iwconfig <Interfaz> key s:<claveWEP>

Verificamos que todo haya quedado bien configurado con el comando: **iwconfig <Interfaz>**

- Encendemos la interface y solicitamos una dirección IP, utilizando dhclient o dhcpcd:

ifconfig <Interfaz> up

***TIP:** Como podrás ver esto de repente no es tan sencillo , pero desviando la salida del escaneo de puntos de acceso a un archivo de entrada a awk para filtrar resultados y obtener solo access points válidos, se puede automatizar el proceso de conexión.*

Muy probablemente encuentres direcciones que empiecen con FE80:--- y/o 2000:--- éstas son direcciones del nuevo protocolo de red IPv6 y si la interfaz lo soporta, mínimo activará una que inicie con FE80 y si la red ya tiene implementado dicho protocolo, seguramente se auto-asignará una que inicie con 2000.

Actividades 8,9:

Montar dispositivos de almacenamiento

En la actualidad los sistemas Linux montan de forma automática las unidades de memoria Flash USB o los discos duros externos; sin embargo es bueno que sepas hacerlo de forma manual. Un ejemplo de la utilidad de esto es el poder montar un archivo ISO (imagen bit por bit de un CD) en una carpeta, ahorrándote la necesidad de quemar un CDROM y aumentando la velocidad de acceso a los archivos dentro de la imagen ISO.

Memorias Flash USB.

- Primero, se conecta la memoria a la computadora. Luego se debe revisar el nombre del dispositivo que esta tomó, con el comando: **dmesg | tail**

La Imagen 2 muestra un ejemplo de la salida que deberías obtener de este comando. Como puedes ver en la última línea, la escritura en la memoria Flash se hará a través del dispositivo: /dev/sdb1

```

usb 1-1: new high speed USB device using ehci_hcd and address 3
usb 1-1: configuration #1 chosen from 1 choice
scsi4 : SCSI emulation for USB Mass Storage devices
usb 1-1: New USB device found, idVendor=1516, idProduct=8628
usb 1-1: New USB device strings: Mfr=1, Product=2, SerialNumber=3
usb 1-1: Product: Drive SK_USB20
usb 1-1: Manufacturer: Flash
usb 1-1: SerialNumber: 8990000000000000356F503D
usb-storage: device found at 3
usb-storage: waiting for device to settle before scanning
usb-storage: device scan complete
scsi 4:0:0:0: Direct-Access    Flash      Drive SK_USB20   1.00 PQ: 0 ANSI: 2
sd 4:0:0:0: [sdb] 1994752 512-byte hardware sectors (1021 MB)
sd 4:0:0:0: [sdb] Write Protect is off
sd 4:0:0:0: [sdb] Mode Sense: 23 00 00 00
sd 4:0:0:0: [sdb] Assuming drive cache: write through
sd 4:0:0:0: [sdb] 1994752 512-byte hardware sectors (1021 MB)
sd 4:0:0:0: [sdb] Write Protect is off
sd 4:0:0:0: [sdb] Mode Sense: 23 00 00 00
sd 4:0:0:0: [sdb] Assuming drive cache: write through
sdb: sdb1
sd 4:0:0:0: [sdb] Attached SCSI removable disk
sd 4:0:0:0: Attached scsi generic sg2 type 0
SELinux: initialized (dev sdb1, type vfat), uses genfs_contexts
[robertoaceves@localhost ~]$

```

Imagen 2

Cada dispositivo de entrada o salida en Linux se considera como un archivo o sistema de archivos, incluyendo dispositivos como el teclado y la pantalla y los sockets (canales de transmisión de datos entre procesos), además de los CDROMS o discos duros o lo que pueda venir en el futuro. El directorio **/mnt** se utiliza para montar sistemas de archivos temporalmente. **/media** se utiliza para medios que pueden conectarse o desconectarse mientras la computadora está preñida, ej: CDROM, memorias flash.

Generalmente el directorio **/media** tiene **/media/cd**, **/media/dvd** y **/media/fl** que pueden ser utilizados para “montar” dispositivos. **“Montar” la unidad significa poder ver su contenido como si fuera un directorio más en el sistema de archivos de Linux.** En el caso en que no se tengan dichos directorios se pueden crear a conveniencia. Este es el camino que se utilizará en la práctica:

- Montarás tu memoria flash en el directorio llamado **/media/memoriaFlash** que vas a crear enseguida:

```

su
cd /media
mkdir memoriaFlash
ls

```

- Ahora se procede a montar la unidad:
mount /dev/<nombreUnidadFlash> /media/memoriaFlash

Generalmente, como se vio en la Imagen 2, el nombre de la unidad Flash tiene el formato: **sda#**, **sdb#**.

- Si todo salió bien, ahora puedes ver el contenido de la memoria Flash:

ls /media/memoriaFlash

- Para desmontar la memoria Flash y evitar cualquier daño a los datos escritos, es necesario desmontar correctamente la memoria Flash con el comando:

umount /media/memoriaFlash

No es necesario borrar el directorio *memoriaFlash*, puede ser re-usado.

Archivos ISO

- Se crea un subdirectorio, siendo **super usuario**, dentro del directorio **/mnt**, donde se montará esta unidad:

```
su
cd /mnt
mkdir archivoisomontado
ls
```

- Puedes utilizar cualquier imagen de disco, por ejemplo la del disco de instalación que usaste para instalar linux en tu computadora. En esta práctica le llamaremos *imagendeisoprueba.iso*.
- A continuación, estando en el directorio donde se encuentra el archivo **iso**, se procede a montar el archivo **iso** con el siguiente comando:

```
mount -o loop imagendeisoprueba.iso /mnt/archivoisomontado
ls /mnt/archivoisomontado/
```

Trata de ejecutar el archivo: */mnt/archivoisomontado/imageniso/pinball*

- Para desmontar el archivo **iso** ejecuta el comando:
- umount /mnt/archivoisomontado**

Discos duros

Para montar particiones o discos duros de un sistema Linux, algunas veces es necesario conocer el tipo de **filesystem** que tiene para pasárselo como argumento a mount. Como ejemplo se muestra la siguiente sintaxis:

```
su
cd /mnt
mkdir discoDuro1
ls
mount -t <filesystem> <device> <dir>
umount <dir>
```

Donde:

- **filesystem**, puede ser por ejemplo: *vfat*, *ext3*, *ntfs*, *hfs*;
- **device**, puede ser alguno de tipo: */dev/sdb1*, */dev/hda2*, */dev/hdb3*;
- **dir**, es el directorio donde se quiere montar.

Ver usuarios conectados

Los sistemas GNU/Linux **son multi-usuarios en tiempo real**; esto significa que pueden por defecto soportar múltiples usuarios (humanos) accediendo de forma simultánea a los recursos del sistema desde el mismo equipo o de forma remota. Por lo mismo existen ciertas herramientas para facilitar este ambiente.

En la práctica sobre shell programming tuviste contacto con el comando **who**, que permite ver qué usuarios están conectados a una computadora. Este comando arroja información sobre el nombre de usuario, la terminal a la que está conectado el usuario, la última fecha de acceso, y la dirección desde donde se realizó la conexión. Por ejemplo:

```
[user@localhost home]$ who
operator    tty0        Jul 15 17:44
al702782    pts/1       Aug 05 19:26      (a estas maquinas)
al194834    pts/2       Aug 05 19:29      (131.178.74.107)
al159884    pts/4       Aug 05 19:13      (a estas maquinas)
al333793    pts/7       Aug 05 19:19      (131.178.74.149)
al172871    pts/10      Aug 05 18:35      (131.178.82.22)
amartine    pts/12      Aug 05 19:45      (tserv.mty.itesm.)
jnolazco    pts/13      Aug 05 19:55      (ciencias.mty.ite)
```

Existe otro, el comando **w**, que muestra información más detallada sobre los usuarios conectados a la computadora, puedes ver que la descripción de los campos con el comando `man w`.

```
[user@localhost home]$ w
11:22:57 up 11 days, 19:17, 3 users, load average: 0.00, 0.01, 0.00
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU   WHAT
robertoa  pts/0    10-11-12-13 11:09    0.00s  0.30s  0.00s  w
sgeadmin  tty8     :0              Mon16    23:45  48.21s 0.08s  x-session-manager
```

TIP: El usuario *Root* puede terminar lo que estos usuarios están haciendo mediante *kill* (aunque primero deberá obtener el *process-ID* del proceso del usuario).

Enviando mensajes a usuarios.

En algunas ocasiones es necesario mandar mensajes a la terminal de otros usuarios conectados a la computadora, por ejemplo para avisarles que se reiniciará el sistema. El comando que permite esta tarea es **write**, y su sintaxis es la siguiente:

```
write <nombre-de-usuario> <ENTER>
cualquier mensaje ...
<CTRL+D>
```

El comando **write** permite la comunicación en un solo sentido, por lo que si un usuario desea responder los mensajes que otro le envía debe también responder mediante el comando **write**. Es una especie de chat. Para terminar la comunicación se deben presionar las teclas `<CTRL+D>`

A continuación se muestra un ejemplo de una conversación entre dos usuarios, puedes seguir la secuencia mediante los números entre paréntesis:

```
(01) [robertoaceves@sge-handler32 ~]$ write paola
(02) hola
(05) como ça va?

(09) Message from paola@sge-handler32 on pts/1 at 11:51 ...
(10) ça va bien
(11) ok
(13) au revoir
```

```

(15) bye
(17) <CTRL+D> EOF

[robertoaceves@sge-handler32 ~]$

paola@sge-handler32:~$
(03) Message from robertoaceves@sge-handler32 on pts/0 at 11:50 ...
(04) hola
(06) como ?ça va??

(07) paola@sge-handler32:~$ write robertoaceves
(08) ça va bien
(12) ok
(14) au revoir
(16) bye
(18) EOF
paola@sge-handler32:~$

```

Existe, también un comando que permite mandar mensajes a las terminales de todos los usuarios: **wall**. Su sintaxis es la siguiente:

```

[user@localhost home]$ wall
mensaje
<CTRL+D>

```

Este comando manda un mensaje en **broadcast** a todos los usuarios, y al igual que el comando anterior, permite la comunicación en solo un sentido.

TIP: Los usuarios que no tienen abierta alguna terminal no recibirán un aviso, lo cual les puede *ser contraproducente si se ejecuta el comando para avisar que se va a apagar el servidor y estos no pudieron guardar sus trabajos*.

Programando la ejecución de tareas

Como administrador, o como usuario, alguna vez tendrás la necesidad de programar la ejecución de algún comando a determinada hora y día. Para ello existen algunos comandos en los sistemas Unix-like, como: at, cron, nice.

El comando **at** permite ejecutar un comando a un tiempo específico. Puedes consultar la lista completa de opciones con el comando **man at**. La sintaxis del comando es la siguiente:

```

[user@localhost home]$ at <hora> <fecha>
at> comando-a-ejecutar
at> uno-por-línea
at> <CTRL+D>

```

Donde las opciones son:

- **<hora>** : Hora a la que se ejecutará el comando, con formato HH:MM
- **<fecha>** : Día que se ejecutará el comando, con formato DD.MM.YY , si se omite se toma como referencia el día de hoy.

“**at>**” simboliza el prompt del comando at, aquí es donde se teclean los comandos, uno por línea, que se desean ejecutar. Para terminar de capturar comandos hay que presionar las teclas <CTRL+D> .

Este comando no produce salida, cualquier mensaje que se imprima a la salida estándar o

cualquier error se registrará en el archivo: /var/mail/nombre-de-usuario

Se muestra un ejemplo del uso del comando at a continuación:

```
[user@localhost home]$ ls
[user@localhost home]$
[user@localhost home]$ date
Mon Sep  7 18:20:44 CDT 2009

[user@localhost home]$ at 18:29 07.09.09
warning: commands will be executed using /bin/sh
at> echo "Esta es una prueba del comando at, que ejecuta un comando en el tiempo programado"
at> mkdir cosas
at> cd cosas
at> echo "1 2 3 4 5" > numeros.txt
at> <EOT>
job 5 at Mon Sep  7 18:29:00 2009

[user@localhost home]$ date
Mon Sep  7 18:25:31 CDT 2009
[user@localhost home]$ date
Mon Sep  7 18:27:12 CDT 2009
[user@localhost home]$ You have mail in /var/mail/nolazco
[user@localhost home]$ date
Mon Sep  7 18:29:32 CDT 2009

[user@localhost home]$ tail /var/mail/nolazco
      id 1M1A7Q-0000G3-6d
      for nolazco@sge-handler32.mty.itesm.mx; Mon, 07 Sep 2009 18:29:00 -0500
Subject: Output from your job          5
To: nolazco@sge-handler32.mty.itesm.mx
Message-Id: <E1M1A7Q-0000G3-6d@sge-handler32.mty.itesm.mx>
From: nolazco@sge-handler32.mty.itesm.mx
Date: Mon, 07 Sep 2009 18:29:00 -0500

Esta es una prueba del comando at, que ejecuta un comando en el tiempo programado

[user@localhost home]$ ls
cosas
[user@localhost home]$ ls cosas
numeros.txt
[user@localhost home]$ cat cosas/numeros.txt
1 2 3 4 5
[user@localhost home]$
```

Actividad 10:

RespalDOS.

Dado que lo más importante que existe en una computadora o servidor es la información que contiene, es vital hacer respaldos de modo periódico. Para una computadora personal, es recomendable tener al menos un respaldo para prevenir pérdida de datos en caso de una falla del disco duro. Una herramienta que facilita esta labor (sobre todo si se utiliza junto con **cron**) es **rsync**, ya que es una forma de respaldo rápida y versátil de copiar archivos ya sea de modo local o de modo remoto.

Una ventaja que tiene **rsync** sobre otras herramientas (**scp** o **ftp**) es su capacidad de manejar

información de modo diferencial; esto significa que puede encontrar qué necesita ser respaldado o sincronizado haciendo una búsqueda de los archivos que han cambiado desde la última sincronización, y solo transfiere la diferencia.

Su uso más común está definido del siguiente modo: `rsync -avz /src/foo/ /dest/foo`

De este modo se sincroniza `/dest/foo` con los cambios hechos en `/src/foo`. Esto se puede hacer a través de la red incluyendo la máquina en la dirección de origen o destino e.g:

```
rsync -avz /src/foo/ maquina-remota:/dest/foo
```

```
mkdir orig dest
cd orig
wget http://launchpad.net/exaile/0.3.0/0.3.0.1/+download/exaile-0.3.0.1.tar.gz
tar xvfz exaile-0.3.0.1.tar.gz
rsync -avz ./ ../dest
echo "prueba" >> exaile-0.3.0.1/README
rsync -avz ./ ../dest
```

Nota como solo se transfiere el archivo README ya que es lo único que se ha modificado.

Actividad 11,12:

Conexiones remotas.

El modo en el que comúnmente se manejan los servidores es teniendo una computadora corriendo servicios sin periféricos conectados a ella, esto es, sin teclado o monitor. La administración de estos servicios se hace de modo remoto gracias a herramientas como **ssh**.

ssh es un programa que permite hacer conexiones de ingreso remotas a los equipos que estén corriendo dicho servicio, de un modo seguro. Una vez conectado a dicha máquina y autenticado como usuario, uno es capaz de ejecutar comandos como si estuviera físicamente junto a la computadora. La sintaxis de este comando es: `ssh usuario@maquina-remota`

Donde **usuario** es tu nombre de usuario, **máquina-remota** es una dirección ip o un nombre de dominio. Por ejemplo, para conectarse a la máquina *cluster.itesm.mx* con el usuario *pedro*:

```
ssh pedro@cluster.itesm.mx
```

Si el servidor tiene instalado Xorg (servidor del sistema de ventanas X) y el ancho de banda de la red es suficiente, uno puede, a través de **ssh**, ejecutar programas con interfaz gráfica en la computadora remota y verla desplegada en la sesión de X local. Esto se logra con la opción “-XYg” de **ssh**.

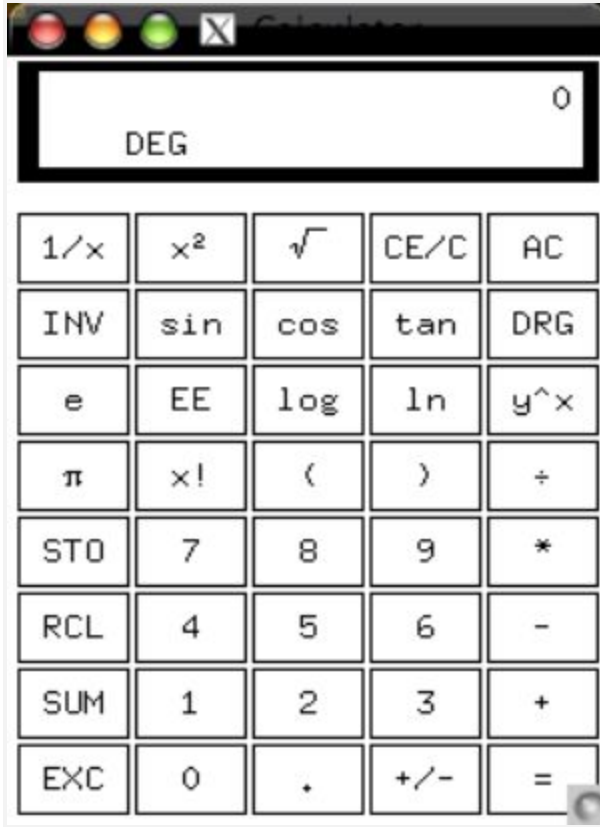
```
ssh -XYg usuario@maquina-remota
```

Por ejemplo, si deseas ejecutar la calculadora (xcalc) de la máquina *cluster.itesm.mx* y tu usuario es *pedro* lo harías de la siguiente manera:

```
[user@localhost home]$ ssh -XYg pedro@cluster.itesm.mx
```



```
[pedro@cluster.itesm.mx home]$ xcalc &
```



scp es una herramienta que permite transferir archivos de un modo seguro a una o desde una computadora remota utilizando el protocolo de **ssh**. Este es uno de los métodos más comunes para subir archivos individuales a los servidores. La sintaxis de este comando es:

```
scp archivo-origen usuario@maquina-remota:/ruta/al/destino
```

Información adicional

En este [blog](#) se puede hallar información de la configuración de visudo, que con un manejo adecuado permite crear un sistema de control del comando **SUDO** de forma eficiente.

Comandos de administración y más allá...

Lo mostrado hasta ahora es solo una pequeña introducción. Algunos comandos útiles para administración se pueden hallar en la siguiente liga: app_guide/unix_commands.html

=== Laboratorio ===

- Actividades a realizar en esta práctica se encuentran descritas en este documento, sus respuestas deben registrarse en :

- [Enlace al formulario](#)

