APUNTE 1. Introducción

SOFTWARE LIBRE / CÓDIGO ABIERTO

1. Una sesión de terminal

Para poder comunicarse con un sistema GNU/Linux es necesario, un nombre de presentación y una contraseña. Una sesión se inicia al entrar en el sistema y termina con la salida de éste. El usuario durante la sesión introducirá líneas de comandos que el sistema intentará ejecutar.

El sistema Unix identifica a los usuarios mediante un nombre de presentación (*login*) que está asociado a la cuenta de usuario, éste es asignado al usuario para que el sistema operativo sea capaz de gestionar los recursos compartidos del sistema (comprobar los derechos de acceso a archivos, utilización de comandos, etc.). Ligado a éste, todo usuario tiene asignada una contaseña (*password*) utilizada por el sistema para autenticar al usuario al inicio de la sesión.

Para iniciar una sesión es necesario poner el nombre de presentación y la contraseña. Por cuestiones de seguridad, cuando el usuario introduce la contraseña no habrá eco desde el sistema. Si la presentación del nombre y la clave ha sido la correcta, el sistema responderá mostrando una serie de mensajes en pantalla. Por último, se mostrará el caracter indicativo o "*prompt*" del intérprete de comandos (*shell*) que el administrador ha asignado al usuario.

Para salir del sistema, hay que poner el comando *exit* que indicará al sistema que debe eliminar todos los procesos asociados al usuario.

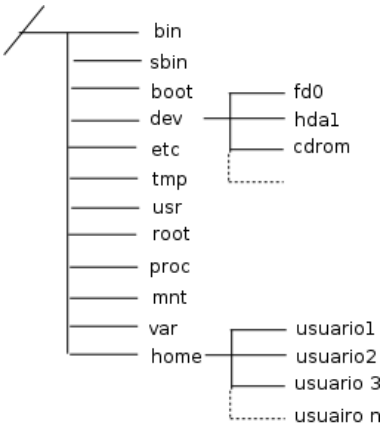
1.1. Lineas de Comandos

La primera componente de una línea de comandos introducida por un usuario debe ser el nombre de un comando, a continuación, y opcionalmente, se situarán los parámetros y modificadores que el usuario estime oportuno, debiendo estar de acuerdo con la sintaxis del comando. Un parámetro no es más que un elemento sobre el que actuará el comando y normalmente se corresponde con cadenas de caracteres o nombres de archivos. Un modificador es un elemento fijo, casi siempre una letra precedida del caracter "-", que permite especializar de alguna forma la función principal del comando. Los diferentes elementos que forman parte de la línea de comandos deben estar delimitados por uno o varios espacios en blanco o unos o varios caracteres de tabulación. Todas las líneas de comando terminan presionando la tecla <*enter*> que provoca la transmisión de la línea de comando al sistema para que éste la ejecute.

Por Ejemplo:

```
$ ls -l
$ find / -iname "imag*"
$ cp archivo1 copia_archivo1
$ date
```

Figura 1: Figu



2. El Sistema de Archivos

GNU/Linux **o**rganiza sus datos en archivos y directorios. La estructura básica del arbol de directorios generalmente es como lo muestra la Figura 1.

Figura 1. Esquema del Arbol de Directorios de GNU/Linux

La jerarquía del sistema de archivos proporciona un método sencillo para almacenar información. El directorio superior se conoce como raíz y se lo representa con una barra invertida "/". Para que todos los sistemas UNIX tengan características comunes, se ha establecido una serie de convenios sobre algunos directorios que deben estar presentes en el sistema. En la Figura1 se muestran estos directorios y a continuación se hace una breve descripción.

- bin: Se encuentran archivos binarios ejecutables: utilidades de configuración, manejo de archivos, etc.
- sbin: Se encuentran archivos binarios ejecutables para mantenimiento del sistema (solo pueden ser usados por root).
- boot: Directorio que contiene los archivos estáticos del cargador de arranque.
- dev: Contiene los archivos de dispositivos que representan a los dispositivos de hardware con los que interactua el sistema.
- etc: Contiene la mayoría de los archivos de configuración del sistema.
- tmp: Contiene los archivos temporales del sistema.
- usr: En este directorio se guardan aplicaciones de usuario, documentación, el código fuente, librerías, etc.
- root: En este directorio, el administrador guarda sus aplicaciones, configuraciones personalizadas, documentación, etc. Solo él puede accederlo.
- proc: Contiene información de uso interno del sistema.
- mnt o media: Se utiliza para montar sistemas de archivos que están en otras particiones/unidades.
- var: Contiene, entre otras cosas, definiciones internas del sistema.

- home: Contiene todos los archivos de los usuarios. Cada usuario posee un directorio personal, donde guarda sus aplicaciones, configuraciones personalizadas, documentacion, etc.
- lib: Directorio que contiene librerias compartidas y modulos del kernel escenciales para el funcionamiento del sistema.

2.1. Trayectorias absolutas y relativas

Una trayectoria no es más que un camino capaz de designar a un archivo o directorio. Dependiendo de dónde se inicie la trayectoria se puede clasificar en absoluta o relativa.

- Trayectoria absoluta: comienza siempre especificando el directorio raíz y sigue con los directorios hasta llegar al archivo o directorio que se desea nombrar.

```
segundo@home$ cat /home/segundo/practico1/ejercicio9.txt
segundo@home$ cat /etc/inittab
```

- Trayectoria relativa: comienza con el directorio actual de trabajo como punto de partida para especificar el archivo o directorio que se desea.

```
segundo@home$ cat segundo/practico1/ejercicio9.txt
segundo@home$ cat ../etc/inittab
```

Trayectorias Absolutas	Trayectoria Relativas
Comienza simpre en /	Comienza siempre en el directorio actual
Es única en toda la jerarquía	No es única en la jerarquía
Suele ser más larga y dificil de recordar	Suele ser mas corta y fácil de recordar
Es válida en cualquier directorio	No es válida en cualquier directorio

2.2. Gestión de Archivos

Cualquier usuario puede conocer su directorio actual de trabajo usando el comando *pwd* (present word directory).

```
$pwd
```

La orden que permite modicar el directorio actual de trabajo es *cd*

```
$ cd <directorio>
$ cd : Sin parámetros regresará al "home" del usuario.
$ cd ~ : Idem anterior. Los ejemplos a continuación son equivalentes si el usuario
es “segundo”.
    $ cd /home/segundo/programación/C
    $ cd ~/programación/C
$ cd / :Irá directamente a la raíz
$ cd ..  :Accederá a la carpeta/directorio inmediatamente superior en jerarquía.
```

El comando ls se utiliza para listar el contenido de un directior

```
$ ls :Lista el contenido del directorio actual.
$ ls -l : Visualiza el contenido del directorio actual  a lo largo de la pantalla.
$ ls /usr: Visualiza el contenido del directorio /usr.
$ ls ../ : Visualiza el contenido del directorio inmediatamente superior en jerarquía.
```

Un directorio es un archivo el cual tiene la propiedad de poder almacenar, archivos y directorios. Existe una forma muy sencilla de crear directorios usando el comando <*mkdir NombreDirectorio*>, por ejemplo:

```
$ mkdir practicos: Crea un directorio llamado practicos.
```

Para borrar directorios se usa el comando *rmdir* <*NombreDirectorio*> y para renombrar es *mv* <*nombreactual*> <*nombre nuevo*>

```
$ mv practicos practica
$ rmdir practica
```

3. Montaje de Particiones

En GNU/Linux para utilizar una partición/unidad es necesario montarla previamente. Para realizar este montaje se utiliza el comando *mount* (*de esta forma solo puede ser utilizado por el usuario root.*).

```
mount -t <tipo de particion> <dispositivo> <directorio>
```

Por ejemplo:

```
mount -t vfat /dev/floppy0 /mnt/floppy
mount -t ntfs /dev/hda1 /mnt/windows
mount -t iso9660 /dev/cdrom /mnt/cdrom
```

Para saber que particiones/unidades están montadas se utiliza el comando df.

```
$ df -h
S.ficheros          Tamaño Usado  Disp Uso% Montado en
/dev/hda3              11G  4,6G  5,2G  48% /
tmpfs                  47M     0   47M   0% /dev/shm
/dev/fd0              1,4M  618K  806K  44% /media/floppy0
```

3.1. Montaje Automático de Particiones

Si se desean realizar montajes automáticos al iniciar el sistema, o si se quiere permitir a los usuarios hacer montajes de algunas particiones y/o unidades, se debe utilizar el archivo de configuración *fstab.*

El archivo fstab se encuentra en */etc/fstab* y tiene la siguiente forma:

```
# /etc/fstab: static file system information.
#
# <file system> <mount point> <type> <options> <dump> <pass>
 /dev/hda3         /          ext2    defaults,errors=remount-ro   0  1
  /dev/hda6        none        swap                sw              0  0
   /dev/hdb    /media/cdrom0   iso9660      user,exec,dev,auto      0  0
   /dev/fd0   /media/floppy0    auto      noauto,user,noexec,rw     0  0
```

La primera columna lleva configurada la ruta del dispositivo a montar. En la segunda columna se define el directorio donde va a ser montado. En la tercera columna se define el tipo de partición. En la cuarta columna van una serie de opciones (ver subsección 3.2). En la quinta columna, se determina si se va a hacer un backup con los errores en tiempo de sistema (*0 desactivado*, *1 activado*). Y, en la sexta columna, indica si se va a comprobar la integridad del disco cada *x* desmontadas o al desmontarlo incorrectamente (*0 desactivado*, *1 activado*). De esta forma, por ejemplo, un usuario puede montar un disquete para lectura/escritura usando el comando *mount* de la siguiente forma

```
mount /media/floppy0
```

3.2. Opciones de Montaje

- *async*: Acelera los sistemas de archivos NFS. Es propenso a la pérdida de datos.
- *auto*: El dispositivo se monta automáticamente. Contrario: *noauto*.
- dev: Otorga permiso a los nodos de los dispositivos del sistema. Contrario: nodev.
- remount: Permite remontar particiones sin interrumpir sus tareas.
- ro: read-only.
- rw: read-write.
- suid: Permite que los UID y GID tengan efecto sobre los archivos.
- sync: Espera luego de hacer una llamada de escritura al hardware. Menos propenso a errores que async.
- user: El dispositivo puede ser montado los usuarios.

4. Inicio del Sistema

4.1. El proceso de booteo

Una vez que el kernel fue cargado en memoria se comienza con el proceso de inicialización de los distintos dispositivos de la máquina. A medida que son cargados y habilitados se va montrando un informe de estado por pantalla.

4.2. Init: Niveles de Ejecución

Luego de la carga y la ejecución del kernel, el sistema comienza a ejecutar una serie de proecesos elementales. El responsable de esta etapa es el proceso *init*.

Init basa su ejecución en un archivo localizado en */etc/inittab* en el cual se encuentran detallados los niveles de ejecución (*run levels*) y sus procesos.

Cada nivel de ejecución es una configuración de software del sistema en la que sólo existe un grupo de procesos.

Los niveles de ejecución son los siguientes:

```
0 - Parado
1 - Modo Monousuario
2 - Modo Multiusuario, sin NFS.
3 - Modo multiusuario completo.
4 - Sin usar.
5 - X11.
6 - Reiniciar
```

4.3. Acciones de Init

- respawn: Si el proceso no existe, lo crea y no espera que termine (continúa con la ejecución del inittab). Si el proceso existe, no hace nada. Cuando el proceso muere, se vuelve a ejecutar.
- wait: Ejecuta un proceso y espera a que termine.
- once: Crea el proceso y continúa con la lectura de inittab. Cuando muere, vuelve a ejecutarlo.

- boot: Crea el proceso solamente durante la etapa de booteo. Cuando muere, no vuelve a ejecutarlo.
- bootwait: Se crea el proceso cuando realiza el cambio entre modo monousuario y modo multiusuario.
- powerfail: Ejecuta el proceso solamente cuando init recibe una señal de falla de alimentación (SIGPWR).
- powerwait: Ejecuta el proceso solamente cuando init recibe una señal de falla de alimentación y espera a que concluya antes de continuar con los demás procesos.
- off: Si el proceso se está ejecutando, envia una señal de advertencia (SIGTERM) y espera 20 segundos antes de enviarle la señal de muerte (SIGKILL). En caso de que el proceso no exista, esta acción es ignorada.
- initdefault: Definel el nivel de ejecución en que se iniciará el sistema por defecto.
- sysinit: Los procesos que lleven esta acción se ejecutarán antes de que init intente accder a la consola para iniciar al sesión login.

Una sección importante del archivo inittab es la siguiente:

```
10:0:wait:/etc/rc.d/rc 0
11:1:wait:/etc/rc.d/rc 1
12:2:wait:/etc/rc.d/rc 2
13:3:wait:/etc/rc.d/rc 3
14:4:wait:/etc/rc.d/rc 4
15:5:wait:/etc/rc.d/rc 5
16:6:wait:/etc/rc.d/rc 6
```

Esta sección ejecuta todos los archivos rc del directirio /etc/rc.d. Estos archivos son scripts que ejecutan diferentes tareas del sistema.