

# Capítulo V: Mis primeras tareas de administración desde la consola Linux:

## ÍNDICE DE CONTENIDOS:

- 5.1 El intérprete de comando II
- 5.2 Colección de comandos I:
  - 5.2.1 Los relativos a directorios y archivos
  - 5.2.2 Los relativos a procesos, usuarios y grupos

Esta obra está protegida por la **Licencia Creative Commons**, bajo las condiciones de: **Reconocimiento - No comercial - Compartir igual**: El material creado por un artista puede ser distribuido, copiado y exhibido por terceros si se muestra en los créditos. No se puede obtener ningún beneficio comercial y las obras derivadas tienen que estar bajo los mismos términos de licencia que el trabajo original.



**Reconocimiento (Attribution)**: El material creado por un artista puede ser distribuido, copiado y exhibido por terceras personas si se muestra en los créditos.



**No Comercial (Non commercial)**: El material original y los trabajos derivados pueden ser distribuidos, copiados y exhibidos mientras su uso no sea comercial.



**Compartir Igual (Share alike)**: El material creado por un artista puede ser modificado y distribuido pero bajo la misma licencia que el material original.

**Versión: 1.0**

**Autor: Alberto Reynolds Moreno.**

alberto.reynolds@gmail.com

**Revisión: Isabel Rueda Rodríguez**

rueda.isabel@gmail.com

Seguimos en este capítulo tratando, a modo de puente, el tema del intérprete de comandos, para enlazar con las primeras ordenes de administración desde una consola o terminal.

Como ya hemos visto anteriormente, el intérprete de comandos será ese terminal donde un administrador de sistemas Linux pasará la mayor parte del su tiempo de trabajo, así que, merece la pena volver a hablar de él.

El intérprete de comandos es el equivalente al MS-DOS de Microsoft, aunque en realidad, es bastante más potente, pero creados con la misma finalidad. Existen diferentes intérpretes de comandos en Linux, los más conocidos son: **bash** y **sh**.

Recordemos que al abrir un intérprete de comandos, nos encontraremos con un **prompt** (solicitud) parecido al siguiente:

```
[albertux@debian]$
```

Por lo general, el intérprete de comandos aparecerá de la siguiente forma: nombre\_usuario@nombre\_máquina, seguido de \$ o #.

El símbolo \$ significa que soy un usuario final (sin privilegios), mientras que el símbolo # nos avisa de que tenemos abierta una sesión como 'root'.

Para pasar de usuario normal a 'root', basta con teclear la palabra 'su' y hacer intro para introducir la clave de administrador. Para salir teclearemos la palabra 'exit' y para cambiar a cualquier otro usuario del sistema escribiremos 'su nombre\_del\_usuario'.

```
[albertux@debian]$ su
contraseña:
```

**Nota:** Los sistemas Unix/Linux son **case sensitive**, es decir, diferencian entre mayúsculas y minúsculas, por lo que no será lo mismo teclear 'su' que 'SU'.

## 5.1 Personalizando nuestro bash:

Vamos a utilizar bash como nuestro intérprete de comandos a usar. Podremos personalizar este intérprete de comandos, del modo que más útil nos sea.

Ejecutando el comando **'alias -p'**, veremos cómo lo tenemos definido.

```
[albertux@debian]$ alias -p
alias l = 'ls -CF'
alias la = 'la -A'
alias ll = 'ls -l'
alias ls = 'ls --color=auto'
```

El archivo donde configurar nuestro bash se llama 'bashrc' y está oculto en cada /home de usuario. Podemos editarlo con cualquier editor de textos.

```
[albertux@debian]$ nano /home/usuario/.bashrc
```

Si nos fijamos, existen definidos cierto alias que hacen referencia a una orden concreta. Según la configuración anterior, si tecleo en mi intérprete de comandos la orden **'la'**, este alias ejecutará el comando **'ls -A'**, cuyo ejecución provoca un listado de todos los archivos y

directorios, incluido los ocultos.

Según esto, nosotros podremos incluir alias a nuestro antojo, y según nuestras necesidades.

Un ejemplo curioso podría ser el crearnos un alias que nos mostrara todos los procesos que están ejecutándose en mi sistema. Para ello, editaremos este archivo y crearemos el alias 'procesos' el cuál realizará la orden: **'ps aux'**

```
alias procesos = 'ps aux'
```

Únicamente bastará con reiniciar mi consola o terminal para ver el resultado.

## 5.2 Colección de comandos I:

Una vez estamos algo más familiarizados con nuestro intérprete de comandos, es buen momento para comenzar a conocer algunos comandos básicos que me permitan administrar mi máquina.

El comando por excelencia y más importante de todos es el comando '**man**'. Este comando es la base y guía del resto de comandos, ya que nos ayudará a conocer la sintáxis correcta y opciones de cualquier otro comando.

Anteriormente hemos usado en nuestro alias el comando 'ps'. Si queremos conocer la sintáxis y el resto de opciones para este comando haremos:

```
[albertux@debian]$ man ps
```

Un intérprete de comandos no tiene sentido si no sabemos sacarle su "jugo", es decir, nos interesa conocer una serie de comandos básicos para darle funcionalidad.



Fuente: [www.tiraecol.net](http://www.tiraecol.net)

### 5.2.1 Los relativos a directorios y archivos:

Recordemos antes un poco, cómo es el árbol de directorios de Linux:

```

/
|_ root
|_ home
|   |_ usuario1
|   |_ usuario2
|       |_ documentos
|_ boot
|_ .....
    
```

Como vemos, todo pasa por el directorio raíz, el directorio /. La orden para movernos de un directorio a otro será: '**cd**' y su sintáxis básica es el siguiente:

```
cd /ruta/del/directorio
```

Podemos usar la ruta absoluta (partiendo del directorio raíz) o la ruta relativa. Imaginemos que estamos en el directorio /home y queremos llegar a un subdirectorio llamado usuario1.

Podemos utilizar la ruta absoluta:

```
[albertux@debian]$ cd /home/usuario1
```

o utilizar la ruta relativa: (en el caso de estar ya en /home sería):

```
[albertux@debian]$ cd usuario1
```

Al estar ya situados en el directorio /home, no es necesario pasar antes por el directorio raíz (/), ni por home, directamente apuntaremos al directorio de destino.

Sería un error estando en '/home' escribir '**cd /usuario1**', ya que estamos intentando llegar a un directorio llamado 'usuario1' que no cuelga del directorio raíz '/', sino del directorio 'home'.

**Nota:** En caso de no saber exactamente en qué directorio estamos es más recomendable utilizar siempre la ruta absoluta.

Dentro de todo directorio existen dos directorios especiales que son '.' y '..'.

El primero hace referencia al directorio actual, es decir, si haces '**cd .**' te quedas donde estás (el directorio especial '.').

El segundo hace referencia al directorio padre, o sea, si estamos en /home/usuario1 y hacemos '**cd ..**' terminaremos en /home.

```
[albertux@debian]$ cd .
[albertux@debian]$ cd ..
```

Si nos hemos perdido por movernos entre tantos directorios... ¡¡que no cunda el pánico!!., tenemos el maravilloso comando "**pwd**" que nos "*chiva*" donde estamos situados.

```
[albertux@debian]$ pwd
[albertux@debian]$ /home/usuario1/documentos
```

Algunos comandos básicos para comenzar a manejar nuestro intérprete de comandos pueden ser:

COMANDO	ACCIÓN
ls	Igual que el dir. Listado de archivos y directorios. <i>Ej.: ls -la (muestra los ocultos y con detalles)</i>
cp	Copia archivos. Con -R copia en modo recursivo los directorios: <i>Ej.: cp -R /home/usuario1 /home/usuario2</i>

mv	Mueve directorios y archivos: <i>Ej.: mv /home/usuario1 /tmp</i>
rm	Elimina archivos. Con -R elimina directorios de modo recursivo: <i>Ej.: rm -R /tmp/directorio</i>
mkdir	Crea un directorio nuevo: <i>Ej.: mkdir mi_directorio</i>
df	Nos indica la cantidad de espacio de disco ocupado: <i>Ej.: df -h</i>

Como ya hemos dicho antes, **TODOS** los comandos pueden apoyarse en el comando 'man' para obtener su ayuda.

### 5.2.2 Los relativos a procesos, usuarios y grupos:

Podemos utilizar diferentes comandos de manera útil con diferentes finalidades.

Por ejemplo el comando '**cat**' nos da la posibilidad de pasar a un archivo el contenido de una salida.

```
[albertux@debian]$ cat > compra-para-barbacoa.txt
patatas fritas
vino
cervezas
cocacola
panceta
pan
```

Esto nos guardará en el archivo 'compra-para-barbacoa.txt' toda la lista anteriormente escrita.

Para verlo:

```
[albertux@debian]$ cat compra-para-barbacoa.txt
```

O cualquier otro editor de textos bajo terminal, por ejemplo:

```
[albertux@debian]$ nano compra-para-barbacoa.txt
```

Utilizaremos la opción '>>' para sobrescribir sobre lo anterior sin borrar nada.

Se nos olvidaba lo básico...

```
[albertux@debian]$ cat compra-para-barbacoa.txt
carbón
```

### PROCESOS:

Interesante también es el uso del comando '**ps**', que nos mostrará los procesos que están corriendo en nuestro sistema. Para saber más:

```
[albertux@debian]$ man ps
```

Así vemos que con '**ps aux**' nos muestra todos los procesos y sus PID (*Identificador de Procesos*). Si recordamos este comando es el que utilizamos en el alias 'procesos' anteriormente.

Si queremos guardar este listado de procesos en un archivo de texto 'procesos.txt', ya sabemos como hacerlo:

```
[albertux@debian]$ ps aux > procesos.txt
```

Recordando más...

Para renombrarlo:

```
[albertux@debian]$ mv procesos.txt procesos-activos.txt
```

Para borrarlo:

```
[albertux@debian]$ rm procesos.txt
```

Otro comando interesante es el comando '**grep**' que nos ayudará en muchas ocasiones para filtrar y buscar una cadena determinada.

Imaginemos que quiero buscar el proceso que está ejecutando mi navegador web mozilla (*iceweasel*). Para ello combinaremos '**ps**' y '**grep**' con las famosas tuberías (|) de este modo:

```
[albertux@debian]$ ps aux | grep iceweasel
usuario1 1787 0.0 3332 760 pts/0 R+ 12:02 0:00 grep iceweasel
```

Vemos que el PID de su proceso es el 1787. Imaginemos que como administrador vemos que este proceso nos ha fallado o nos está dando problemas. Podemos matarlo con la orden '**kill**'.

```
[albertux@debian]$ kill 1787
```

Otros comandos muy útiles para un administrador de Linux son los relativos a la gestión de permisos, usuarios y grupos de archivos y directorios:

COMANDO:	ACCIÓN:
chmod	Cambia los permisos de un archivo. Con '-R' los hará de modo recursivo a los directorios. <i>Ej.: chmod -R 755 procesos.txt</i>
chown	Cambia de propietario a un archivo. Con '-R' lo aplica de modo recursivo a los directorios. <i>Ej.: chown -R alberto /home/alberto</i>
chgrp	Cambia de grupo a un archivo. Con '-R' lo aplica de modo recursivo a los directorios. <i>Ej.: chgrp -R www-data /home/alberto</i>
adduser	Añade un nuevo usuario.
deluser	Elimina un usuario existente.
addgroup	Añade un nuevo grupo de usuarios.
delgroup	Elimina un grupo de usuarios ya existente.

En sucesivos capítulos trataremos con más profundidad la administración de usuarios, grupos y permisos para mantener una política de seguridad precisa en nuestros equipos.