educación IT
CODE YOUR FUTURE

Introducción a Linux

Módulo 2

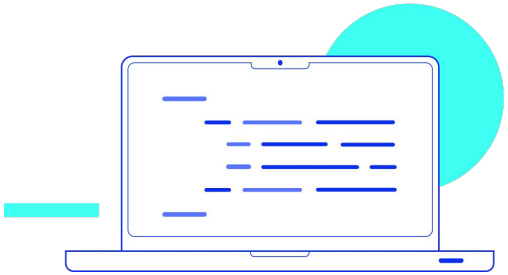
Algunos comandos básicos

El comando mkdir

Este comando se utiliza para **crear directorios.**

```
# mkdir testing
# ls -ld testing/
drwxr-xr-x 2 root root 4096 Jun 12 02:56 testing/
```

Otro caso es si queremos crear un directorio que se encuentra dentro de otro. Es necesario tener en cuenta que si ninguno de los directorios que ponemos existe nos dará error.



Ejemplo con error:

```
# mkdir es/otro/directorio/lejano
mkdir: cannot create directory `es/otro/directorio/lejano': No such
file or directory
```

Ejemplo para crear un directorio y una cadena de subdirectorios:

```
# mkdir -p es/otro/directorio/lejano
# ls -ld es/otro/directorio/lejano/
drwxr-xr-x 2 root root 4096 Jun 12 02:57 es/otro/directorio/lejano/<
```

El comando cp

El comando **cp** se utiliza para **copiar archivos/directorios.**

Sintaxis

```
cp [ opciones ] archivo1 archivo2
cp [ opciones ] archivos directorio
```

Opciones más frecuentes:

`-f`: fuerza la reescritura si ya existe el destino.

`-i`: habilita el "*prompt*" interactivo para evitar la sobreescritura.

`-p`: preserva todos los permisos de la estructura del directorio y/o archivo, incluyendo la fecha de creación.

`-r,-R`: hace una copia recursiva, para poder copiar todo un directorio y sus subdirectorios.

`-v`: modo "*verbose*", hace que el comando muestre detalles adicionales de lo que se hace, por ej., mostrar qué archivos copia y hacia dónde.

Entonces, lo que se puede hacer es copiar un archivo para que en el destino tenga otro nombre; también podemos copiar varios archivos a un directorio.

```
# cp archivo1 archivo2
```

El resultado fue la copia del `archivo1` al `archivo2`.

```
# ls -l archivo2
-rw-r--r-- 1 root root 0 Jun 5 05:41 archivo2
```

El archivo creado es nuevo (se nota por la fecha).

Para copiar varios archivos a un directorio.

- Primero, crear un directorio.
- Segundo, copiar los dos archivos (`archivo1`, `archivo2`) al directorio creado.

```
# mkdir dir1
# cp archivo1 archivo2 dir1
# ls -l dir1
-rw-r--r-- 1 root root 0 Jun 5 05:43 archivo1
-rw-r--r-- 1 root root 0 Jun 5 05:43 archivo2
```

Atención al error que aparece. Esto se debe a que le dimos un *path* (una ruta) absoluto y el directorio `dir1` no existe en `/dir1`, sino que está en `/root/dir1` porque fue creado en nuestro home.

Luego, al ejecutar `ls` al directorio `dir1`, tampoco usamos un *path* absoluto (`/root/test/dir1`, sino `ls -l dir1`) porque, precisamente, estamos parados dentro de un directorio que contiene a "`dir1`".

Con el comando **pwd**, nos dirá en cuál directorio estamos parados.

```
# pwd
/root/test
# ls -l
total 4
-rw-r--r-- 1 root root 0 Jun 5 05:41 archivo1
-rw-r--r-- 1 root root 0 Jun 5 05:41 archivo2
drwxr-xr-x 2 root root 4096 Jun 5 05:43 dir
```

Ejemplos

Funcionamiento de la opción **-v**.

```
# cp -v archivo1 archivo2
`archivo1' -> `archivo2'
```

Aquí agregamos el **-i** ,para que vean que nos preguntará si queremos sobreescribir.

```
# cp -i archivo1 archivo2
cp: overwrite `archivo2'?
```

Aquí vemos el funcionamiento de la opción **-p**.

Listamos con **-l** para que vean los permisos y la fecha.

```
# ls -l archivo1
-rw-r--r-- 1 root root 0 Jun 5 05:41 archivo1
# date
Sun Jun 5 05:54:02 ART 2011
```

Ahora lo copiamos.

```
# cp -p archivo1 nuevoarch
# ls -l nuevoarch
-rw-r--r-- 1 root root 0 Jun 5 05:41 nuevoarch
```

Y como ven, el archivo nuevo conserva la misma fecha; distinto sería si hubiésemos hecho esto.

```
# cp archivo1 narch
# ls -l narch
-rw-r--r-- 1 root root 0 Jun 5 05:55 narch
```

Aquí vamos a copiar recursivamente (un directorio y todo su contenido):

```
# cp -r /etc/ dir1/
# ls -l dir1/
total 4
-rw-r--r-- 1 root root 0 Jun 5 05:43 archivo1
-rw-r--r-- 1 root root 0 Jun 5 05:43 archivo2
drwxr-xr-x 34 root root 4096 Jun 5 05:57 etc
```

Al ejecutar **`ls -l dir`**, vemos que está el directorio "`etc`", pero la fecha es reciente. Si usamos la opción **`-p`** se mantienen la fecha y demás privilegios.

Manual: **`man cp`**

Comando mv

Este comando se utiliza para **mover o renombrar archivos y directorios.**

Sintaxis

```
mv [opciones] origen destino
```

Opciones más usadas:

`-f`: si ya existe lo fuerza a sobreescribir.

`-i`: activa el modo interactivo para aceptar los cambios uno por uno.

Ejemplo para cambiar de nombre:

```
# mv archivo1.txt nuevonombre.txt
```

Ejemplo para mover:

```
# mv archivo1.txt /directorio
```

Comando touch

Este comando **cambia la fecha/hora de acceso o modificación** de los archivos.

Sintaxis

```
touch [opciones] archivo
```

Opciones

`-a`: solo cambia el tiempo de acceso.

`-m`: cambia el de modificación.

`-t`: timestamps definir la fecha a utilizar [[CC]YY]MMDDhhmm[.ss].

Por ejemplo la fecha y hora para el 12 de Enero 2001 a las 18:45 es 200101121845

Ejemplo:

```
# ls -l archivo1
-rw-r--r-- 1 root root 0 Jun 5 05:41 archivo1
# touch archivo1
# ls -l archivo1
-rw-r--r-- 1 root root 0 Jun 12 04:52 archivo1
```

```
# ls -l archivo2
-rw-r--r-- 1 root root 0 Jun 12 04:53 archivo2
# touch -t 200101121845 archivo2
# ls -l archivo2
-rw-r--r-- 1 root root 0 Jan 12 2001 archivo2
```

Veamos un ejemplo usando metacaracteres:

```
# touch hola{1,2,4,5}
hola1 hola2 hola4 hola5
```

Comando rm

Este comando sirve para **borrar tanto directorios como archivos.**

Sintaxis

```
rm [opciones] archivos
```

Opciones

-f: hace un borrado sin preguntar nada.

-i: en modo interactivo.

-r,-R: para borrar recursivamente.

Ejemplo para borrar un directorio y TODO su contenido:

```
# rm -rf /directorio
```

Comando rmdir

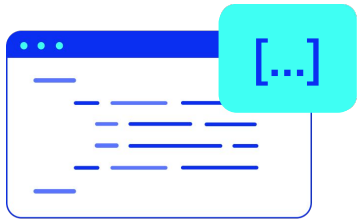
Se utiliza solamente para **borrar directorios vacíos.**

Sintaxis

```
rmdir [opciones] directorio
```

```
# rmdir dir
```

Nota: generalmente se utiliza el comando `rm -rf`.



educación IT
CODE YOUR FUTURE

¡Sigamos trabajando!