LINUX: BACKUP POR COMANDOS

Juan Carlos Navidad García

Seguridad Informática

¿Qué es un backup?:

Un **backup** es una **copia de seguridad** de los datos realizada en un soporte de almacenamiento adecuado (un disco duro externo, por ejemplo). Al hacer un **backup**, se crea una **copia de seguridad** de los datos a partir de la cual se pueden **restaurar posteriormente en caso de pérdida**.

En esta práctica vamos a realizar un **backups** por comandos en Ubuntu 18.04.

¿Cómo realizar el backup?:

Para Linux vamos a probar la herramienta **rsync**. Es una herramienta sencilla pero muy potente. Básicamente nos permite sincronizar directorios en una misma máquina o entre dos máquinas

Abrimos la terminal he iniciamos como **root** mediante **sudo -i**, **sudo su** o aplicamos todos los comandos con el sudo.

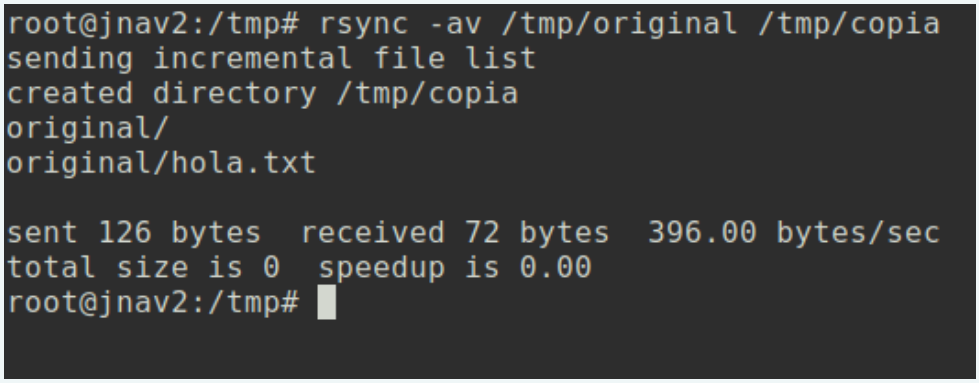
Nos vamos al directorio **/tmp** y creamos un directorio llamado **original** con un fichero **hola.txt.**

```
root@jnav2:/tmp# mkdir original
root@jnav2:/tmp#
```

```
root@jnav2:/tmp/original# touch hola.txt
root@jnav2:/tmp/original#
```

Volvemos al directorio **/tmp** y creamos una copia de original mediante el comando:

"**rsync -av /tmp/original /tmp/copia**"



Como se puede ver en la imagen, la herramienta nos avisa de que va a crear el directorio **/tmp/copia** (no lo habíamos creado) y muestra los ficheros que ha traspasado y un resumen de bytes transferidos.

El directorio **/copia** reproduce la estructura de carpetas de original, no solo los ficheros.

¿En qué se diferencia de hacer una copia normal mediante cp? Pues en que **rsync** no copia todo, sino solo los ficheros nuevos o los que han cambiado. Por ejemplo, creamos un fichero nuevo y sincronizamos. Solo se traspasa ese fichero, no traspasa toda la carpeta de nuevo sobrescribiendo lo que ya había.

Si creamos otro archivo en **/original** y actualizamos con el mismo comando que hemos utilizado anteriormente, la carpeta actualizada con el archivo creado se añade a **/copia**:

```
root@jnav2:/tmp# date > original/adios.txt
root@jnav2:/tmp# rsync -av /tmp/original /tmp/copia
sending incremental file list
original/
original/adios.txt

sent 188 bytes  received 39 bytes  454.00 bytes/sec
total size is 29  speedup is 0.13
```

```
root@jnav2:/tmp# ls copia/original/
adios.txt  hola.txt
root@jnav2:/tmp#
```

Si hemos borrado un fichero en **/original** y queremos que se actualice la copia, hay que incluir el parámetro

"**—delete**"

Con lo que hemos visto hasta ahora solo podemos hacer **backups completos**. El directorio **/copia** lo podemos llevar a cualquier dispositivo extraíble o podría ser un disco en red. Para hacer **backups incrementales** ejecutaremos el siguiente comando:

"**rsync -avvb --delete --backup-dir=tmp/backup1 /tmp/original /tmp/copia**"

```
root@jnav2:/tmp# rsync -avvb --delete --backup-dir=tmp/backup1 /tmp/original /tmp/copia

sending incremental file list
(new) backup_dir is tmp/backup1
delta-transmission disabled for local transfer or --whole-file
original/hola.txt is uptodate
total: matches=0  hash_hits=0  false_alarms=0 data=0

sent 84 bytes  received 156 bytes  480.00 bytes/sec
total size is 0  speedup is 0.00
root@jnav2:/tmp#
```

Esta vez la sincronización deja en el directorio **/tmp/backup1** los ficheros que resultan modificados o eliminados; en **/tmp/copia** siempre está la versión actual.

En nuestro ejemplo vamos a borrar el fichero **adios.txt** y al sincronizar vemos que ya no está en **/original** ni en **/copia**, pero sí en **/backup1**.

```
root@jnav2:/tmp# ls original/ && ls copia/original/ && ls backup1/original/
hola.txt
hola.txt
adios.txt
root@jnav2:/tmp#
```

Esto se debe a que hemos recuperado una **copia incremental**, es decir, hemos recuperado la última modificación del **backup**.

Los atributos **-v** significa que queremos que nos **detalle la información** de lo que realiza el comando, **-b** significa que realizamos un **backup** y **-a** significa que estaremos trabajando con **archivos**.

Finalmente, como es imprescindible que el **backup** se ejecute con regularidad, vamos a probar a meterlo en el **cron**. Este script está pensado para que se ejecute **cada minuto** (lo normal sería una vez al día), pero nosotros crearemos el **script básico**. Crearemos un **script mibackup.sh** que se invoca desde el **cron**. Para distinguir las distintas **copias incrementales**, el script utiliza la fecha en que se ejecuta. Dejamos un **log** para comprobar diariamente que todo ha ido bien.

El script tendría la siguiente forma:

FECHA=`date +%y%m%d%H%M`

rsync -avvb --backup-dir=/tmp/backup_$FECHA --delete /tmp/original /tmp/copia >> /tmp/log_$FECHA

```
root@jnav2:/tmp# cat mibackup.sh
FECHA=`date +%y%m%d%H%M`
rsync -avvb --backup-dir=/tmp/backup_$FECHA --delete /tmp/original /tmp/copia >> /tmp/log_$FECHA
root@jnav2:/tmp#
```

Para ejecutarlo, debemos darle permisos de ejecución con el comando **chmod** y para arrancarlo, utilizaremos **./mibackup.sh**.

```
root@jnav2:/tmp# chmod 777 mibackup.sh
root@jnav2:/tmp# ./mibackup.sh
root@jnav2:/tmp#
```

Al ejecutarlo no nos saldrá nada, puesto que **redirecciona la salida del script en el archivo log<<fecha>>**.

Haremos un **ls** para comprobarlo:

Si nos fijamos, vemos que hay **tres logs**, es porque he ejecutado tres veces el script, pero si hacemos un **cat** para visualizar su interior, veremos lo siguiente: