

SWAP: Clonar la información de un sitio web

David Cabezas Berrido

dxabezas@correo.ugr.es

19 de marzo de 2021

Índice

1. Objetivos	2
2. Copiar archivos locales en equipos remotos: tar y scp	2
3. Sincronizar archivos locales en equipos remotos: rsync	3
4. Acceso sin contraseña con SSH	4
5. Programación de tareas: crontab	5

1. Objetivos

TODO

2. Copiar archivos locales en equipos remotos: tar y scp

Si queremos copiar el directorio `~/directorio` de la máquina 1 a la 2, tenemos varias opciones. Con el comando **tar**, realizamos en M1

```
tar czf - directorio | ssh dxabezas@192.168.56.102 'cat > ~/directorio.tgz'
```

El directorio es comprimido y con un cauce lo enviamos directamente al directorio `~` de la máquina 2, sin ocupar espacio extra en la máquina 1.

- La opción **c** (**create**) indica que queremos crear un nuevo archivo.
- La opción **z** (**gzip**) aplica la herramienta de compresión **gzip**.
- La opción **f** (**file**) indica el archivo a guardar.

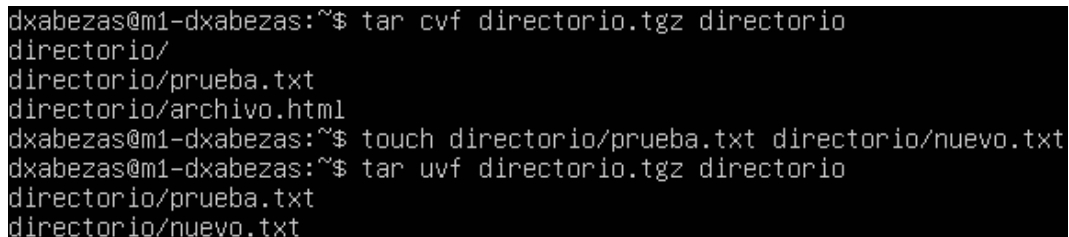
Ahora en la máquina 2 comprobamos con **ls** que el archivo ha sido recibido el archivo `~/directorio.tgz`, y procedemos a extraerlo con

```
tar -xzf directorio.tgz -C ~/
```

- La opción **x** (**extract, get**) indica que queremos extraer archivos.
- La opción **z** (**gzip**) aplica la descompresión de **gnupzip**.
- La opción **C** (**directory**) indica el directorio donde queremos extraer los archivos.

Opcionalmente, se puede usar la opción **v** (**verbose**) para que liste los archivos que va comprimiendo o extrayendo.

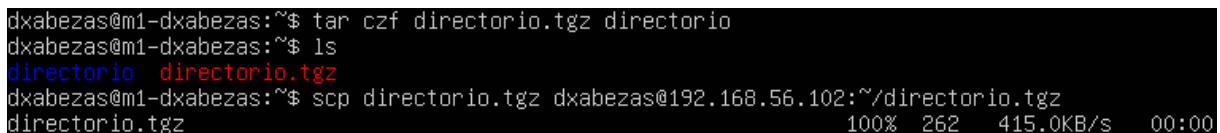
Señalamos también la opción **u** (**update**), que permite añadir al archivo `.tgz` sólo los archivos que han sido modificados desde la última copia. La desventaja es que no funciona en ficheros comprimidos.



```
dxabezas@m1-dxabezas:~$ tar cvf directorio.tgz directorio
directorio/
directorio/prueba.txt
directorio/archivo.html
dxabezas@m1-dxabezas:~$ touch directorio/prueba.txt directorio/nuevo.txt
dxabezas@m1-dxabezas:~$ tar uvf directorio.tgz directorio
directorio/prueba.txt
directorio/nuevo.txt
```

Figura 1: Suprimiendo la opción **z** por incompatibilidad, guardamos el contenido del directorio en un fichero `.tgz`. Editamos un archivo existente y creamos uno nuevo, y utilizamos **update** en lugar de **create** para que añada sólo los archivos recientes.

La otra alternativa es la herramienta **SCP** (Secure Copy), que usa SSH para hacer copias seguras de archivos.



```
dxabezas@m1-dxabezas:~$ tar czf directorio.tgz directorio
dxabezas@m1-dxabezas:~$ ls
directorio directorio.tgz
dxabezas@m1-dxabezas:~$ scp directorio.tgz dxabezas@192.168.56.102:~/directorio.tgz
directorio.tgz                                100% 262  415.0KB/s  00:00
```

Figura 2: En la máquina 1 comprimimos el directorio y después lo enviamos por SSH a la máquina 2 con SCP.

Ya podemos extraer los archivos en la máquina 2 como hicimos antes.

Esta opción tiene la desventaja de que la copia ocupa espacio en M1, podemos evitarlo utilizando:

```
dxabezas@m1-dxabezas:~$ scp -r directorio dxabezas@192.168.56.102:~/directorio
prueba.txt          100% 0      0.0KB/s   00:00
archivo.html        100% 136    82.6KB/s  00:00
```

Figura 3: Enviamos el directorio directamente desde M1 a M2, sin comprimir.

- La opción **r** indica que copie directorios enteros de forma recursiva.

Señalamos cuatro opciones de SCP que pueden sernos de utilidad:

- La opción **o** permite pasar opciones a SSH en el formato del fichero de configuración del cliente (**ssh.config**).
- La opción **P** permite seleccionar el puerto de SSH, es útil cuando no es el 22 (default).
- La opción **q** deshabilita los mensajes, útil si hay muchos archivos y no queremos que se nos llene la terminal.
- La opción **v**, como de costumbre, imprime mensajes sobre el progreso. Tanto de SCP como de SSH.

Notamos que en ningún momento nos pide la contraseña porque configuramos el acceso sin contraseña de SSH en la anterior práctica.

3. Sincronizar archivos locales en equipos remotos: rsync

Ejecutamos `sudo apt install rsync`, y descubrimos que ya está instalado en ambas máquinas.

Hacemos al usuario dueño del directorio `/var/www`:

```
sudo chown dxabezas:dxabezas -R /var/www
```

Desde la máquina 2, ejecutamos lo siguiente.

```
dxabezas@m2-dxabezas:~$ rsync -avz -e ssh dxabezas@192.168.56.101:/var/www/ /var/www/
receiving incremental file list
./
html/
html/ejemplo.html
html/index.html
html/info.php

sent 92 bytes  received 3,558 bytes  2,433.33 bytes/sec
total size is 11,081  speedup is 3.04
```

Figura 4: Desde M2, nos conectamos a M1 y sincronizamos los contenidos del directorio `/var/www`.

Ahora podemos conectarnos desde el navegador al servidor de Apache2 de M2, que sirve los contenidos que originalmente eran de M1.

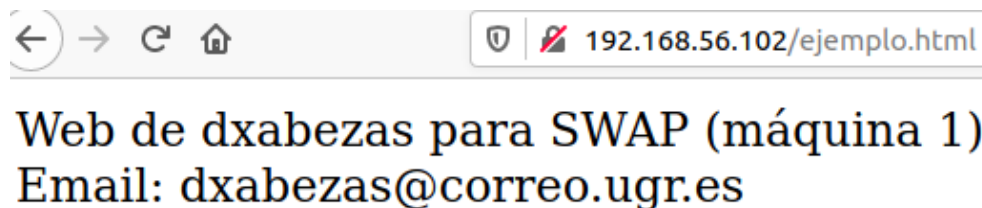


Figura 5: Ahora el directorio `/var/www` de M2 tiene los contenidos que originalmente eran de M1.

Explicamos el significado de las opciones:

- La opción **a** (**archive**) indica que usamos recursividad (para copiar los directorios completos) y preservar la estructura de archivos. Una excepción son los enlaces duros, que deben explicitarse con la opción **H**.
- La opción **v** para verbose.

- La opción **z** para que los archivos se transfieran comprimidos.
- La opción **e** para especificar el shell remoto a utilizar, por defecto SSH.

Otras opciones que pueden ser útiles:

- La opción **p** para preservar los permisos, implícito en **a**.
- La opción **r** para recursividad en los directorios.
- La opción **t** para preservar los tiempos de modificación originales, también implícito en **a**.

La **--delete** indica que se borren de M2 los archivos que se hayan eliminado en M1, de tal forma que la copia sea idéntica. Por último, la opción **--exclude** se usa para no copiar ciertos archivos, por ejemplo: **--exclude=**/patata.txt** no copiará el fichero **/var/www/patata.txt** de M1 a M2.

Igualmente, tampoco se nos ha solicitado la contraseña del usuario por la configuración de SSH que realizamos en la anterior práctica.

4. Acceso sin contraseña con SSH

En la anterior práctica configuramos el acceso sin contraseña por SSH con la utilidad **ssh-copy-id**. Hablaremos de algunas opciones para esta tarea y explicaremos la alternativa manual.

Cuando se genera la clave con **ssh-keygen**, la opción **-t** se utiliza para indicar el tipo de clave: **rsa**, **dsa** (digital signature algorithm, para firma digital), **ecdsa** (elliptic curve DSA). La opción **-f** se utiliza para indicar el fichero con la clave, si la omitimos lo preguntará luego. Por defecto el fichero con la clave privada es **~/.ssh/id_rsa**, y para la clave pública se añade la extensión **.pub**. También la opción **-b** permite especificar la longitud (en bits) de la clave, por defecto 2048.

En caso de no utilizar una clave en el archivo por defecto, debemos indicar con la opción **-i** la ruta de la clave tanto para la copia de la clave pública:

```
ssh-copy-id -i /ruta/clave.pub usuario@ip
```

como para la privada al acceder:

```
ssh -i /ruta/clave usuario@ip
```

Por último, podemos visualizar la fingerprint y el randomart de la clave con la siguiente orden:

```
dxabezas@m1-dxabezas:~/.ssh$ ssh-keygen -lv
Enter file in which the key is (/home/dxabezas/.ssh/id_rsa):
2048 SHA256:PIIwIXF2Y0+IYNx69vL1yJ+HK61qhbR9XFfWyata0ss dxabezas@m1-dxabezas (RSA)
+---[RSA 2048]-----+
|
|=00+0. . 0
|0+0+0+0 *
|. 0 +.. + .
| 0 * E + 0
|. * * + S
|. = + = .
| + 0 0 .
| . ..0 + .
| . ...0+0+
|-----[SHA256]-----+
```

Figura 6: La opción **-l** permite visualizar la fingerprint de una clave existente, y añadir **-v** permite visualizar su randomart.

Finalmente, configuraremos el acceso sin contraseña manualmente. Para ello, deshaceremos primero la configuración automática de la práctica 1, lo que ya dará una pista de como se hace la configuración manual. Las claves públicas guardadas están en el fichero **~/.ssh/authorized_keys** del servidor (M1), ahí encontramos la que guardamos con **ssh-copy-id**.

```

dxabezas@m1-dxabezas:~/.ssh$ cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDC2bJ67hu8ddsNhN86fyGui+x0Vkk0euCdxLt3dL6147fHtCLzuAoaJpIPD43a
qeL59hVIZuGyA5n04S3u2W2IH3skJ17tGG0hM+1/NL9T01mCAJRzNa1f6PnBxHTTVH1AsY/w71nGwD1YDMCGKLWh2AR28b5Rxjmj
egKBGCBQsJVWDDyqEzPqKwdaHhA5hdaIO+R65gw35KA5Da1bEyitH1VByvwVLSGfH3GZSTHU6LumR51UQ19XIskFR17vMk067SK
q+VoY+e36eSi0gaBANIW+jUBbbuLV1Fwz740JVm/SPSU1BnTQ0IZPzobu7jrKY9+LG+o8FEoh4dMSo9b dxabezas@m2-dxabeza
S

```

Figura 7: La clave pública de M2 guardada en M1. Se puede ver que es de M2 en el final de la clave, porque lo que indicamos con la opción `-C` al crear la clave.

Borramos la clave con nano, y comprobamos que efectivamente nos pide la contraseña para acceder.

Ahora escribimos de forma manual la clave que generamos en la primera práctica de M2 en el fichero `authorized_keys` de M1.

```

dxabezas@m2-dxabezas:~/.ssh$ cat id_rsa.pub | ssh dxabezas@192.168.56.101 "cat >> ~/.ssh/authorized_
keys"
dxabezas@192.168.56.101's password:
dxabezas@m2-dxabezas:~/.ssh$ ssh dxabezas@192.168.56.101
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-136-generic x86_64)

```

Figura 8: Copiamos manualmente la clave pública de M2 en M1, y ya podemos acceder sin contraseña.

5. Programación de tareas: crontab

En la máquina 2, utilizamos el demonio cron para que sincronice su directorio `/var/www` con el de M1 cada hora, concretamente en el minuto 0. Esto lo conseguimos haciendo `crontab -e` en la máquina 2 y añadiendo la siguiente línea:

```

# m h dom mon dow command
0 * * * * rsync -avz -e ssh dxabezas@192.168.56.101:/var/www/ /var/www/ --delete

```

Como se nos indica el fichero, no hace falta escribir el usuario porque ya sabe cuál es. Esperamos un poco y comprobamos que se sincroniza correctamente.

Hemos usado `--delete` para que el contenido de las carpetas sea idéntico.

Ahora señalaremos algunas opciones avanzadas para crontab. En primer lugar, el carácter `-` puede usarse para indicar intervalos, y el carácter `/` para indicar saltos. Por ejemplo, si ponemos `0 12 1-15/3 * *`, la tarea se ejecutará los días 1,4,7,10,12 y 15 de cada mes a las 12 del mediodía.

También tenemos atajos, que se marcan con el carácter `@`. Por ejemplo, `@weekly` equivale a `0 0 * * 0`, que ejecuta la tarea cada domingo (día 0) a las 00:00. De igual forma, existen `@yearly` y `@annually` para ejecutar la tarea el 1 de enero a las 00:00; y `@hourly`, `@daily` y `@monthly`.

Finalmente tenemos `@reboot`, que lanza la tarea tras cada vez que apagamos y encendemos la máquina.