

Práctica 2:

Clonar la información de un sitio web

Índice:

- 1) Copiar archivos locales en un equipo remoto*
- 2) Clonar contenido entre máquinas con rsync*
- 3) Configurar el servicio SSH para acceder a máquinas remotas sin contraseña*
- 4) Programar tareas con Crontab*

1) Copiar archivos locales en un equipo remoto.

Vamos a crear un fichero tar con el contenido de “/var/www/html” de la máquina m2 al home de la máquina m1 ejecutando la siguiente orden en la máquina m2:

```
tar cfz - /var/www/html | ssh joselepedraza@192.168.56.101 'cat > ~/tar.tgz'
```

En este caso estamos creando el tar.tgz de un equipo para dejarlo en el equipo destino, creándolo directamente en este equipo destino con SSH (esto es muy útil cuando no dispongamos de espacio en el disco local). Para ello, indicamos al comando tar anterior que queremos que use stdout como destino y mandamos con una pipe la salida al SSH. Esto coge la salida del tar y la escribe en un fichero. De esta forma en el equipo destino (máquina m1) tendremos creado el archivo tar.tgz.

```
joselepedraza@m2:~$ tar cfz - /var/www/html | ssh joselepedraza@192.168.56.101 'cat > ~/tar.tgz'
tar: Removing leading `/' from member names
joselepedraza@192.168.56.101's password:
joselepedraza@m2:~$ _
```

Descomprimos el archivo tar.tgz en la maquina m1 y comprobamos que se ha copiado el contenido correctamente con la siguiente opción:

```
tar --extract -f tar.tgz
```

```
joselepedraza@m1:~$ ls
joselepedraza@m1:~$ ls -a
.  .bash_history  .bashrc  .gnupg  .ssh  .vim
.. .bash_logout  .cache   .profile .sudo_as_admin_successful .viminfo
joselepedraza@m1:~$ ls
tar.tgz
joselepedraza@m1:~$ tar --extract -f tar.tgz
joselepedraza@m1:~$ ls
tar.tgz  var
joselepedraza@m1:~$ ls var/www/html/
ejemploM2.html  index.html
joselepedraza@m1:~$ _
```

Esto también sería posible utilizando SCP que hace uso de SSH para hacer copias seguras y encriptadas de archivos o directorios como sigue:

```
tar -czvf directorio archivo.tgz
scp archivo.tgz usuario@equiporemoto:~/archivo.tgz
```

o directamente con:

```
scp -r directorio usuario@equiporemoto:/directorio
```

2) Clonar contenido entre máquinas con rsync.

Sin embargo, el método anterior, que puede ser útil en algún momento dado, no nos servirá para sincronizar grandes cantidades de información, para lo cual utilizaremos la herramienta rsync.

Esta herramienta se encuentra disponible junto con su documentación en:

<http://rsync.samba.org/>

En nuestras máquinas Ubuntu m1 y m2 podemos instalarlo usando apt-get:

```
sudo apt-get install rsync
```

Para proceder a clonar un directorio (en nuestro caso /var/www/) de la maquina m1 en la máquina m2, lo primero que debemos hacer es que el usuario sea dueño del directorio con el siguiente comando:

```
chown usuario:usuario -R /var/www
```

En principio este comando lo podremos ejecutar sin privilegios, pero para la posterior programación de tareas con cron es obligatorio ejecutarla con privilegios root por lo que deberemos anteponer sudo al comando o trabajar como super usuario con sudo su.

```
sudo su
```

```
chown joselepedraza:joselepedraza -R /var/www
```

```
joselepedraza@m1:~$ sudo su
[sudo] password for joselepedraza:
root@m1:/home/joselepedraza# chown joselepedraza:joselepedraza -R /var/www
root@m1:/home/joselepedraza# _

joselepedraza@m2:~$ sudo su
[sudo] password for joselepedraza:
root@m2:/home/joselepedraza# chown joselepedraza:joselepedraza -R /var/www
root@m2:/home/joselepedraza#
```

Una vez hecho esto, el usuario tiene los permisos sobre la carpeta a clonar. Pasamos a ejecutar la herramienta rsync desde la máquina m2 especificando la dirección ip de destino, la ip de la máquina m1 en nuestro caso:

```
rsync -avz -e ssh ipm1:/var/www/ /var/www/
```

La opción *--delete* → Para un clonado perfecto, indica que aquellos ficheros de que se hayan eliminado en la máquina origen, también se borren de la máquina destino.

La opción *--exclude* → Indica que ciertos directorios o ficheros no deben copiarse (por ejemplo, archivos de log o error).


```
rsync -avz --delete --exclude=**/stats --exclude=**/error --  
exclude=**/files/pictures -e ssh maquina1:/var/www/ /var/www/
```

Con la orden anterior estaremos haciendo la copia completa del directorio /var/www pero excluyendo /var/www/error, /var/www/stats y /var/www/files/pictures.

En nuestro caso concreto:

```
rsync -avz -e ssh 192.168.56.101:/var/www/ /var/www/
```

```
joselepedraza@m2:/$ cat /var/www/html/ejemplo.html  
<html>  
  <body>  
    Web de ejemplo de joselepedraza para SWAP(m2)  
  </body>  
</html>  
joselepedraza@m2:/$ rsync -avz -e ssh 192.168.56.101:/var/www/ /var/www/  
joselepedraza@192.168.56.101's password:  
receiving incremental file list  
./  
html/  
html/ejemplo.html  
html/index.html  
  
sent 175 bytes  received 315 bytes  108.89 bytes/sec  
total size is 10,997  speedup is 22.44  
joselepedraza@m2:/$ cat /var/www/html/ejemplo.html  
<html>  
  <body>  
    Web de ejemplo de joselepdraza para SWAP(m1)  
  </body>  
</html>  
joselepedraza@m2:/$ _
```

 m1 [Corriendo] - Oracle VM VirtualBox

Archivo Máquina Ver Entrada Dispositivos Ayuda

```
joselepedraza@m1:/$ cat /var/www/html/ejemplo.html  
<html>  
  <body>  
    Web de ejemplo de joselepdraza para SWAP(m1)  
  </body>  
</html>  
joselepedraza@m1:/$ _
```

Para comprobar el correcto funcionamiento, primeramente, muestro el contenido de ejemplo.html en la máquina m1 y en la máquina m2. Después en la máquina m2 ejecuto el rsync de la carpeta /var/www/ (especificando la dirección ip de la máquina m1). Finalmente compruebo que ha cambiado el archivo ejemplo.html de la máquina m2 y ahora muestra el de la máquina m1, lo que quiere decir, que está correctamente sincronizado el directorio.

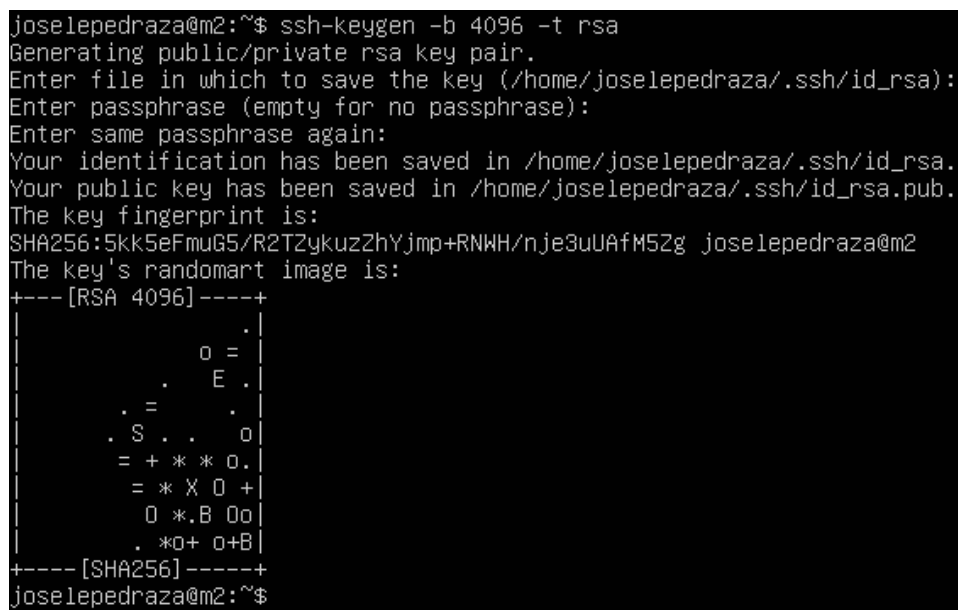
3) Configurar el servicio SSH para acceder a máquinas remotas sin contraseña.

El uso de rsync nos será de gran ayuda para mantener el contenido de varias máquinas actualizado e idéntico. Sin embargo, esa actualización debería hacerse automáticamente, a través de scripts que no requieran la intervención del administrador que vaya tecleando contraseñas. Al realizar scripts que se conectan mediante SSH a equipos remotos para ejecutar alguna acción, como copias de seguridad o clonado, nos podemos encontrar que se queden esperando la contraseña.

Para evitarlo, normalmente se usa autenticación con un par de claves pública-privada (para usar SSH sin tener que introducir contraseña al conectarnos a otra máquina).

Con la herramienta ssh-keygen podemos generar la clave necesaria con el siguiente comando:

```
ssh-keygen -b 4096 -t rsa
```

A terminal window showing the execution of the ssh-keygen command. The prompt is 'joselepedraza@m2:~\$'. The command 'ssh-keygen -b 4096 -t rsa' is entered. The output shows the generation of a public/private RSA key pair. It prompts for a file name (defaulting to /home/joselepedraza/.ssh/id_rsa), a passphrase (empty), and confirmation of the passphrase. It then shows the key's fingerprint (SHA256:5kk5eFmuG5/R2TZykuz2hYjmp+RNWH/nje3uUAFM52g) and a randomart image. The randomart image is a square grid of characters representing the key's fingerprint. The command prompt returns to 'joselepedraza@m2:~\$'.

```
joselepedraza@m2:~$ ssh-keygen -b 4096 -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/joselepedraza/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/joselepedraza/.ssh/id_rsa.
Your public key has been saved in /home/joselepedraza/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:5kk5eFmuG5/R2TZykuz2hYjmp+RNWH/nje3uUAFM52g joselepedraza@m2
The key's randomart image is:
+---[RSA 4096]---+
|                 .|
|                o =|
|               . E .|
|              . =  .|
|             . S . . o|
|            = + * * o.|
|           = * X o +|
|          o *.B oo|
|         . *o+ o+B|
+-----[SHA256]-----+
joselepedraza@m2:~$
```

Para indicar el tamaño de la clave usamos la opción -b y para el tipo de clave -t:

- rsa1 → Genera fichero ~/.ssh/identity para clave privada y ~/.ssh/identity.pub para clave pública. (formato valido protocolo1 SSH).
- rsa → Genera fichero ~/.ssh/id_rsa para clave privada y ~/.ssh/id_rsa.pub para clave pública. (formato valido protocolo2 SSH). Opción usada.
- dsa → Genera fichero ~/.ssh/id_dsa para clave privada y ~/.ssh/id_dsa.pub para clave pública. (formato valido protocolo2 SSH).

Ahora debemos copiar la clave desde la máquina m2 a la máquina m1 (a la máquina principal) añadiéndola al fichero `~/.ssh/authorized_keys` con el comando:

```
ssh-copy-id 192.168.56.101
```

Se nos preguntará acerca de la ruta donde deseamos copiar la clave; luego la passphrase, que dejaremos vacía ya que no le queremos añadir mayor seguridad a la clave privada en este caso ya que lo que queremos es poder conectar equipos sin contraseña (lo dejamos doblemente en blanco).

Finalmente, si se nos requerirá la contraseña de m1 por última vez antes de copiar la clave pública-privada.

```
joselepedraza@m2:~$ ssh-copy-id 192.168.56.101
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/joselepedraza/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install all the new keys
joselepedraza@192.168.56.101's password:

Number of key(s) added: 1

Now try logging into the machine, with:  "ssh '192.168.56.101'"
and check to make sure that only the key(s) you wanted were added.

joselepedraza@m2:~$ ssh 192.168.56.101
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-91-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

System information as of Fri Mar 27 18:13:15 UTC 2020

System load:  0.04               Processes:            93
Usage of /:   48.7% of 9.78GB    Users logged in:     1
Memory usage: 33%               IP address for enp0s3: 10.0.2.15
Swap usage:   0%                IP address for enp0s8: 192.168.56.101

 * Latest Kubernetes 1.18 beta is now available for your laptop, NUC, cloud
   instance or Raspberry Pi, with automatic updates to the final GA release.

   sudo snap install microk8s --channel=1.18/beta --classic

 * Multipass 1.1 adds proxy support for developers behind enterprise
   firewalls. Rapid prototyping for cloud operations just got easier.

   https://multipass.run/

Pueden actualizarse 15 paquetes.
0 actualizaciones son de seguridad.

Last login: Fri Mar 27 17:59:24 2020
joselepedraza@m1:~$
```

Comprobamos que podemos establecer la conexión SSH de m2 a m1 sin necesidad de introducir la contraseña. Para ejecutar comandos en remoto simplemente añadirlo al final de la orden ssh.

Si este no fuera el caso, probamos a darle permisos al fichero `~/.ssh/authorized_keys` y repetimos el proceso anterior:

```
chmod 600 ~/.ssh/authorized_keys
```

4) Programar tareas con Crontab

Teniendo preparado el acceso sin contraseña por ssh, podemos hacer uso de rsync desde scripts que se ejecuten automáticamente con la herramienta cron (por ejemplo, cada noche a cierta hora).

Antes de nada, rescatamos el archivo `/var/www/html/ejemplo.html` de la máquina m2 a su original (para las posteriores comprobaciones de cron ya que estaba ya sincronizado con el de la máquina principal).

Cron se ejecuta en background y revisa cada minuto la tabla del fichero `/etc/crontab` en búsqueda de las tareas programadas.

Los 7 campos que conforman las líneas del archivo crontab están organizados de la siguiente manera:

Minuto	Hora	DiadelMes	Mes	DiadelaSeman	Usuario	Comando
0-59	0-23	1-31	1-12	1-7	joselepedraza	rsync...

Se puede usar la orden `crontab -e` para editar una tarea crontab en lugar de editar el archivo `/etc/crontab`.

En la máquina m2 editamos el archivo `/etc/crontab` con vim (como root) y añadimos la siguiente tarea al mismo, que realizará el rsync de la carpeta `/var/www/` cada hora (para tener la réplica de m1 en m2 actualizada cada hora):

```
0 * * * * joselepedraza rsync -avz -e ssh 192.168.56.101:/var/www/ /var/www/
```

```
# Unlike any other crontab you don't have to run the `crontab`
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 * * * * root    cd / && run-parts --report /etc/cron.hourly
25 6 * * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6 * * 7 root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6 1 * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
)
0 * * * * joselepedraza rsync -avz -e ssh 192.168.56.101:/var/www/ /var/www/
#
~
```

Vemos que el fichero `ejemplo.html` en m2 aun no se ha actualizado, esperaremos una hora para comprobar que funciona correctamente.

```
joselepedraza@m2:~$ cat /var/www/html/ejemplo.html
<html>

    <body>

        Web de ejemplo de joselepedraza para SWAP(m2)

    </body>

</html>
joselepedraza@m2:~$ _
```

El contenido que deberá tener el fichero ejemplo.html transcurrida una hora es el de la máquina m1:

```
joselepedraza@m1:~$ cat /var/www/html/ejemplo.html
<html>
    <body>
        Web de ejemplo de joselepedraza para SWAP(m1)
    </body>
</html>
joselepedraza@m1:~$
```

Como vemos funciona correctamente y el contenido del directorio /var/www/ de la máquina m1 se replicará cada hora en el directorio de mismo nombre de la máquina m2:

```
joselepedraza@m2:~$ cat /var/www/html/ejemplo.html
<html>
    <body>
        Web de ejemplo de joselepedraza para SWAP(m1)
    </body>
</html>
joselepedraza@m2:~$ _
```

Recordar que cualquier cosa que hagamos en el directorio /var/www/ de la máquina m2 será machacado por el de la máquina m1 cada hora.