1 Usos de SSH

1.1 Sumario

2 Instalación de SSH

2.1 Instalación de openssh-server

openssh-server es un paquete disponible para GNU/Linux. Por defecto no viene instalado en las distribuciones, sin embargo muchos instaladores, como en el caso de Debian, que usa tasksel durante la instalación, permiten su despliegue durante el propio proceso de instalación.

Si no está disponible en nuestro sistema podemos instalarlo, en este caso en Debian 9 (stretch), con:

```
apt install -y openssh-server
```

Una vez instalado el servidor podremos comprobar si está funcionando correctamente ejecutando

```
systemctl status ssh
```

NOTA: El comando anterior sería equivalente a

```
service ssh status
```

en cualquiera de los dos casos el resultado de la salida debería contener el texto:

```
Active: active (running)
```

Otro modo de comprobar la operatividad del servidor ssh es comprobando si en el puerto 22, el puerto que se usa por defecto para el servicio, hay un servicio escuchando. Esto lo podemos comprobar con

```
lsof -i :22
```

Debería mostrar algo del tipo

```
COMMAND PID USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
sshd    344 root    3u  IPv4  11210      0t0  TCP *:ssh (LISTEN)
sshd    344 root    4u  IPv6  11212      0t0  TCP *:ssh (LISTEN)
```

Si queremos editar algún parámetro de configuración del servidor ssh, deberíamos hacerlo en el archivo **/etc/ssh/sshd_config**

algunas directivas a modificar podrían ser:

- Puerto en el que escucha el servidor
  - Activamos en el archivo la directiva (descomentando la línea correspondiente), para que el servidor escuche en el puerto 8022

```
Port 8022
```

- Direcciones IP en las que se mantiene a la escucha el servidor
- Parámetros de autenticación, como por ejemplo si se permite acceder con el usuario root
    - ♦ Activamos en el archivo la directiva (descomentando la línea correspondiente), para permitir el acceso con el usuario root

```
PermitRootLogin yes
```

- Cierre automático de conexiones abandonadas
- Reenvío de X
- etc

tras los cambios reiniciamos el servicio ssh

```
systemctl restart ssh
```

3 Acceso remoto con ssh

Para acceder remotamente a un servidor ssh utilizamos el comando ssh. La sintaxis es la siguiente

ssh usuario@ip_o_hostname_remoto -p puerto

La opción -p permite especificar el puerto de escucha del servidor ssh. Si no se especifica se presupone el puerto 22.

Otras opciones del comando pueden verse ejecutando

```
ssh --help
```

Veamos un ejemplo

```
ssh root@10.200.10.9 -p 8022
```

el comando anterior representa un acceso ssh con las siguientes características:

- accedemos con el usuario **root**
- a la máquina con dirección IP **10.200.10.9**
- el **puerto** de escucha del servidor ssh en esta máquina es el **8022**

tras la ejecución del comando el cliente ssh, es decir el comando ssh que utilizamos para acceder al servidor ssh, muestra:

```
The authenticity of host '[10.200.10.9]:8022 ([10.200.10.9]:8022)' can't be established.
ECDSA key fingerprint is SHA256:7VUBc58vgoKFa46Qsi33B7JgWOkm8zQWldAhBXbFUz4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[10.200.10.9]:8022' (ECDSA) to the list of known hosts.
root@10.200.10.9's password:
```

el mensaje anterior hace referencia a la huella dactilar o ?**fingerprint**? del servidor ssh. Cada servidor ssh dispone de su propia fingerprint, de este modo se comprueba su identidad. El mensaje anterior aparecerá la primera vez que accedemos al servidor, para advertirnos de que estamos accediendo a un servidor desconocido por nuestro cliente. Si es, efectivamente, la primera vez que accedemos al mismo, deberemos aceptar su identidad representada por el fingerprint; para ello respondemos yes a la pregunta que nos plantea. Una vez aceptada la fingerprint nos pide la password del usuario con el que tratamos de acceder remotamente.

Las fingerprint de los servidores ssh a los que accedemos se almacenan en el archivo ~/**.ssh**/**known_hosts**.

Si tratando de acceder a un servidor ?conocido?, es decir uno del que ya conozcamos su identidad a través del fingerprint, recibimos un mensaje similar al anterior, deberemos de sospechar que puede tratarse de una suplantación, ya que, si fuera el servidor conocido, ya dispondría de su fingerprint y, por tanto, no sería necesaria su aceptación. Por tanto hay que estar atento a esta circunstancia al utilizar ssh.

3.1 Acceso remoto con el usuario root

Por defecto el servidor openssh-server no permite el acceso remoto con usuario root con contraseña. Esto es una medida de seguridad que trata de evitar ataques de repetición bajo la suposición de que todos los sistemas GNU/Linux dispone de un usuario root habilitado. Un atacante podría intentar

un acceso con root a través de un robot que utilice un diccionario, para generar contraseñas ?probables?, y que envíe periódicamente un nuevo intento.

Por este motivo no se permite el acceso con root. Para permitirlo debemos editar el archivo /**etc/ssh/sshd_config**

Concretamente establecemos la directiva **PermitRootLogin**

- **PermitRootLogin yes**: permite acceso con root mediante cualquier sistema de autenticación permitido, incluida contraseña
- **PermitRootLogin without-password**: permite acceso con root pero sin contraseña, por lo general haciendo uso de un certificado de clave pública del usuario

3.1.1 Colores al acceder con cliente ssh

Es probable que al acceder con ssh con el usuario root, los listados del comando ls no se vean coloreados. Para habilitar esta posibilidad deberemos editar el archivo /root/.bashrc del sistema remoto, descomentando las líneas a continuación de la línea de comentario

```
You may uncomment the following lines if you want `ls' to be colorized:
```

3.1.2 Envío de comandos remotos por ssh

Otro de los usos habituales de ssh es enviar comandos a un sistema remoto sin necesidad de iniciar una sesión interactiva.

Mediante la sintaxis:

ssh user@hostremoto 'comando'

Se ejecuta comando en el sistema hostremoto, en nombre del usuario user

Ejemplo

```
ssh jefe@miserver.com 'ls -la /'
```

Pueden concatenarse comandos intercalando el carácter **;** entre una lista de comandos pasados en el último parámetro

Si el comando es interactivo y necesita intervención por parte del usuario usamos la opción -t

ssh -t user@hostremoto 'comando'

Ejemplo

```
ssh -t jefe@miserver.com 'top'
```

4 Configuración de autenticación automática

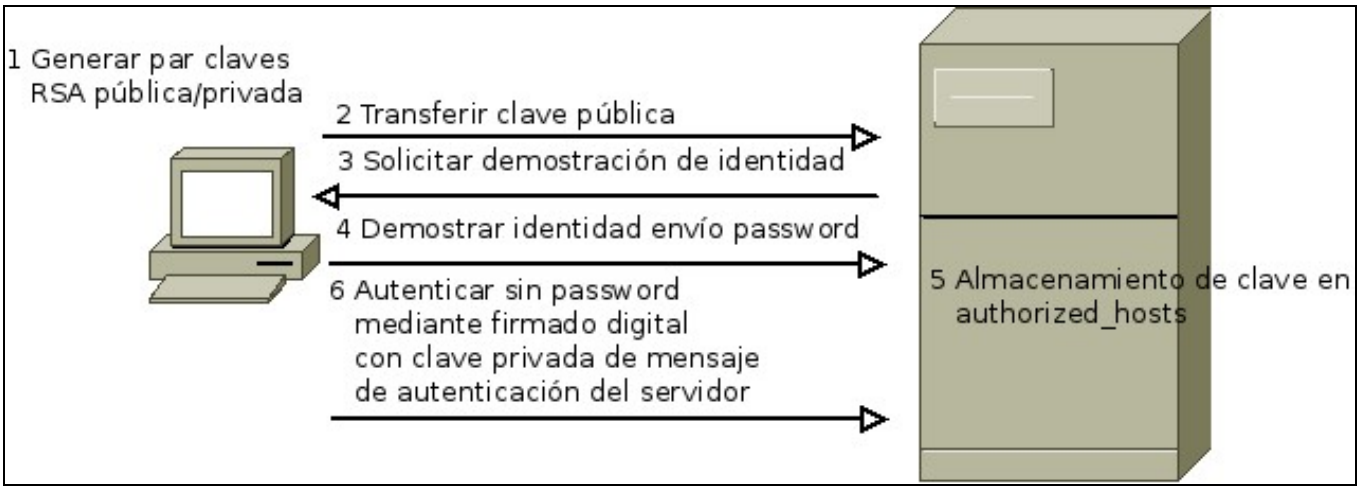
Es bastante común el tener que utilizar el protocolo ssh desde scripts o tareas automáticas, en las cuales el usuario no actúa directamente sobre el sistema de destino.

Por ejemplo, imaginemos una tarea automática de copia de seguridad que se ejecuta diariamente y que culmina con el envío de un archivo con la copia a un servidor mediante ssh. En el punto en el que se realiza el acceso remoto no hay ningún usuario disponible para introducir la credencial de acceso, por ejemplo la contraseña, al mismo.

Para estos casos el protocolo ssh permite autenticación basada en criptografía asimétrica o de clave pública. Esta técnica hace uso de un par de certificados público/privado, que permiten

- **Confidencialidad**: Cifrar toda la comunicación entre cliente y servidor
- **Autenticación**: Acreditar la identidad del cliente al servidor

En la siguiente imagen veremos el proceso necesario para desplegar la autenticación automática mediante par de claves



Veamos el procedimiento para ponerlo en marcha

4.1 Creación de par de claves RSA

Para poder utilizar autenticación automática ssh necesitamos generar un par de claves ssh en el cliente. Para ello ejecutamos

```
ssh-keygen
```

aceptando las opciones por defecto generará un par de claves RSA pública/privada

- **~/.ssh/id_rsa.pub**: Clave pública, será la utilizada para la autenticación remota, que deberá ser transferida al servidor para acreditar la identidad del cliente, y por tanto autenticarlo.
- **~/.ssh/id_rsa**: Clave privada, será utilizada para cifrar un mensaje de autenticación que el servidor descifrará con la clave pública, acreditando por tanto la identidad del cliente, ya que éste será el único usuario con acceso a la clave privada. Es muy importante que esta clave no salga de la máquina del cliente, y que ningún otro usuario tenga acceso a ella.

4.2 Transferencia de la clave pública al servidor

El siguiente paso en el procedimiento es transferir la clave pública al servidor ssh. De este modo, cuando se envíe el mensaje de autenticación desde el cliente, cifrado con su clave privada, podrá acreditar su autenticidad, utilizando para descifrarlo la clave pública, ya que el cliente es el único poseedor legítimo de la clave privada.

Para transferir la clave pública desde el cliente al servidor ssh ejecutamos

```
ssh-copy-id -i ~/.ssh/id_rsa.pub root@10.200.10.9 -p 8022
```

Durante la transferencia de la clave pública, llevada a cabo por el comando anterior, y solamente en esa ocasión, deberemos de introducir la contraseña de acceso al servidor para el usuario root.

Una vez completado el comando, podremos comunicarnos remotamente mediante ssh con el servidor desde el cliente, con el usuario de sesión del cliente, para iniciar sesión con el usuario root en el servidor.

Si necesitamos acceder con otro usuario al servidor, o bien utilizando otro usuario en el cliente para acceder, debemos repetir los pasos anteriores para los usuarios en cuestión.

En el servidor las claves públicas de acceso ssh se almacenan en el archivo **~/.ssh/authorized_keys**

Ahora podremos acceder, sin necesidad de especificar la contraseña, al sistema remoto:

```
ssh root@10.200.10.9 -p 8022
```

5 Copia remota con scp

Otra utilidad que viene disponible en cliente del protocolo ssh disponible en las distribuciones GNU/Linux es el comando scp. Con el comando scp podremos realizar copias de archivos remotas desde y hacia servidores ssh.

Veamos su uso mediante un ejemplo

```
scp ~/.profile root@10.200.10.9:/tmp -P 8022
```

El comando anterior copiaría el archivo .profile del directorio home del usuario (abreviado con el carácter ~) al directorio /tmp en el servidor 10.200.10.9, y ejecuta esa transferencia haciendo uso de la cuenta del usuario root en el servidor.

Si queremos copiar un directorio deberemos usar scp con la opción -R, recursiva, la cual permite copiar directorios y todos sus contenidos

```
scp -R /tmp root@10.200.10.9: -P 8022
```

el comando anterior copiaría el directorio /tmp del cliente en el directorio home del usuario root en el servidor con dirección IP 10.200.10.9. Notar que el puerto del servidor en scp se indica con el parámetro **-P** (mayúscula)

IMPORTANTE: Al final de la ruta remota del comando debe especificarse la ruta de copia en el sistema remoto, si no se indica hay que dejar al menos el carácter ?:?, el cual hace referencia al directorio home del usuario remoto utilizado.

NOTA: También puede utilizarse scp a la inversa, es decir, copiando desde el sistema remoto al sistema local, para ello basta con invertir los parámetros de la copia

```
scp -R root@10.200.10.9:/tmp . -P 8022
```

copiaría el directorio /tmp remoto al directorio actual en el sistema local

6 Túneles ssh

Otro uso interesante de ssh se basa en la posibilidad de utilizar una conexión SSH con un servidor para "encapsular" comunicaciones de otros servicios. Por ejemplo, podemos acceder a un servidor MySQL en el servidor mediante un túnel SSH abierto en el servidor, sin tener que abrir explícitamente el puerto del MySQL.

Este mecanismo ofrece varias ventajas:

- No es necesario abrir puertos conocidos de servicios
- La información viaja cifrada en un túnel de seguridad

6.1 Comando para abrir un túnel SSH en localhost con forwarding de puertos remoto

Ejecutando el comando

```
ssh -f usuario@ssh.aloxa.proba -L 10000:192.168.56.14:80 -N
```

El servidor SSH, **ssh.aloxa.proba** en este caso, escucharía en el puerto 22 (por defecto). En caso de usar un puerto diferente, lo indicarámos con la opción **-p**. La opción **-f** ejecuta el comando en background, de modo que la consola no quede asociada al comando.

La opción **-L** indica los parámetros del túnel. En este caso estamos redirigiendo el puerto 10000, en el sistema local, al puerto 80 de la máquina remota con IP 192.168.56.14, pero lo haremos a través del puerto 22, asociado a ssh. La opción **-N** indica que se no ejecute ningún comando en la máquina remota.

El túnel sería accedido desde la máquina local (localhost) a través del puerto 10000. Es decir, podríamos acceder al servidor web (puerto 80) del servidor remoto 192.168.56.14, a través de localhost:10000, mediante el túnel SSH establecido a través de la máquina ssh.aloxa.proba.

Para que cualquier máquina pueda acceder al túnel, desde la misma red que la máquina en la que empieza el mismo, usaríamos la opción **-g**

```
ssh -f usuario@ssh.aloxa.proba -L 10000:192.168.56.14:80 -g -N
```

También pueden establecerse túneles inversos, en este caso permitiendo el acceso desde máquinas remotas a la máquina en la que se define el túnel

```
ssh usuario@ssh.aloxa.proba -R 10000:10.0.0.1:22
```

7 Acceso a sistemas de archivos remotos con sshfs

Otro uso muy simple y útil es la posibilidad de utilizar ssh para montar sistemas de archivos remotos en rutas locales. Para conseguir este objetivo tenemos que instalar el paquete sshfs

```
apt install sshfs
```

A continuación vamos a ver mediante un ejemplo el uso de esta utilidad

```
mkdir /tmp/remoto
sshfs root@10.200.10.9:/root /tmp/remoto -p 8022
```

El primer comando crea el punto de montaje dentro del directorio /tmp.

A continuación ejecuta el montaje del directorio /root del servidor 10.200.10.9 en el punto de montaje que creamos anteriormente.

Veamos el resultado del comando

```
mount -l | grep remoto
```

muestra:

```
root@10.200.10.9:/root on /tmp/remoto type fuse.sshfs (rw,nosuid,nodev,relatime,user_id=1758992858,group_id=1758986753)
```

así de fácil...

Podríamos también definir puntos de montaje en el inicio del sistema definidos en el archivo **/etc/fstab.**

Para el ejemplo anterior añadiríamos una línea como la siguiente:

```
root@10.200.10.9:/root  /tmp/remoto  fuse.sshfs \
noauto,x-systemd.automount,_netdev,users,idmap=user,IdentityFile=/root/.ssh/id_rsa,\
allow_other,reconnect,port=8022 0 0
```

Breve explicación:

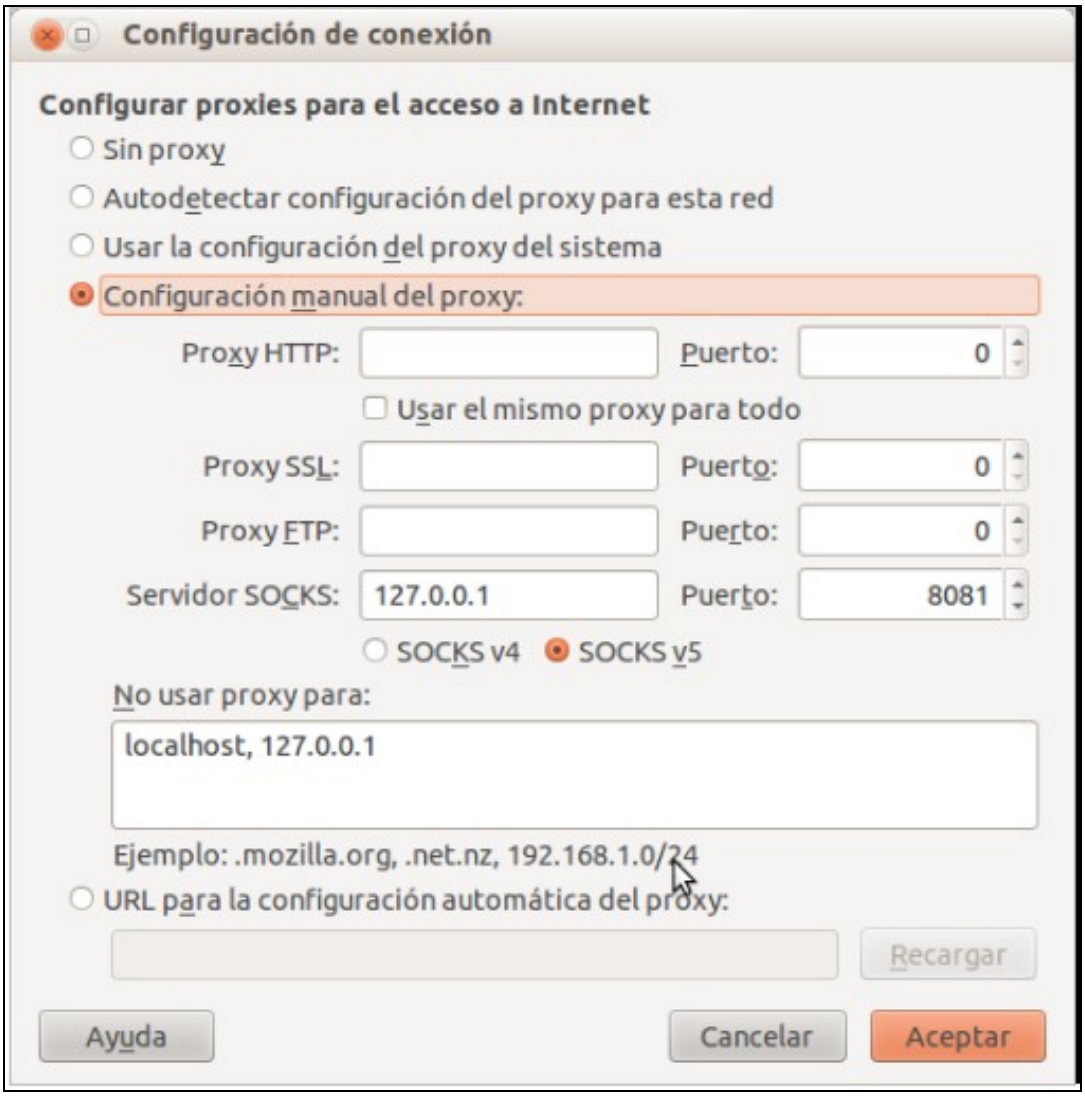
- Accedemos al directorio /root del servidor 10.200.10.9 y lo montará en /tmp/remoto en el sistema local
- El sistema de archivos para el montaje es fuse.sshfs
- A continuación vienen las opciones de montaje típicas de sshfs. Notar el uso del parámetro IdentityFile dentro de las opciones del punto de montaje, el cual identifica en este caso el archivo de clave privada del usuario que accede al sistema remoto.

8 Navegación a través de proxy SOCKS con ssh

Con el siguiente comando

```
ssh -f -D 8081 -p 22 root@hostremoto -N
```

Establecemos un túnel de tipo SOCKS proxy que comienza en el puerto local 8081 y que reenvía todo el tráfico al puerto 22 del host remoto, permitiendo navegar a través del sistema remoto. Este tipo de túnel se utiliza ampliamente para navegar a través de un equipo remoto, evitando posibles restricciones de proxys y cortafuegos en la red local. Para que funcione es necesario establecer en navegador web la opción de salida por un proxy de tipo SOCKS, en este caso localhost, a través del puerto 8081 en este caso. Por ejemplo, en Firefox:



En general esta idea puede extendese a cualquier servicio para evitar las restricciones salientes de los firewalls. Por ejemplo, el comando:

```
ssh -f -L 3000:talk.google.com:5222 root@server.home -N
```

provocaría que todo el tráfico hacia el servicio google talk pase desde el puerto 3000 local (para ello hay que reconfigurar el cliente de mensajería para que utilizar como server localhost y como puerto el 3000), al otro extremo del túnel, el host **server.home**. Es decir, todo el tráfico de salida hacia ese servicio se manda, vía túnel, a través de un servidor intermedio (server.home)

Volver

JavierFP 18:57 20 nov 2017 (CET)