Instalación y configuración del servicio SSH

José Luis Palencia Segura — ASTRA

25 de octubre de 2024

debian

Índice

1. Instalación

Si bien el paquete que incluye las funcionalidades tanto de servidor como cliente, se incluye en la paquetería por defecto de la mayoría de distribuciones. El paquete normalmente utilizado es "ssh", que instalaríamos con la linea:

```
sudo apt install ssh
```

No obstante este es un metapaquete, que contiene los paquetes **openssh-server**, **openssh-client** y **openssh-sftp-server**, que pueden ser instalados de manera independiente según nuestras necesidades.

1.1. Paquetes de instalación

Cada uno de estos paquetes tiene utilidades diferentes relacionadas con el servicio:

1.1.1. openssh-server

Este es el paquete que nos permite habilitar el servicio de ssh y recibir conexiones de otros clientes. Permite configuraciones generales, y específicas para cada usuario, y opera por defecto sobre el puerto 22.

1.1.2. openssh-client

Este paquete nos permitirá realizar una conexión a un equipo que tenga activo el servicio sshd, autenticandonos como un usuario del servidor, que permita esta conexion ya sea aportando sus credenciales, o haciendo uso de una clave *(fingerprint)*.

1.1.3. openssh-sftp-server

Este paquete es un añadido al servicio de sshd, que permite enviar archivos entre equipos *(ftp)*, aprovechando la conexión cifrada de ssh.

2. Configuración del servicio

La configuración de ssh solo es necesaria en el equipo servidor, y se encuentra en los archivos y directorios (**/etc/ssh/sshd_config**), (**/etc/ssh/sshd_config.d/**) y (**/home/user/.ssh/**).

2.1. Configuración de ssh

2.1.1. sshd_config

Este es el archivo de configuración general, que aplica por defecto a todos los usuarios cuando no hay configuraciones específicas. Los parámetros mas comunmente modificados son los siguientes:

Parámetro	Valor	Descripción
Port	22	Puerto empleado por defecto en conexiones ssh
ListenAddress	0.0.0.0	Dirección de origen permitida (no admite rangos)
PermitRootLogin	yes/no	Importante no permitir acceso a root desde ssh
PubkeyAuthentication	yes/no	Autenticación con par de claves
AllowUsers	usuario	Permitir ssh a los usuarios indicados
MaxAuthTries	3	Numero de intentos fallidos antes de desconectar
AllowTcpForwarding	yes/no	Permitir reenvío de puertos
X11Forwarding	yes/no	Mostrar aplicaciones graficas remotas localmente

2.1.2. sshd_config.d/*

Este directorio permite organizar las diferentes configuraciones, dandonos la posibilidad de separar en varios archivos los parámetros anteriores y muchos otros de configuración específica, como la que veremos a continuación para configurar sftp.

2.1.3. .ssh/*

Este directorio se encuentra en la raiz del directorio personal de cada usuario, y permite configuraciones específicas creando el archivo opcional 'config', además de los archivos:

- known_hosts : contiene la huella digital de equipos que han realizado una conexion.
- authorized_keys : lista de claves públicas que pueden autenticarse en el servidor.

2.2. Configuración de sftp

Este servicio cuelga del mismo que ssh, y se configura en el mismo archivo de configuración. Por defecto el directorio raiz será el del usuario con el que nos autenticamos, pero es posible realizar una configuración específica para cambiar dicha raiz.

Para habilitar sftp hay que asegurarse de que la siguiente linea se encuentre descomentada en el archivo sshd_config :

```
Subsystem sftp /usr/lib/openssh/sftp-server
#Tambien puede encontrarse de la siguiente manera, ambas son validas:
Subsystem sftp internal-sftp
```

La configuración específica para usuarios o grupos, tambien debe incluirse en uno de los archivos de configuración mencionados anteriormente, y se realiza de la siguiente manera:

```
Match [Group/User] [Nombre de Usuario/Grupo]
  ChrootDirectory %h #Directorio raiz
  ForceCommand internal-sftp #Fuerza sftp, sin acceso a comandos de shell
  AllowTcpForwarding no #Deshabilita el reenvio de puertos
  X11Forwarding no #Deshabilita el reenvio de graficos X11.
```

Es importante que los directorios a los que el usuario debe acceder, tengan los permisos necesarios, de lo contrario no podremos realizar el envío/descarga de archivos.

3. Puesta en marcha

Una vez modificada la configuración para adaptarla a nuestra situación de uso, levantaremos/reiniciaremos el servicio y lo habilitaremos para inciarse al mismo tiempo que el sistema:

```
sudo systemctl enable sshd.service
sudo systemctl restart sshd.service
```

Por defecto el servcio se encuentra iniciado, pero debemos reiniciarlo cada vez que realicemos cambios en la configuración para aplicarla.

3.1. Servidor

Es importante que el servidor permita el tráfico a los equipos indicados, por lo que si tenemos una configuración de firewall (ufw/iptables) debemos adaptarla. Si el servicio no está funcionando correctamente podemos comprobar su estado y proceder a solucionar los errores en base a esta información con el comando:

```
sudo systemctl status sshd.service
```

3.2. Cliente

Los clientes de ssh podran acceder con el mismo paquete, tanto al servicio de ssh como sftp, e incluso utilizar pares de claves para acceder sin utilizar una constraseña. Para realizar una conexión empleamos la siguiente estructura:

```
ssh [-i priv_key/-A/-p 22/-(L/R) 8080:localhost:80] usuario@direccion_ip
# -i indicar una clave privada para autenticarse.
# -A utiliza una clave privada cargada anteriormente.
# -p indicar puerto a utilizar en la conexion.
# -L redirige un puerto local a un puerto del servidor.
# -R redirige un puerto del servidor a un puerto local.
```

La misma estructura es aplicable a sftp, donde una vez conectados utilizaremos los comandos **put** y **get**, para cargar o descargar archivos sobre los que el usuario tenga permisos de lectura.

4. Opcionales

Configuraciones convenientes para mejorar la seguridad y permitir otras funcionalidades.

4.1. Generación de Claves

Para generar y usar un par de claves, utilizaremos la función **ssh-keygen**, incluida en el metapaquete ssh. Esta funcion puede ser acompañada de una serie de parámetros para especificar el tipo y tamaño de clave.

```
ssh-keygen -t rsa -b 4096 #Genera una clave RSA de 4096 bits
```

Tambien podemos añadir el parámetro -C seguido de un nombre, para identificar fácilmente la clave.

Ahora que tenemos el par de claves se introduce la pública (.pub) en el **authorized_keys** del usuario al que buscamos conectarnos sin contraseñas. Si hemos generado la clave en el cliente podemos enviar la clave de la siguiente manera:

```
#PowerShell - Windows
type $env:USERPROFILE\.ssh\clave_rsa.pub | ssh ususario@direccion_ip
"cat>>.ssh/authorized_keys"

#Bash - Linux
ssh-copy-id usuario@direccion_ip
```

4.2. Carga de claves en RAM

Para que la clave pueda utilizarse desde el equipo al que nos conectamos, haciendo de puente hacia otros equipos sin necesidad de que este tenga la clave en local, tendremos que realizar la conexión indicando el parámetro -A.
Este parámetro requiere haber añadido anteriormente la clave privada a ssh-agent, el cual tendremos que iniciar antes de poder lanzar el comando ssh-add:

```
#PowerShell - Windows
ssh-agent.exe
ssh-add.exe .ssh\clave_rsa

#Bash - Linux
eval $(ssh-agent)
ssh-add .ssh/clave_rsa
```

Esta es una práctica muy común al trabajar con servicios en la nube como puede ser EC2 de AWS, donde se suele establecer un servidor como Bastión (Servidor intermediario que concentra la mayor seguridad de la red, y es la conexión de la VPC con el exterior).

De esta manera se conecta usando la clave con los equipos de la VPC, sin necesidad de que estos contengan la clave.

Atención: En Windows la clave no se carga correctamente, por lo que es conveniente utilizar programas como PuTTY.