**OPENSSH**

**SSH** **S**ecure **SH**ell es un conjunto de estándares y protocolos de red que permite establecer una comunicación a través de un canal seguro entre un cliente y un servidor. Utiliza una clave pública cifrada para autenticar el servidor remoto y, opcionalmente, permitir a dicho servidor autenticar al usuario

**SSH** provee confidencialidad e integridad en la transferencia de los datos utilizando criptografía y **MAC** (**M**essage **A**uthentication **C**odes)

De modo predeterminado, escucha peticiones a través del puerto 22 **TCP**

**Instalación**

Los paquetes necesarios para la instalación son **openssh-server** y **openssh-client**

**Inicio, parada y reinicio del servicio openSSH**

service ssh start | stop | restart

**Ficheros de configuración**

Se encuentran en el directorio /etc/ssh

sshd\_config configuraciones del servidor **openSSH**

ssh\_config configuraciones del cliente

**Configuración de un servidor openSSH**

|  |  |  |
| --- | --- | --- |
| **OPCIONES GENERALES** | | |
| **OPCIÓN** | **DESCRIPCIÓN** | **VALOR POR DEFECTO** |
| Port | Puerto por el que recibe peticiones el servidor openSSH  Se pueden especificar varias entradas para que openSSH trabaje en diferentes puertos | 22 |
| Protocol | Versión del protocolo soportado por el servidor y orden de preferencia | 2 |
| ListenAddress | Dirección IP local que escucha las conexiones entrantes  Pueden especificarse varias entradas para indicar varias direcciones de red  Se usa cuando el servidor dispone de varias tarjetas de red | Todas |
| Banner | Muestra un mensaje antes de acceder al servidor SSH | vacio |
| **OPCIONES DE USUARIOS Y GRUPOS** | | |
| **OPCIÓN** | **DESCRIPCIÓN** | **VALOR POR DEFECTO** |
| AllowGroups | Lista de nombres de grupos separados por espacios cuyos miembros, sea como grupo  Primario o grupo suplementario, tienen permitido el acceso el sistema mediante SSH  Se pueden utilizar caracteres comodines \* y ¿ | Todos los grupos |
| AllowUsers | Lista de nombres de usuarios separados por espacios cuyo acceso al sistema  está permitido pos SSH  Se puede utilizar de la forma usuario@ordenador, comprobando entonces tanto  El nombre del usuario como el ordenador desde el que se intenta acceder  Se pueden utilizar caracteres comodines \* y ¿ | Todos los usuarios |
| DenyGroups | Lista de nombres de grupos separados por espacios cuyos miembros, sea como grupo  Primario o grupo suplementario, no tienen permitido el acceso el sistema mediante SSH  Se pueden utilizar caracteres comodines \* y ¿ | Ningún grupo |
| DenyUsers | Lista de nombres de usuarios separados por espacios cuyo acceso al sistema no  está permitido pos SSH  Se puede utilizar de la forma usuario@ordenador, comprobando entonces tanto  El nombre del usuario como el ordenador desde el que se intenta acceder  Se pueden utilizar caracteres comodines \* y ¿ | Ningún usuario |
| **OPCIONES DE ACCESO** | | |
| **OPCIÓN** | **DESCRIPCIÓN** | **VALOR POR DEFECTO** |
| LoginGraceTime | Tiempo, en segundos, antes de que se cierra la sesión de autentificación |  |
| PasswordAuthentication | Permite la autentificación mediante password | Yes |
| PermitRootLogin | Permite el acceso de root mediante SSH | Yes |
| PermitEmptyPasswords | Permite el acceso de usuarios sin contraseña | No |
| PubkeyAuthentication | Permite autenticación mediante clave pública/privada | Yes |
| MaxAuthTries | Número máximo de intentos de autenticación por conexión | 6 |
| MaxStartups | Número máximo de conexiones simultaneasen estado de autenticación | 10 |
| UsePAM | Indica si se utiliza PAM para comprobar el acceso de los usuarios  Esta directiva está relacionada con **PasswordAuthentication** | Yes |
| **OPCIONES DE REENVÍO DE CONEXIONES X11** | | |
| **OPCIÓN** | **DESCRIPCIÓN** | **VALOR POR DEFECTO** |
| X11Forwarding | Permite la ejecución remota de aplicaciones gráficas | Yes |

El orden de procesamiento de las opciones de usuarios y grupos es DenyUsers, AllowUsers, DenyGroups, AllowGroups

Si se especifican entradas en DenyUsers o DenyGroups, dichos usuarios o grupos tendrán denegado el acceso mediante **SSH**, mientras se permite el acceso al resto de usuarios y grupos.

Si se especifican entradas en AllowUsers o AllowGroups, dichos usuarios o grupos tendrán permitido el acceso mediante **SSH**, mientras se deniega el acceso al resto de usuarios y grupos.

NOTA Para comprobar la correcta sintaxis del archivo de configuración podemos utilizar el comando sshd -t

**Uso del cliente openSSH**

La sintaxis del cliente **openSSH** es

ssh opciones usuario\_remoto@ordenador\_remoto

Donde

**usuario\_remoto** es el usuario con el que queremos iniciar sesión en el servidor

**ordenador\_remoto** es el ordenador remoto al que queremos acceder

Ejemplo. Conectar al servidor 192.168.1.xx con el usuario usuario

ssh [usuario@192.168.1.xx](mailto:usuario@192.168.1.xx)

La primera vez que nos conectamos a un ordenador remoto, este, se añadirá al archivo known\_host del cliente. Dicho archivo está en el directorio /home/usuario/.ssh/

Opciones

**-p** indica el puerto cuando se configura un puerto diferente al 22

**Reenvío de conexiones mediante openSSH**

Además de proporcionar una interfaz de línea de comandos segura, **SSH** permite redirigir conexiones X11 y puertos inseguros por un canal **SSH** previamente abierto

Reenvío de X11

Para poder reenviar X11, el cliente de **SSH** debe estar ejecutando un servidor X, además, tanto el cliente como el servidor remoto tienen que tener habilitado en su configuración la posibilidad de permitir reenvío X11

Para hacer uso de reenvío X11 habrá que indicarlo en la conexión **SSH** mediante la opción **-X**

Ejemplo. Ejecutar xclock mediante reenvío X11

ssh –X usuario\_remoto@servidor\_remoto

xclock

**Transferencia de archivos mediante openSSH**

La trasferencia de archivos mediante **openSSH** entre dos ordenadores se puede realizar mediante dos programas **SCP** y **SFTP**

**SCP**

**SCP** se utiliza para transferir ficheros entre ordenadores de forma segura y encriptada. Su sintaxis es

scp fichero\_local usuario@ordenador\_destino:/ruta\_fichero\_remoto

Donde

**fichero\_local** es el fichero que quiero enviar

**usuario** es el usuario con el que nos conectaremos al ordenador destino

**ordenador\_destino** es el ordenador al que deseamos enviar el fichero

**:/ruta\_fichero\_remoto** ruta donde enviaremos el archivo

Opciones

**-r** copia recursiva de directorios

**-p** conserva las propiedades de/los archivos (permisos, fechas de creación, fechas de modificación)

**-P** establece puerto de conexión en caso de ser diferente al puerto 22

Ejemplo. Enviar el archivo hola.txt mediante **scp** al directorio /home/usuario/Documentos

scp hola.txt usuario\_remoto@ip\_ordenador\_remoto:/home/usuario\_remoto/Documentos

Ejemplo. Recibir el archivo adiós.txt mediante **scp** al directorio home del ordenador local

scp [usuario\_remoto@ip\_ordenador\_remoto:/home/usuario\_remoto/Documentos/adios.txt /home/usuario\_local/Documentos/](mailto:usuario_remoto@ip_ordenador_remoto:/home/usuario_remoto/Documentos/adios.txt%20/home/usuario_local/Documentos/)

Ejemplo. Recibir todos los archivos del directorio Música del ordenador remoto mediante **scp**

scp [usuario\_remoto@ip\_ordenador\_remoto:/home/usuario\_remoto/Música/\* /home/usuario\_local/Documentos/](mailto:usuario_remoto@ip_ordenador_remoto:/home/usuario_remoto/Documentos/adios.txt%20/home/usuario_local/Documentos/)

**SFTP**

**SFTP** permite abrir una conexión segura e interactiva de **FTP**, pudiendo ejecutar cualquier comando disponible en **FTP** a través de esta conexión encriptada. Su sintaxis es

sftp usuario\_remoto@ordenador\_remoto

Opciones

**-oPort** Especifica el número de puerto del servidor SSH

Ejemplo. Conectar al servidor SSH mediante sftp por el puerto 24

sftp –oPort=24 usuario\_remoto@ordenador\_remoto

Ejercicio. Configuración servicio SFTP “enjaulando” en su HOME al usuario usftp

Consideraciones previas

* El directorio del usuario tiene que pertenecer al usuario y grupo **root**
* Si se quiere permitir subir archivos al servidor, se deberá crear un directorio dentro del propio HOME del usuario
* Las directivas **Match user**, **ChrootDirectory** y **ForceCommand** tienen que especificarse tras la directiva **UsePAM**

**NOTA** Si se quiere encerrar más de un usuario a la vez en su HOME basta agregar tantas directivas **Match user** y **ChrootDirectory** como sea necesario

1. Crear un usuario para la conexión y configurar su HOME

sudo useradd –s /bin/bash usftp

sudo passwd usftp

sudo mkdir /home/usftp

sudo usermod –d /home/usftp usftp

1. Configurar el archivo /etc/ssh/sshd\_config

#Subsystem sftp /usr/lib/openssh/sftp-server

Subsystem sftp internal-sftp

Match user usftp

ChrootDirectory /home/usftp

ForceCommand internal-sftp

**Acceso remoto mediante claves RSA pública/privada**

Existe la posibilidad de permitir que un usuario acceda a un servidor **SSH** utilizando autentificación basada en un par de claves pública/privada

El comando para la creación de estas claves es

ssh-keygen –t tipo

Donde **tipo** puede ser **RSA** o **DSA**. En todo caso se nos pedirá una contraseña de acceso cifrada utilizando **DES3** a la clave privada, la cual puede dejarse vacía si no queremos introducirla cada vez

Los ficheros creados se guardan serán

|  |  |
| --- | --- |
| **FICHERO** | **DESCRIPCIÓN** |
| ~/.ssh/id\_rsa | Clave privada RSA para el protocolo SSH2 |
| ~/.ssh/id\_rsa.pub | Clave pública RSA para el protocolo SSH2 |
| ~/.ssh/id\_dsa | Clave privada DSA para el protocolo SSH2 |
| ~/.ssh/id\_dsa.pub | Clave privada DSA para el protocolo SSH2 |

Cuando queramos acceder a un ordenador remoto, además de que este esté configurado para permitir dicha autentificación, deberemos añadir al fichero **~/.ssh/authorized\_keys** en el usuario del servidor remoto la clave pública con la deseemos acceder

|  |  |  |
| --- | --- | --- |
| **OPCIÓN** | **DESCRIPCIÓN** | **VALOR POR DEFECTO** |
| PubkeyAuthentication | Permite la autenticación mediante claves públicas | yes |
| AuthorizedKeyFile | Fichero con las claves públicas usadas para la autenticación | %h/.shh/authorized\_keys |
| PasswordAuthentication | Permite la autentificación mediante password | yes |

Ejemplo. Crear un par de claves pública/privada **RSA** para acceder a un servidor remoto sin clave de paso

1. En el host cliente (desde donde nos queremos conectar) generamos el par claves pública/privada

ssh-keygen –t rsa

1. Copiamos la clave pública al host servidor (donde nos queremos conectar)

ssh-copy-id –i /home/usuario/.ssh/id\_rsa.pub usuario\_servidor@ip\_servidor

1. Asegurarnos de que el archivo **/etc/ssh/sshd\_config** admite conexiones mediante claves RSA

PubkeyAuthentication yes

#si queremos que todas las conexiones se realicen mediante claves públicas

PasswordAuthentication no

1. Conectarnos al servidor mediante ssh

ssh usuario\_servidor@ipservidor

**Asegurando el servicio openSSH mediante TCPWrappers**

Basta agregar el servicio **openSSH** en el archivo **hosts.allow**

Ejemplo. Permitir la ejecución del servicio **openSSH** a todos los ordenadores de la red

#HOSTS.ALLOW (/ETC)

sshd: ALL

****

**DenyHOSTS** es un script python que permite asegurar entre otros, un servidor **openSSH** de intentos de acceso no autorizados, los cuales, en la mayoría de los casos se basan en ataques de fuerza bruta (método ensayo/error) o ataques basados en diccionario

**Instalación**

El paquete necesario para la instalación es **denyhosts**

**Inicio, parada y re inicio del servicio DenyHOSTS**

service denyhost start | stop | restart

**Ficheros de configuración**

Se encuentran en el directorio /etc/denyhosts.conf

**Opciones de configuración**

|  |  |  |
| --- | --- | --- |
| **OPCIONES GENERALES** | | |
| **OPCIÓN** | **DESCRIPCIÓN** | **DEFECTO** |
| SECURE\_LOG | Fichero log a partir del cual DenyHOSTS extrae información sobre posibles atacantes | /var/log/auth.log |
| HOSTS\_DENY | Fichero donde se filtrarán las direcciones IP filtradas | /etc/hosts.deny |
| PURGE\_DENY | Tiempo en el cual se desbloquearán las direcciones IP filtradas cuando se usa la opción --purge | nunca |
| BLOCK\_SERVICE | Define que servicios se bloquearán  Puede utilizarse la opción ALL | sshd |
| WORK\_DIR | Directorio de trabajo de denyHOSTS | /var/lib/denyhosts |
| **OPCIONES DE FILTRADO** | | |
| DENY\_THRESHOLD\_INVALID | Para una cuenta no existente en el sistema, intentos de autenticación fallidos necesarios para filtrar una dirección IP | 5 |
| DENY\_THRESHOLD\_VALID | Para una cuenta existente en el sistema, intentos de autenticación fallidos necesarios para filtrar una dirección IP | 10 |
| DENY\_THRESHOLD\_ROOT | Para la cuenta de usuario root, intentos de autenticación fallidos necesarios para filtrar una dirección IP | 1 |
| DENY\_THRESHOLD\_RESTRICTED | Para usuarios deshabilitados, intentos de autenticación fallidos necesarios para filtrar una dirección IP  Para utilizar esta opción, se tiene que crear un archivo con las cuentas que queremos denegar explícitamente. | 1 |

**Desbloquear “purgar” direcciones IP filtradas**

1. Detener el servicio denyHOSTS

sudo service denyhosts stop

1. Ejecutar el comando denyHOSTS con la opción --purge

sudo denyhosts --purge

**Ejercicios**

1. Configurar el servidor **openSSH** de forma que cumpla
   1. Permitir el acceso al servidor de todos los usuarios del sistema menos del usuario **root**
   2. Permitir solamente el acceso mediante el par de claves pública/privada
   3. Enjaular a dos usuarios en su HOME permitiendo que puedan subir archivos al servidor mediante **sftp**
   4. No permitir reenvío de conexiones X11
2. Suponiendo que nuestro servidor posee dos interfaces de red, uno público y otro privado de IP 192.168.1.xx, configurar **openSSH** de forma que cumpla
   1. Escuche peticiones tan solo a través de la interfaz privada
   2. Permitir tan solo la versión 2 del protocolo **SSH**
   3. Permitir reenvío de conexiones X11
   4. Permitir tan solo el acceso a los usuarios del grupo administración
3. Configurar el servidor **openSSH** de forma que cumpla
   1. Solo se permita la versión 2 del protocolo **SSH**
   2. Solo permitir el acceso de dos usuarios ussh y usshbis y deniegue a cualquier otro usuario incluido **root**
   3. Al acceder al sistema, se deberá mostrar un mensaje de acceso. Ejemplo gracias por utilizar mi servidor SSH
   4. Permitir el intento de acceso simultaneo al servidor de solo un usuario
   5. Habilitar la posibilidad de que uno de los usuarios pueda acceder, desde otro ordenador, sin necesidad de introducir contraseña