



**ELABORÓ:**

**José Emmanuel Cerón García**

**Luis Alberto Almeraya García**

**Héctor Iván Cabrera Segura**

22 DE MAYO DE 2017

DOCUMENTACIÓN

Manual de Configuración

Contenido

[Introducción 4](#_Toc483436538)

[Requisitos de Instalación 6](#_Toc483436539)

[INSTALACION DE OPENSSH 6](#_Toc483436540)

[Resumen previo a la configuración 6](#_Toc483436541)

[Ubicación del archivo de configuración 6](#_Toc483436542)

[Configuración 7](#_Toc483436543)

[Enjaular un usuario con OpenSSH en Ubuntu 10](#_Toc483436544)

[Opciones adicionales de Seguridad 10](#_Toc483436545)

[Crear un túnel SSH para acceder a un servidor MYSQL Remoto 11](#_Toc483436546)

[X11 Forwarding a través de SSH 12](#_Toc483436547)

[Anexos 14](#_Toc483436548)

[CÓMO PERMITIR EL ACCESO REMOTO A UNA BASE DE DATOS MYSQL 14](#_Toc483436549)

[Referencias Bibliográficas 15](#_Toc483436550)

Copyright © 2017 La Escuela Superior de Cómputo Todos los derechos reservados.

Publicado por Escuela Superior de Cómputo

Av. Juan de Dios Bátiz S/N

Col Nueva Industrial Vallejo, 07738

01 55 5729 6000

www.cic.ipn.mx/

La información en este documento ha sido detalladamente revisada y se considera correcta. Sin embargo, La Escuela Superior de Cómputo no asume responsabilidad por ninguna inconsistencia que pudiera encontrarse en dicho documento. La Escuela Superior de Cómputo no podrá ser responsable por daños directos, indirectos, especiales, incidentales o por consecuencia resultado de un defecto u omisión en el presente documento, aun bajo el aviso de la posibilidad de dicho daño.

Esta publicación contiene información propiedad de La Escuela Superior de Cómputo y ninguna parte de esta publicación puede ser reproducida total o parcial, almacenada en un sistema, o transmitida en cualquier forma o por cualquier medio, sea electrónico, mecánico, fotocopia, grabación o cualquier otro sin el permiso previo por escrito de La Escuela Superior de Cómputo.

Control de versiones del documento

|  |  |  |  |
| --- | --- | --- | --- |
| Versión | Descripción | Fecha | Responsable |
| 1.0 | Manual de Configuración del servidor SSH | 23/05/2017 | Iván Cabrera  Luis Almeraya |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Aprobación de documentos

|  |  |  |  |
| --- | --- | --- | --- |
| Versión | Aprobador | Rol | Firma /Fecha |
| 1.0 | Emmanuel Cerón | IT Coordinator |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Introducción

**¿Qué es SSH?**

Las siglas corresponden a Secure SHell. Sirve para acceder a máquinas remotas, igual que hace telnet, pero de una forma segura ya que la conexión va cifrada. El transporte se hace mediante TCP, por tanto nos garantiza que las órdenes van a llegar a su destino (conectivo, fiable, orientado a conexión).

**Seguridad**

El cifrado de SSH proporciona autenticidad e integridad de los datos transmitidos por una red insegura como internet.

Utiliza llaves públicas para la autenticación en la máquina remota.

SSH no sólo sirve para usar comandos en máquinas remotas, sino para transferencias de ficheros de forma segura ya sea por SCP o sFTP y servicios de escritorio remoto.

Al conectarse un cliente de SSH con el servidor, se realizan los siguientes pasos:

1. El cliente abre una conexión TCP al puerto 22 del host servidor.

2. El cliente y el servidor acuerdan la versión del protocolo a utilizar, de acuerdo a su configuración y capacidades7.

3. El servidor posee un par de claves pública/privada de RSA (llamadas ``claves de host''). El servidor envía al cliente su clave pública.

4. El cliente compara la clave pública de host recibida con la que tiene almacenada, para verificar su autenticidad. Si no la conociera previamente, pide confirmación al usuario para aceptarla como válida.

5. El cliente genera una clave de sesión aleatoria y selecciona un algoritmo de cifrado simétrico.

6. El cliente envía un mensaje conteniendo la clave de sesión y el algoritmo seleccionado, cifrado con la clave pública de host del servidor usando el algoritmo RSA.

7. En adelante, para el resto de la comunicación se utilizará el algoritmo de cifrado simétrico seleccionado y clave compartida de sesión.

8. Luego se realiza la autenticación del usuario. Aquí pueden usarse distintos mecanismos. Más adelante analizaremos los más importantes.

9. Finalmente se inicia la sesión, por lo general, interactiva.

Una de las principales fortalezas de SSH es la seguridad: ni bien se establece la conexión, y antes de la autenticación del usuario, queda establecido un canal cifrado seguro. En adelante, todo el tráfico de la sesión será indescifrable.



# Requisitos de Instalación

Se requiere la previa instalación de:

* Ubuntu 16.04 (64 bits)

# INSTALACION DE OPENSSH

Vamos a usar OpenSSH por tanto vamos a instalarlo:

sudo apt-get install openssh-server

Ahora procedemos a su configuración.

# Resumen previo a la configuración

## Ubicación del archivo de configuración

El archivo de configuración se encuentra en la siguiente ubicación:

sudo gedit /etc/ssh/sshd\_config

Para arrancar el servidor:

sudo /etc/init.d/ssh start

\* Starting OpenBSD Secure Shell server sshd

Para parar el servidor:

sudo /etc/init.d/ssh stop

\* Stopping OpenBSD Secure Shell server sshd

Para reiniciar el servidor:

sudo /etc/init.d/ssh restart

\* Restarting OpenBSD Secure Shell server sshd

# Configuración

Una vez instalado, vamos a configurar el servidor, escribimos en consola:

sudo gedit /etc/ssh/sshd\_config

Y podremos editar sus opciones, a continuación se muestra las partes que se configuraron:

# Package generated configuration file

# See the sshd\_config(5) manpage for details

# Ponemos el puerto a escuchar por el SSH, por defecto es el 22. Deberemos abrir un puerto en nuestro router redirigiendo hacia la IP interna de la máquina donde lo tengamos.

Port 1234

# Usaremos el protocolo 2 de SSH, mucho más seguro, por tanto, forzamos a que siempre conecten por protocolo 2.

Protocol 2

# HostKeys for protocol version 2. El lugar donde se guardan las keys.

HostKey /etc/ssh/ssh\_host\_rsa\_key

HostKey /etc/ssh/ssh\_host\_dsa\_key

#Privilege Separation is turned on for security

UsePrivilegeSeparation yes

# Lifetime and size of ephemeral version 1 server key

KeyRegenerationInterval 3600

ServerKeyBits 2048

# Logging

SyslogFacility AUTH

LogLevel INFO

# Authentication, importante la parte PermitRootLogin…es vuestra decisión

LoginGraceTime 120

PermitRootLogin no

StrictModes yes

RSAAuthentication yes

PubkeyAuthentication yes

#AuthorizedKeysFile %h/.ssh/authorized\_keys

# Don’t read the user’s ~/.rhosts and ~/.shosts files

IgnoreRhosts yes

# For this to work you will also need host keys in /etc/ssh\_known\_hosts

RhostsRSAAuthentication no

# similar for protocol version 2

HostbasedAuthentication no

# Uncomment if you don’t trust ~/.ssh/known\_hosts for RhostsRSAAuthentication

#IgnoreUserKnownHosts yes

# To enable empty passwords, change to yes (NOT RECOMMENDED)

PermitEmptyPasswords no

# Change to yes to enable challenge-response passwords (beware issues with

# some PAM modules and threads)

ChallengeResponseAuthentication no

# Change to no to disable tunnelled clear text passwords

#PasswordAuthentication yes

# Kerberos options

#KerberosAuthentication no

#KerberosGetAFSToken no

#KerberosOrLocalPasswd yes

#KerberosTicketCleanup yes

# GSSAPI options

#GSSAPIAuthentication no

#GSSAPICleanupCredentials yes

X11Forwarding yes

X11DisplayOffset 10

PrintMotd no

PrintLastLog yes

TCPKeepAlive yes

#UseLogin no

#MaxStartups 10:30:60

#Banner /etc/issue.net

# Allow client to pass locale environment variables

AcceptEnv LANG LC\_\*

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

UsePAM yes

MaxAuthTries 2

Si usamos SFTP enjaulado debemos poner esto y comentar la línea (Subsystem sftp /usr/lib/openssh/sftp-server):

Subsystem sftp internal-sftp

Match user servidor

ChrootDirectory /home/jail/home

AllowTcpForwarding no

ForceCommand internal-sftp

## Enjaular un usuario con OpenSSH en Ubuntu

Enjaular a un usuario nos permitirá darle acceso restringido a cierto directorio de nuestro sistema operativo.

Nos bajamos este archivo:

http://www.redeszone.net/app/uploads/cdn/down/soft/redes/make\_chroot\_jail.sh

Y lo ponemos en el directorio raíz por comodidad.

Le asignamos permisos 700:

sudo chmod 700 make\_chroot\_jail.sh

A continuación escribimos:

bash make\_chroot\_jail.sh sanobis

Tecleamos yes, y a continuación metemos la clave que queramos que tenga (estamos creando un usuario nuevo).

Ahora, su directorio enjaulado es /home/jail/home/sanobis

Salimos y entramos por ssh y veremos que efectivamente no podrá salir de ese directorio. No hace falta que reiniciemos el servidor SSH.

## Opciones adicionales de Seguridad

Podemos también instalar el programa fail2ban para banear IPs que hagan muchos intentos de conexión fallidos (que metan mal la clave).

Podemos instalarlo escribiendo:

sudo apt-get install fail2ban

Ahora hacemos:

sudo gedit /etc/fail2ban/jail.local

Y pegamos esto:

[ssh]

enabled = true

port = 1234

filter = sshd

logpath = /var/log/auth.log

maxretry = 3

Y lo iniciamos:

sudo /etc/init.d/fail2ban start

Para ver los LOGS de conexión:

cd /var/log/

gedit auth.log

También tenemos otro programa dedicado exclusivamente al SSH, se llama DenyHost y tiene una gran base de datos de IPs conocidas como atacantes. Con esto estaremos un poco más seguros.

## Crear un túnel SSH para acceder a un servidor MYSQL Remoto

Un túnel SSH consiste en una conexión segura creada a partir de una conexión SSH. En general puede asegurar cualquier transferencia o protocolo inseguro. Por ejemplo, podríamos hacer una transferencia a través de FTP segura usando este tipo de túneles

En este caso nuestro foco no será solo encriptar la comunicación con la base de datos, si no que será darle acceso a un computador, por ejemplo, de escritorio, al servidor de base de datos a través de un servidor que tenga acceso a ella.

En mi caso tengo un servidor mysql local, por lo que asignaremos el puerto 3307 a la base de datos que nos conectaremos. Necesitamos lo siguiente :

1. Dirección IP o Nombre de dominio de la BD.
2. Dirección IP o Nombre de dominio del servidor con acceso a la BD.
3. Obviamente los accesos correspondientes al servidor y a la base de datos.

El comando SSH con el que se genera el túnel quedaría de la siguiente manera.

ssh -L 3307:DIRECCIONBDD:3306 USUARIOSERVER@IPSERVER

1. DIRECCIONBDD es la dirección de la BD.
2. USUARIOSERVER es el usuario del servidor que tiene el acceso a la BD.
3. IPSERVER es la dirección IP del servidor que tiene el acceso a la BD.

Si todo sale bien ahora podremos conectarnos directamente a la BD de manera local usando el puerto 3307. Existe la posibilidad que tu BD no ocupe el puerto 3306, en ese caso simplemente la cambiar para el que corresponda.

**Accediendo a la BD**

Para probar usaremos el cliente MySQL por consola, pero puedes usar MySQL Workbench, PHPMyAdmin o el que estimes conveniente.

mysql -u USUARIO -pPASSWD -h 127.0.0.1 -P 3307

Si tu usuario y password son correctos ya podrías acceder a la BD aunque sea remota como se fuera local y aunque no tuvieras permisos de acceso a través de tu IP. Esto es para mantener la seguridad de tu BD.

## X11 Forwarding a través de SSH

X11 es el servidor gráfico que usan casi todas las distribuciones Linux. Este servidor permite, entre otras cosas, forwarding a través de SSH. Esto significa que es posible ejecutar aplicaciones gráficas de una máquina remota exportando el display a nuestro escritorio. Es decir, la aplicación se ejecuta en el servidor remoto, pero la interfaz gráfica la visualizamos en nuestro escritorio local.

Pasos a seguir:

1.- En el servidor, editar el archivo /etc/ssh/ssh\_config y modificar la opción X11Forwarding para que quede así:

X11Forwarding yes

Luego de este cambio, posiblemente sea necesario reiniciar el demonio ssh. La forma de hacer esto varía de acuerdo a cada distribución Linux. Lo más sencillo es reiniciar la máquina.

2.- En el escritorio local, loguearse al servidor a través de SSH utilizando el parámetro -X:

ssh -X user@hostname

Donde user es el nombre de usuario utilizado para loguearse en el servidor y hostname es el IP o el alias del servidor.

3.- Para ejecutar una aplicación, simplemente hay que hacerlo desde el terminal. Por ejemplo:

Firefox

**Trusted X11 forwarding**

Al habilitar Trusted X11 forwarding es posible acelerar un poco la velocidad de conexión, ya que se evitan algunos pasos vinculados a la seguridad de la misma.

En caso de que la velocidad sea más importante que la seguridad, todo lo que hay que hacer es lo siguiente:

1.- En el servidor, editar el archivo /etc/ssh/ssh\_config y modificar la opción ForwardX11Trusted para que quede así:

ForwardX11Trusted yes

2.- En el escritorio local, loguearse al servidor a través de SSH utilizando el parámetro -Y:

ssh -Y user@hostname

**Compressed X11 forwarding**

En aquellos escenarios en los que la conexión entre el servidor y el cliente no sea la mejor, es posible comprimir los datos enviados por el servidor.

Para ello, a la hora de loguearse al servidor vía SSH hay que agregar el parámetro -C:

ssh -X -C user@hostname

# **Anexos**

## CÓMO PERMITIR EL ACCESO REMOTO A UNA BASE DE DATOS MYSQL

Cuando desarrollo código relacionado con páginas o aplicaciones web, siempre utilizo máquinas virtuales con una configuración exactamente igual que la que tendrá el servidor donde dicha página será alojada. Por otra parte, casi en todos los casos necesito acceso a bases de datos que se encuentren dentro de la máquina virtual, pero MySQL no está configurado por defecto para permitir conexiones remotas. Debido a esto, hay que modificar algunos parámetros de configuración de MySQL, así como dar permisos a un determinado usuario para que pueda acceder.

**Paso 1**

Modificar el archivo de configuración de MySQL:

sudo nano /etc/mysql/my.cnf

Buscaremos las siguientes líneas y se descomentarán en caso de estar comentadas:

skip-external-locking

bind-address = 127.0.0.1

La IP que le vamos a dar a bind-address será 0.0.0.0. En caso de que tengáis una IP específica a la que queráis dar acceso, simplemente ponedla.

**Paso 2**

En este paso se reiniciará el servicio de MySQL para que se apliquen los cambios:

sudo service mysql restart

**Paso 3**

Hay que dar permisos específicos de acceso al usuario de la base de datos. Yo para desarrollo siempre utilizo usuario root y contraseña root, pero cada uno que ponga el suyo. Antes de nada, hay que entrar a la consola de MySQL:

mysql –u root -p

Tras esto, se le dan los privilegios al usuario en cuestión. En este caso root:

GRANT ALL PRIVILEGES ON \*.\* TO root@'%' IDENTIFIED BY 'password';

FLUSH PRIVILEGES;

Gracias a estos sencillos pasos, ya tendremos acceso remoto a la base de datos de un servidor o máquina virtual en la que estemos desarrollando.

# Referencias Bibliográficas

<https://www.redeszone.net/gnu-linux/servidor-ssh-en-ubuntu/>

<https://www.cristiantala.cl/crear-un-tunel-ssh-para-acceder-a-un-servidor-mysql-remoto/>

<https://geekytheory.com/como-permitir-el-acceso-remoto-a-una-base-de-datos-mysql>

<https://blog.desdelinux.net/x11-forwarding-a-traves-de-ssh/>

<https://www.linuxtotal.com.mx/index.php?cont=info_seyre_004>