Configuración de SSH

Reiniciar el servicio SSH

Las más frecuentes manipulaciones en la configuración del servicio SSH pasan por hacer cambios en el fichero `/etc/ssh/sshd_config`. Pero los cambios no tendrán efecto hasta que reinicie el servicio SSH.

En Ubuntu/Debian el reinicio puede ser hecho con la orden:

```
sudo service ssh restart
```

En CentOS/Fedora:

```
sudo service sshd restart
```

Para reiniciar el servicio `ssh/sshd` en Mac OS X, simplemente puede ejecutar las siguientes dos órdenes juntas:

```
sudo launchctl stop com.openssh.sshd
sudo launchctl start com.openssh.sshd
```

Su `sshd` se reiniciará y su nuevo archivo de configuración `ssh` también se cargará. No es aconsejable ejecutar:

```
sudo launchctl reboot com.openssh.sshd
```

en cambio, ya que su computadora se reiniciará inmediatamente.

Opciones de Configuración del Lado del Servidor

Esta sección contiene algunas opciones comunes de configuración del lado del servidor que pueden determinar la forma en que su servidor responde y establece qué tipos de conexiones están permitidas.

Muchas de las operaciones que explicaremos requieren actuar como superusuario en el servidor remoto. La primera intención suele ser conectar al mismo como `root`; pero esto es muy desaconsejable ya que si el acceso es comprometido por alguna vía, ello daría un acceso más vigoroso al sistema. En su lugar deberíamos conectar como un usuario interpuesto, sin privilegios y ya dentro del sistema ganarlos con `su` o `sudo`.

Deshabilitar la Autenticación con Contraseña

Si tiene claves SSH configuradas, probadas y funcionando correctamente, probablemente sea una buena idea deshabilitar la autenticación de contraseña. Esto evitará que cualquier usuario inicie sesión con SSH utilizando una contraseña.

Para hacer esto, conéctese a su servidor remoto y abra el archivo `/etc/ssh/sshd_config` con privilegios de `root` o `sudo`:

```
sudo nano /etc/ssh/sshd_config
```

Dentro del archivo, busque la directiva `PasswordAuthentication`. Si está comentado, descoméntela y ajústela a `no` para deshabilitar los inicios de sesión con contraseña:

```
PasswordAuthentication no
```

Después de realizar el cambio guárdelos y cierre el archivo. Para implementar los cambios, debe reiniciar el servicio SSH.

Cuando haya sido reiniciado el servicio será imposible entrar en las cuentas del mismo vía el password.

Cambiar el Puerto en el Que Se Ejecuta el SSH Daemon

Algunos administradores sugieren que cambie el puerto predeterminado en el que se ejecuta SSH. Esto puede ayudar a disminuir la cantidad de intentos de autenticación a los que su servidor está sujeto desde bots automatizados.

Para cambiar el puerto en el que escucha el demonio SSH, deberá iniciar sesión en su servidor remoto. Abra el archivo `sshd_config` en el sistema remoto con privilegios de `root`:

```
sudo nano /etc/ssh/sshd_config
```

Una vez que esté dentro, puede cambiar el puerto en el que se ejecuta SSH al encontrar la especificación del `Port 22` y modificarla para reflejar el puerto que desea usar. Por ejemplo, para cambiar el puerto a 4444, ponga esto en su archivo:

```
#Port 22
Port 4444
```

Guarde y cierre el archivo cuando haya terminado. Para implementar los cambios, debe reiniciar el demonio SSH.

Limitar a los Usuarios Que Pueden Conectarse a Través de SSH

Para limitar explícitamente las cuentas de usuario que pueden iniciar sesión a través de SSH, puede tomar algunos enfoques diferentes, cada uno de los cuales implica editar el archivo de configuración del demonio SSH.

En su servidor remoto, abra este archivo ahora con privilegios de `root` o `sudo`:

```
sudo nano /etc/ssh/sshd_config
```

El primer método para especificar las cuentas que pueden iniciar sesión es usar la directiva `AllowUsers`. Busque la directiva `AllowUsers` en el archivo. Si no existe, créela en cualquier lugar. Después de la directiva, enumere las cuentas de usuario que deberían poder iniciar sesión a través de SSH:

```
AllowUsers user1 user2
```

Guarde, cierre el archivo y reinicie el demonio para implementar los cambios.

Si se siente más cómodo con la administración de grupos, puede usar la directiva `AllowGroups` en su lugar. Si este es el caso, sólo agregue un único grupo al que se debe permitir el acceso SSH (crearemos este grupo y agregaremos miembros momentáneamente):

```
AllowGroups sshmembers
```

Guarde, cierre el archivo y reinicie el demonio para implementar los cambios.

Ahora, puede crear un grupo en el sistema (sin un directorio de inicio) que coincida con el grupo que especificó escribiendo:

```
sudo groupadd -r sshmembers
```

Asegúrese de agregar las cuentas de usuario que necesite a este grupo. Esto se puede hacer escribiendo:

```
sudo usermod -a -G sshmembers user1
sudo usermod -a -G sshmembers user2
```

Reinicie el demonio para implementar los cambios.

Deshabilitar Inicio de Sesión Como `root`

A menudo es aconsejable deshabilitar completamente el inicio de sesión como `root` a través de SSH después de haber configurado una cuenta de usuario SSH que tiene privilegios de `sudo`.

Para hacer esto, abra el archivo de configuración del demonio SSH con `root` o `sudo` en su servidor remoto.

```
sudo nano /etc/ssh/sshd_config
```

En el interior, busque una directiva llamada `PermitRootLogin`. Si está comentado, descoméntelo. Cambie el valor a `no`:

```
PermitRootLogin no
```

Guarde y cierre el archivo. Para implementar sus cambios, reinicie el demonio SSH.

Permitir Acceso Como `root` para Órdenes Específicas

En algunos casos, es posible que desee deshabilitar el acceso como `root` en general, pero habilítelo para permitir que ciertas aplicaciones se ejecuten correctamente. Un ejemplo de esto podría ser una rutina de respaldo.

Esto se puede lograr a través del archivo de claves autorizadas del usuario raíz, que contiene claves SSH que están autorizadas para usar la cuenta.

Agregue la clave de su computadora local que desea utilizar para este proceso (recomendamos crear una nueva clave para cada proceso automático) al archivo de claves autorizadas del usuario `root` en el servidor. Haremos aquí una demostración con la orden `ssh-copy-id` pero puede usar cualquiera de los métodos de copia de claves discutido en otras secciones:

```
ssh-copy-id root@remote_host
```

Ahora inicie sesión en el servidor remoto. Tendremos que ajustar la entrada en el archivo `authorized_keys`, así que ábralo con acceso `root` o `sudo`:

```
sudo nano /root/.ssh/authorized_keys
```

Al comienzo de la línea con la clave que cargó, agregue un `command=` enumere las órdenes para los que esta clave es válida. Esto debería incluir la ruta completa al ejecutable, más cualquier argumento:

```
command="/path/to/command arg1 arg2" ssh-rsa ...
```

Guarde y cierre el archivo cuando haya terminado.

Ahora abra el archivo `sshd_config` con privilegios de `root` o `sudo`:

```
sudo nano /etc/ssh/sshd_config
```

Busque la directiva `PermitRootLogin` y cambie el valor a `forced-commands-only`. Esto sólo permitirá que los inicios de sesión de clave SSH usen `root` cuando se haya especificado la orden para la clave:

```
PermitRootLogin forced-commands-only
```

Guarde y cierre el archivo. Reinicie el demonio SSH para implementar sus cambios.

Reenvío de Pantallas de Aplicaciones X al Cliente

El demonio SSH se puede configurar para reenviar (forwarding) automáticamente la visualización de aplicaciones X en el servidor a la máquina del cliente. Para que esto funcione correctamente, el cliente debe tener un sistema X Windows configurado y habilitado.

Para habilitar esta funcionalidad, inicie sesión en su servidor remoto y edite el archivo `sshd_config` como `root` o con privilegios de `sudo`:

```
sudo nano /etc/ssh/sshd_config
```

Busque la directiva `X11Forwarding`; si está comentado, descomente. Créelo si es necesario y establezca el valor en `yes`:

```
X11Forwarding yes
```

Guarde y cierre el archivo. Reinicie el demonio SSH para implementar sus cambios.

Para conectarse al servidor y reenviar la pantalla de una aplicación, debe pasar la opción `-X` del cliente al conectarse:

```
ssh -X username@remote_host
```

Las aplicaciones gráficas iniciadas en el servidor en esta sesión deben mostrarse en la computadora local. El rendimiento puede ser un poco lento, pero es muy útil en caso de apuro.

Opciones de Configuración del Lado del Cliente

En esta sección nos centraremos en algunos ajustes que puede realizar usted en el lado del cliente de la conexión.

Definición de Información de Conexión Específica del Servidor

En su computadora local puede definir configuraciones individuales para algunos o todos los servidores a los que se conecta. Esto puede almacenarse en el archivo `~ /.ssh/config`, que su cliente SSH lee cada vez que es llamado.

Cree o abra este archivo en su editor de texto en su computadora local:

```
nano ~/.ssh/config
```

Dentro puede definir opciones de configuración individuales introduciendo cada una con una palabra clave Host, seguida de un alias. Debajo de esto y con sangría, puede definir cualquiera de las directivas que se encuentran en la página de manual de `ssh_config`:

```
man ssh_config
```

Una configuración de ejemplo podría ser:

```
Host testhost
    HostName example.com
    Port 4444
    User demo
```

Seguidamente puede conectarse a `example.com` en el puerto `4444` usando el nombre de usuario "demo" simplemente escribiendo:

```
ssh testhost
```

También puede usar comodines para hacer coincidir más de un host. Tenga en cuenta que las coincidencias posteriores pueden anular las anteriores. Debido a esto, debe colocar sus coincidencias más generales en la parte superior. Por ejemplo, puede predeterminar todas las conexiones para no permitir el reenvío X, con una anulación para `example.com` por tener esto en su archivo:

```
Host *
    ForwardX11 no

Host testhost
    HostName example.com
    ForwardX11 yes
    Port 4444
    User demo
```

Guarde y cierre el archivo cuando haya terminado.

Mantener las Conexiones Vivas Para Evitar la Expiración de la Sesión

Si se encuentra desconectado de las sesiones de SSH antes de estar listo, es posible que su conexión haya expirado.

Puede configurar su cliente para enviar un paquete al servidor cada cierto tiempo para evitar esta situación:

En su computadora local puede configurar esto para cada conexión editando su archivo `~ /.ssh/config`. Ábralo ahora:

```
nano ~/.ssh/config
```

Si aún no existe, en la parte superior del archivo, defina una sección que coincida con todos los hosts. Establezca el `ServerAliveInterval` en `120` para enviar un paquete al servidor cada dos minutos. Esto debería ser suficiente para notificar al servidor que no cierre la conexión:

```
Host *
    ServerAliveInterval 120
```

Guarde y cierre el archivo cuando haya terminado.

Deshabilitar la Comprobación de Host

De forma predeterminada, cada vez que se conecte a un nuevo servidor, se le mostrará la huella digital de la clave de host del demonio SSH remoto.

```
 The authenticity of host '111.111.11.111 (111.111.11.111)' can't be established.
ECDSA key fingerprint is fd:fd:d4:f9:77:fe:73:84:e1:55:00:ad:d6:6d:22:fe.
Are you sure you want to continue connecting (yes/no)? yes
```

Esto está configurado para que pueda verificar la autenticidad del host al que intenta conectarse y detectar instancias en las que un usuario malintencionado puede estar intentando hacerse pasar por el host remoto.

En ciertas circunstancias, es posible que desee deshabilitar esta función. Note que esto puede ser un gran riesgo de seguridad, así que asegúrese de saber lo que está haciendo si configura su sistema de esta manera.

Para realizar el cambio, abra el archivo `~ /.ssh/config` en su computadora local:

```
nano ~/.ssh/config
```

Si aún no existe, en la parte superior del archivo, defina una sección que coincida con todos los hosts. Establezca la directiva `StrictHostKeyChecking` en `no` para agregar nuevos hosts automáticamente al archivo `known_hosts`. Establezca `UserKnownHostsFile` en `/dev/null` para que no advierta sobre hosts nuevos o modificados:

```
Host *
    StrictHostKeyChecking no
    UserKnownHostsFile /dev/null
```

Puede habilitar la verificación caso por caso invirtiendo esas opciones para otros hosts. El valor predeterminado para `StrictHostKeyChecking` es `ask`:

```
Host *
    StrictHostKeyChecking no
    UserKnownHostsFile /dev/null

Host testhost
    HostName example.com
    StrictHostKeyChecking ask
    UserKnownHostsFile /home/demo/.ssh/known_hosts
```

Guarde y cierre el archivo cuando haya terminado.

Multiplexación SSH Sobre una Única Conexión TCP

Hay situaciones en las que establecer una nueva conexión TCP puede llevar más tiempo del que desea. Si está haciendo múltiples conexiones a la misma máquina, puede aprovechar la multiplexación.

La multiplexación SSH reutiliza la misma conexión TCP para múltiples sesiones SSH. Esto elimina parte del trabajo necesario para establecer una nueva sesión, posiblemente acelerando las cosas. Limitar el número de conexiones también puede ser útil por otros motivos.

Para configurar la multiplexación, puede configurar manualmente las conexiones, o puede configurar su cliente para que use automáticamente la multiplexación cuando esté disponible. Demostraremos la segunda opción aquí.

Para configurar la multiplexación, edite el archivo de configuración de su cliente SSH en su máquina local:

```
nano ~/.ssh/config
```

Si aún no tiene una definición de host comodín en la parte superior del archivo, agregue una ahora (como `Host *`). Estableceremos los valores `ControlMaster`, `ControlPath` y `ControlPersist` para establecer nuestra configuración de multiplexación.

El `ControlMaster` debe configurarse en `auto` para permitir automáticamente la multiplexación si es posible. `ControlPath` establecerá la ruta para controlar el socket. La primera sesión creará este socket y las sesiones posteriores podrán encontrarlo porque está etiquetado por nombre de usuario, host y puerto.

Establecer la opción `ControlPersist` en `1` permitirá que la conexión maestra inicial tenga un fondo. E `1` especifica que la conexión TCP debería terminar automáticamente un segundo después de que se cierre la última sesión SSH:

```
Host *
    ControlMaster auto
    ControlPath ~/.ssh/multiplex/%r@%h:%p
    ControlPersist 1
```

Guarde y cierre el archivo cuando haya terminado. Ahora, necesitamos crear realmente el directorio que especificamos en la ruta de control:

```
mkdir ~/.ssh/multiplex
```

Ahora, cualquier sesión que se establezca con la misma máquina intentará usar el socket y la conexión TCP existentes. Cuando existe la última sesión, la conexión se cortará después de un segundo.

Si por alguna razón necesita omitir temporalmente la configuración de multiplexación, puede hacerlo pasando el distintivo `-S` con `none`:

```
ssh -S none username@remote_host
```