**SSH - PROXY**

El proxy Secure Shell más simple para controlar el acceso de su(s) equipo(s) de ingeniería/soporte a servidores privados.

**DEFINICION:**

Esta herramienta es para una empresa que brinda soporte a clientes, pequeños equipos de ingeniería o nuevas empresas. Cualquiera que necesite un gobierno de acceso centralizado, seguro y sencillo a un servidor privado.

Ofrece una solución alternativa a las *claves\_autorizadas*.

**¿Por qué no autorizar\_claves en el servidor?**

Imagine que necesita solicitar acceso de root en el servidor del cliente. Si coloca claves públicas de todos sus ingenieros en el servidor del cliente, debe mantener una lista de servidores del cliente para eliminar estas claves y debe divulgar la lista de su gente.

Todo esto es una mala idea, especialmente cuando corromperá las claves autorizadas en el servidor ejecutando su herramienta de automatización y el cliente simultáneamente.

Esta herramienta permitirá colocar solo una clave pública en el servidor y mantener el acceso a través de esta clave.

En segundo lugar, resuelve un problema de provisión de acceso SSH en el entorno de nube ad-hoc, donde los nuevos servidores van y vienen automáticamente.

Como**beneficios**, obtienes

1. Sin LDAP, Kerberos o cualquier otra tecnología de pesadilla
2. No es necesario compartir la clave privada con todo su equipo, incluidas las personas despedidas
3. Todas las acciones se registran para que pueda encontrar quién ha abandonado la base de datos de producción.

Tenga en cuenta que la solución aún está en desarrollo.

**Características clave**

* Proxy de shell seguro.
* Reenvío de puertos seguro a través de stdio.
* Despliegue listo para usar en entornos basados ​​en Docker.

**Empezando**

El proxy SSH es un demonio que lo ayuda a controlar el acceso de su equipo de soporte a los servidores de los clientes con el siguiente flujo de trabajo:

1. Tú creas el par de claves de tu equipo
2. Dar clave pública a todos los clientes
3. Almacene la clave privada en un servidor privado que ejecute un proxy. El acceso a este servidor debe estar limitado a usted mismo
4. Tome la clave pública de su personal de soporte
5. Subirlos en ese servidor proxy
6. Ahora su material de soporte puede iniciar sesión en el servidor del cliente a menos que revoque este acceso

Use el proxy para controlar el acceso de su equipo de ingeniería a servidores en la nube con un flujo de trabajo similar

1. Utilice la consola para generar pares de claves para su entorno.
2. Cargue la clave privada en un servidor proxy ssh.
3. Tome la clave pública de sus ingenieros (por ejemplo, identidad de github)
4. Cargue claves públicas en ese servidor proxy.
5. Ahora su material de soporte puede iniciar sesión en servidores en la nube a menos que revoque este acceso

**Ejecutando el proxy**

La forma más fácil de ejecutar el proxy Secure Shell son los contenedores Docker, hay imágenes preconstruidas disponibles en flussonic/ssh-proxy. Alternativamente, puede usar**[Erlang escript](http://erlang.org/doc/man/escript.html)** para generar un demonio, pero requiere una instalación de **[Erlang OTP/19 o una versión posterior](http://www.erlang.org/).**

docker run -it --rm --name ssh-proxy \

-p ${CONFIG\_SSH\_PORT}:2022 \

-v ${CONFIG\_SSH\_AUTH}:/opt/data/auth \

-v ${CONFIG\_SSH\_USERS}:/opt/data/users \

flussonic/ssh-proxy

Use variables de entorno u otros medios para configurar el contenedor de proxy

## defines a port used by proxy

export CONFIG\_SSH\_PORT=2022

## location of server's private key

export CONFIG\_SSH\_AUTH=/tmp/ssh/auth

## location of user's publick key. Only these user will be able to build a tunnel

export CONFIG\_SSH\_USERS=/tmp/ssh/users

**Configurar una clave privada**

Cargue una clave privada del equipo (la clave que proporciona acceso a todos los servidores privados) id\_rsaa la ${CONFIG\_SSH\_AUTH} carpeta en el servidor proxy ssh.

**Agregar/revocar acceso de usuarios**

Cargue la clave pública de los usuarios en la ${CONFIG\_SSH\_USERS} carpeta del servidor proxy ssh. Nombre el archivo después del nombre de los usuarios. El acceso del usuario se revoca si elimina esta clave del proxy.

**Establecer sesión de Secure Shell**

Su equipo necesita actualizar ~/.ssh/configel archivo con los detalles del proxy ssh

Host ssh-proxy

HostName 127.0.0.1

Port 2022

User my-user-name

IdentityFile ~/.ssh/my-public-key

Tenga en cuenta que el proxy tiene una sintaxis especial para identificar servidores privados. El nombre de usuario, el host y los puertos deben especificarse como user/host/port.

ssh user/private-host@ssh-proxy

**Reenvío de puertos**

[**El subsistema Erlang SSH**](http://erlang.org/pipermail/erlang-questions/2018-January/094706.html) no admite un reenvío de puerto ssh estándar. El demonio proxy implementa un reenvío de puertos utilizando E/S estándar. Usando una sintaxis especial:

ssh user/private-host~forward-host/port@ssh-proxy

Una vez que se establece la conexión SSH, cualquiera stdinse entrega forward-host/porty su respuesta está disponible en stdoutsu proceso ssh local. Los siguientes scripts lo ayudan a adjuntar ssh stdio a cualquier puerto local.

mkfifo pipe

while [ 1 ]

do

nc -l 8080 < pipe | ssh -T user/private-host~forward-host/port@ssh-proxy | tee pipe > /dev/null

done

**Cómo contribuir**

El proyecto acepta contribuciones a través de solicitudes de extracción de GitHub.

1. bifurcarlo
2. Crea tu rama de características git checkout -b my-new-feature
3. Confirma tus cambios git commit -am 'Added some feature'
4. Empuje a la rama git push origin my-new-feature
5. Crear nueva solicitud de extracción

El desarrollo del proxy requiere **[Erlang OTP/19 o una versión posterior](http://www.erlang.org/).**

Use el siguiente comando para ejecutar el proxy localmente para propósitos de RnD

escript ssh-proxy.erl \

-p 2022 \

-i /tmp/ssh/auth \

-u /tmp/ssh/users \

-t /tmp/ssh/server