Configuración de Túneles

Configuración de Túneles Locales a un Servidor

Tunelizar un tráfico a través de un túnel SSH seguro es una excelente manera de evitar configuraciones de cortafuegos (firewall) restrictivas. También es una excelente manera de cifrar el tráfico de red que de otro modo no estaría cifrado.

Las conexiones SSH se pueden usar para canalizar el tráfico desde los puertos del host local a los puertos de un host remoto.

Una conexión local es una forma de acceder a una ubicación de red desde su computadora local a través de su host remoto. Primero, se establece una conexión SSH a su host remoto. En el servidor remoto, se realiza una conexión a una dirección de red externa (o interna) proporcionada por el usuario y el tráfico a esta ubicación se canaliza a su computadora local en un puerto específico.

A menudo esto se usa para hacer un túnel a un entorno de red menos restringido pasando por alto un cortafuegos. Otro uso común es acceder a una interfaz web "localhost-only" desde una ubicación remota.

Para establecer un túnel local a su servidor remoto, debe usar el parámetro `-L` al conectarse y debe proporcionar tres piezas de información adicional:

- El puerto local donde desea acceder a la conexión de túnel.
- El host al que desea que se conecte su host remoto.
- El puerto al que desea que se conecte su host remoto.

Estos se dan, en el orden anterior (separados por dos puntos), como argumentos para el indicador `-L`. También usaremos el indicador `-f`, que hace que SSH pase a segundo plano antes de ejecutarse y el indicador `-N`, que no abre un shell ni ejecuta un programa en el lado remoto.

Por ejemplo, para conectarse a `example.com` en el puerto 80 en su host remoto, haciendo que la conexión esté disponible en su máquina local en el puerto 8888, puede escribir:

```
ssh -f -N -L 8888:example.com:80 username@remote_host
```

Ahora, si apunta su navegador web local a 127.0.0.1:8888, debería ver cualquier contenido en example.com en el puerto 80.

Si desde el navegador intentamos acceder a la dirección webhttp://localhost:8888 es probable que veamos un mensaje de error (con código éste 404). Es un problema del navegador y debe ser configurado correctamente. Que veamos este mensaje de error es positivo ya que esto significa que la conexión con el servidor se ha podido realizar a traves de nuestra conexión `ssh`.

Si vemos este error es debido a la cabecera `Host` que se manda con cada cabecera `http`. En este caso nuestro navegador está accediendo a la dirección `localhost:8888` y establece en la cabecera `Host` el valor `localhost`. Cuando nuestro servidor redirecciona la petición `http` a través de `ssh`, lo hace con la misma

cabecera y entonces el servidor remoto al que estamos intentando acceder recibe en la cabecera `Host` el valor `localhost`, un valor que no espera.

Podemos arreglar esto modificando la cabecera `http` en el navegador mediante el uso de alguna extensión o configurarlo para que use el proxy de manera adecuada. En nuestro caso vamos a ver que funciona correctamente haciendo una petición a través de la utilidad `curl` del sistema.

Para ello usaremos la orden:

```
curl -L --header "Host: example.es" localhost:8888
```

En esta orden, la opción `-L` se usa para seguir posibles redireccionamientos y `--header` nos permite modificar la cabecera `http`.

Como la conexión está en segundo plano, se deberá encontrar su PID para conseguir eliminarla. Puede hacerlo buscando el puerto que usted reenvió:

```
ps aux | grep 8888


1001 5965  0.0  0.0  48168  1136 ? Ss 12:28 0:00 ssh -f -N -L \
                         8888:example.com:80 username@remote_host
1001 6113  0.0  0.0  13648  952 pts/2 S+ 12:37 0:00 grep --colour=auto 8888
```

Luego puede eliminar el proceso apuntando al PID, que es el número en la segunda columna de la línea que coincide con su orden SSH:

```
kill 5965
```

Otra opción es iniciar la conexión sin la opción `-f`. Esto mantendrá la conexión en primer plano, evitando que use la ventana de terminal durante la redirección. El beneficio de esto es que puede matar fácilmente el túnel escribiendo `CTRL-c`.

Configuración de un Túnel Desde un Servidor Remoto a una Computadora Local

Este tipo de operación puede ser útil, por ejemplo, cuando es necesario permitir el acceso a una red interna que está bloqueada a conexiones externas. Si el firewall permite conexiones fuera de la red, esto posibilitará conectarse a una máquina remota y hacer un túnel del tráfico desde esa máquina a una ubicación en la red interna.

Durante la creación del túnel se especifica un puerto remoto. Este puerto, en el host remoto, se canalizará a una combinación de host y puerto al que se conecta desde la computadora local. Esto permitirá que la computadora remota acceda a un host a través de su computadora local.

Para establecer un túnel remoto a su servidor, debe usar el parámetro `-R` al conectarse y debe proporcionar

tres datos adicionales:

- El puerto por el que el host remoto puede acceder a la conexión de túnel.
- El host al que desea que se conecte su computadora local.
- El puerto al que desea que se conecte su computadora local.

Estos se dan, en el orden anterior (separados por dos puntos), como argumentos para el indicador `-R`. También usaremos el parámetro `-f`, que hace que SSH pase a segundo plano antes de ejecutarse y el indicador `-N`, que no abre un shell ni ejecuta un programa en el lado remoto.

Por ejemplo, para conectarse a `example.com` en el puerto `80` en nuestra computadora local, haciendo que la conexión esté disponible en nuestro host remoto en el puerto `8888`, puede escribir:

```
ssh -f -N -R 8888:ejemplo.com:80 username@remote_host
```

Ahora, en el host remoto, abrir un navegador web en `127.0.0.1:8888` le permitiría ver cualquier contenido en `example.com` a través del puerto `80` de la computadora local.

Un esquema más general de la sintaxis es el siguiente:

```
ssh -R remote_port:site_or_IP_to_access:site_port username@remote_host
```

Como la conexión está en segundo plano, deberá encontrar su PID para eliminarla. Puede hacerlo buscando el puerto de redirección:

```
ps aux | grep 8888

1001 5965  0.0  0.0  48168  1136 ?  Ss  12:28  0:00 ssh -f -N -R \
    8888:example.com:80 username@remote_host
1001 6113  0.0  0.0  13648  952 pts/2    S+   12:37   0:00 grep --colour=auto 8888
```

Luego puede eliminar el proceso apuntando al PID, que es el número en la segunda columna, de la línea que coincide con su orden SSH:

```
kill 5965
```

Otra opción es iniciar la conexión sin la bandera `-f`. Esto mantendrá la conexión en primer plano, evitando que use la ventana de terminal durante la redirección. El beneficio de esto es que puede matar fácilmente el túnel escribiendo `CTRL-C`.

Configuración de un Túnel Dinámico para un Servidor Remoto

Un túnel dinámico es similar a un túnel local ya que permite que la computadora local se conecte a otros recursos a través de un host remoto. Un túnel dinámico hace esto simplemente especificando un solo puerto local. Las aplicaciones que deseen aprovechar este puerto para la tunelización deben poder comunicarse utilizando el protocolo SOCKS para que los paquetes se puedan redirigir correctamente al otro lado del túnel.

El tráfico que se pasa a este puerto local se enviará al host remoto. A partir de ahí, el protocolo SOCKS se interpretará para establecer una conexión con la ubicación final deseada. Esta configuración permite que una aplicación compatible con SOCKS se conecte a cualquier número de ubicaciones a través del servidor remoto, sin múltiples túneles estáticos.

Para establecer la conexión, pasaremos la seña `-D` junto con el puerto local donde deseamos acceder al túnel. También usaremos el indicador `-f`, que hace que SSH pase a segundo plano antes de ejecutarse y el indicador `-N`, que no abre un shell ni ejecuta un programa en el lado remoto.

Por ejemplo, para establecer un túnel en el puerto `7777`, puede escribir:

```
ssh -f -N -D 7777 username@remote_host
```

Desde aquí, puede comenzar a apuntar su aplicación compatible con SOCKS (como un navegador web) al puerto que seleccionó. La aplicación enviará su información a un socket asociado con el puerto.

El método de dirigir el tráfico al puerto SOCKS será diferente según la aplicación. Por ejemplo, en Firefox, la ubicación general es `Preferencias > Avanzado > Configuración > Configuraciones de proxy manuales`. En Chrome, puede iniciar la aplicación con `--proxy-server = flag set`. Deberá utilizar la interfaz `localhost` y el puerto que reenvió.

Como la conexión está en segundo plano, deberá encontrar su PID para eliminarla. Puede hacerlo buscando el puerto que reenvió:

```
ps aux | grep 8888

1001 5965  0.0  0.0  48168     1136 ?    Ss   12:28   0:00 ssh -f -N -D \
                7777 username@remote_host
1001 6113  0.0  0.0  13648  952 pts/2    S+   12:37   0:00 grep --colour=auto 8888
```

Luego puede eliminar el proceso apuntando al PID, que es el número en la segunda columna, de la línea que coincide con su orden SSH:

```
kill 5965
```

Otra opción es iniciar la conexión sin la seña `-f`. Esto mantendrá la conexión en primer plano, evitando que use la ventana de terminal durante la redirección. El beneficio de esto es que puede matar fácilmente el túnel escribiendo `CTRL-C`.

Uso de Códigos de Escape SSH para Controlar Conexiones

Incluso después de establecer una sesión SSH, es posible ejercer control sobre la conexión desde el terminal. Podemos hacer esto con algo llamado "códigos de escape SSH", que permiten interactuar con nuestro software SSH local dentro de una sesión.

Forzar una Desconexión del Lado del Cliente

Trataremos aquí de cómo salir de una sesión bloqueada o congelada. Una de las características más útiles de OpenSSH que pasa desapercibida es la capacidad de controlar ciertos aspectos de la sesión desde dentro de la misma.

Estas órdenes se pueden ejecutar comenzando con el carácter de control `~` dentro de una sesión SSH. Las órdenes de control sólo se interpretarán si son lo primero que se escribe después de una nueva línea, por lo que siempre presione `intro` una o dos veces antes de usar uno.

Uno de los controles más útiles da la capacidad de iniciar una desconexión del cliente. El servidor normalmente cierra las conexiones SSH, pero esto puede ser un problema si el servidor tiene dificultades o si la conexión se ha interrumpido. Al usar una desconexión del lado del cliente, la conexión se puede cerrar limpiamente desde el cliente.

Para cerrar una conexión desde el cliente, usaremos el carácter de control `~`), con un punto. Si su conexión tiene problemas, es probable que se encuentre en lo que parece ser una sesión de terminal atascada. Escriba las órdenes a pesar de la falta de comentarios para realizar una desconexión del lado del cliente:

```
[intro]
~.
```

La conexión debería cerrarse de inmediato, volviendo a su sesión de shell local.

Colocar una Sesión SSH en Segundo Plano

Trataremos aquí de la capacidad de poner una sesión SSH en segundo plano. Para hacer esto, debemos proporcionar el carácter de control (~) y luego ejecutar el método abreviado de teclado convencional para ejecutar una tarea en segundo plano (CTRL-z):

```
[intro]
~[CTRL-z]
```

Esto colocará la conexión en segundo plano y lo regresará a su sesión de shell local. Para volver a su sesión SSH, puede usar los mecanismos convencionales de control de trabajos.

Puede reactivar inmediatamente su tarea en segundo plano más reciente escribiendo:

```
fg
```

Si tiene varias tareas en segundo plano, puede ver los trabajos disponibles escribiendo:

```
jobs
```

esta actuación produciría algo así como:

```
[1]+  Stopped                 ssh username@some_host
[2]   Stopped                 ssh username@another_host
```

A continuación, puede poner en primer plano cualquiera de las tareas utilizando el índice en la primera columna con un signo de porcentaje:

```
fg %2
```

Cambiar las Opciones de Redirección de Puertos en una Conexión SSH Existente

Trataremos aquí de cómo un usuario puede modificar la configuración de redirección de puertos después de que la conexión ya se haya establecido. Esto permite crear o eliminar reglas de redirección de puertos sobre la marcha.

Estas capacidades son parte de la interfaz de línea de órdenes SSH, a la que se puede acceder durante una sesión utilizando el carácter de control (`~`) y `C`:

```
[intro]
~C
```

y esto nos responde con:

```
ssh>
```

Recibirá un símbolo del sistema SSH, que tiene un conjunto muy limitado de órdenes válidas. Para ver las opciones disponibles, puede escribir `-h` desde este indicador. Si no se devuelve nada, es posible que tenga que aumentar la verbosidad de su salida SSH utilizando `~v` varias veces:

```
[intro]
~v
~v
~v
~C
-h
```

y se producirá un diálogo como este:

```
Commands:
  -L[bind_address:]port:host:hostport    Request local forward
  -R[bind_address:]port:host:hostport    Request remote forward
  -D[bind_address:]port                  Request dynamic forward
  -KL[bind_address:]port                 Cancel local forward
  -KR[bind_address:]port                 Cancel remote forward
  -KD[bind_address:]port                 Cancel dynamic forward
```

Como se vé, puede implementar fácilmente cualquiera de las opciones de redirección utilizando las opciones apropiadas. También puede destruir un túnel con la orden `kill` asociado especificado con una `K` antes de la letra de tipo de redirección. Por ejemplo, para matar una redirección local (`-L`), puede usar la orden `-KL`; sólo

necesitará proporcionar el puerto para esto.

Entonces, para configurar un puerto local de redirección, puede escribir:

```
 [intro]
~C
-L 8888:127.0.0.1:80
```

El puerto 8888 en su computadora local ahora podrá comunicarse con el servidor web en el host al que se está conectando. Cuando haya terminado, puede derribar esa redirección escribiendo:

```
 [intro]
~C
-KL 8888
```