Tunneling HOWTO

Jaime Pérez Crespo (japecre at pantuflo.escet.urjc.es)

Última actualización: 11 de Julio de 2003

Motivación

Internet se construyó desde un principio como un medio inseguro. Muchos de los protocolos utilizados hoy en día para transferir datos de una máquina a otra a través de la red carecen de ningún tipo de encriptación o medio de seguridad que evite que nuestras comunicaciones puedan ser interceptadas y espiadas. HTTP, FTP, POP3 y otros muchos protocolos ampliamente usados, utilizan comunicaciones que viajan en claro a través de la red. Esto supone un grave problema, en todas aquellas situaciones en las que queremos transferir entre máquinas información sensible, como pueda ser una cuenta de usuario (nombre de usuario y contraseña), y no tengamos un control absoluto sobre la red, a fin de evitar que alguien pueda interceptar nuestra comunicación por medio de la técnica del hombre en el medio (man in the middle), como es el caso de la Red de redes.

En este pequeño HOWTO explicaremos cómo podemos realizar comunicaciones seguras (mediante encriptación) entre dos máquinas de la red, con los protocolos inseguros de los que ya disponemos, como los mencionados FTP o POP3.

¿Qué es el tunneling?

El problema de los protocolos que envían sus datos en claro, es decir, sin encriptarlos, es que cualquier persona que tenga acceso físico a la red en la que se sitúan nuestras máquinas puede ver dichos datos. Es tan simple como utilizar un sniffer, que básicamente, es una herramienta que pone nuestra tarjeta de red en modo promiscuo (modo en el que las tarjetas de red operan aceptando todos los paquetes que circulan por la red a la que se conectan, sean o no para esa tarjeta). De este modo, alguien que conecte su máquina a una red y arranque un sniffer recibirá y podrá analizar por tanto todos los paquetes que circulen por dicha red. Si alguno de esos paquetes pertenece a un protocolo que envía sus comunicaciones en claro, y contiene información sensible, dicha información se verá comprometida. Si por contra, encriptamos nuestras comunicaciones con un sistema que permita entenderse sólo a las dos máquinas que queremos sean partícipes de la comunicación, cualquiera que intercepte desde una tercera máquina nuestros paquetes, no podrá hacer nada con ellos, al no poder desencriptar los datos.

Una forma de evitar el problema que nos atañe, sin dejar por ello de utilizar todos aquellos protocolos que carezcan de medios de encriptación, es usar una útil técnica llamada tunneling. Básicamente, esta técnica consiste en abrir conexiones entre dos máquinas por medio de un protocolo seguro, como puede ser SSH (Secure SHell), a través de las cuales realizaremos las transferencias inseguras, que pasarán de este modo a ser seguras. De esta analogía viene el nombre de la técnica, siendo la conexión segura (en este caso de ssh) el túnel por el cual enviamos nuestros datos para que nadie más aparte de los interlocutores que se sitúan a cada extremo del túnel, pueda ver dichos datos. Ni que decir tiene, que este tipo de técnica requiere de forma imprescindible que tengamos una cuenta de acceso seguro en la máquina con la que nos queremos comunicar.

Un ejemplo: POP3

Para ilustrar el uso de esta técnica, vamos a poner un sencillo ejemplo de su utilización con servicios de correo que utilizan POP3. Gracias al tunneling, podremos consultar el correo en un servidor, sin que nadie conectado a la red que pueda interceptar la comunicación, pueda obtener nuestra cuenta de usuario o leer los correos. Para ello, comenzamos por abrir una conexión segura contra el servidor de correo, por ssh:

```
[icarus@mordor]$ ssh -L puerto:dirección_servidor:110 nuestro_usuario@dirección_servidor
nuestro_usuario@nombre_del_servidor password: nuestro_password
```

Esto abrirá una conexión de ssh con el servidor de correo, cuyo nombre es nombre_del_servidor y con dirección en Internet dirección_servidor. Por otra parte, nuestro_usuario es el nombre de usuario o login que tenemos en esa máquina para su utilización, y puerto es un puerto local, al que podamos atarnos sin

necesidad de estar identificados como superusuario (aquellos menores de 1000), en el que establecemos una correspondencia con el servicio POP3 prestado por el servidor en su puerto remoto 110.

Una vez identificados y correctamente conectados al servidor remoto, abrimos nuestro cliente de correo favorito, como puede ser evolution, kmail, o cualquiera que sea el que utilicemos. Editamos ahora las propiedades de la cuenta que tenemos en el servidor en cuestión, y en el apartado que el programa disponga para que le indiquemos la dirección del servidor POP o POP3 tecleamos localhost:puerto, sustituyendo a la dirección del servidor, y donde puerto es el puerto que indicamos previamente al abrir la conexión por ssh.

Hecho esto, cada vez que leamos el correo, lo haremos encriptando la comunicación a través del túnel creado mediante ssh, lo cual nos salvaguarda de miradas inoportunas. Es importante tener en cuenta que la conexión por ssh que establecimos en un principio, debe estar abierta durante todo el tiempo que mantengamos la comunicación, y no sirve establecer dicha conexión e inmediatamente cerrarla.

Para facilitar aún más las cosas, podemos crear un script que abra la conexión y arranque nuestro programa de correo todo en uno, de forma que tecleando un simple comando podamos consultar el correo mediante túneles, o incluso asociarle un icono que arranque dicho script. A continuación se muestra un posible ejemplo, con evolution como cliente de correo, y 10110 como puerto habitual para el otro extremo del túnel:

```
#!/bin/bash
evolution &
ssh -L 10110:dirección_servidor:110 nuestro_usuario@dirección_servidor
```

Otro ejemplo: FTP

Como los más despiertos ya se habrán percatado, esta técnica sirve para cualquier servicio no seguro que queramos utilizar a través del túnel. FTP es otro ejemplo igualmente válido de protocolo inseguro ampliamente utilizado, que podemos mejorar gracias a la técnica del tunneling. Igual que antes, lo primero que haremos será abrir una conexión ssh con la máquina que alberga el servidor FTP:

```
[icarus@mordor]$ ssh -L puerto:dirección_servidor:21 nuestro_usuario@dirección_servidor
nuestro_usuario@nombre_del_servidor password: nuestro_password
```

Como se puede observar, el establecimiento de la conexión ssh es exactamente el mismo que el anterior, salvo por una particularidad, que consiste en el puerto remoto indicado, en este caso el 21, puerto asociado por convenio con el servicio FTP. En caso de que el servidor FTP remoto estuviese atado en cualquier otro puerto, deberíamos indicar el puerto al que esté atado en lugar del 21. Una forma rápida de ver los puertos asociados comúnmente con ciertos servicios en Internet, es la consulta del fichero /etc/services de todas las máquinas Unix:

```
[icarus@mordor]$ cat /etc/services | grep nombre_servicio
```

Abierto el túnel sobre el que realizar las transferencias, podemos arrancar nuestro programa de correo, como puede ser gftp, o por supuesto utilizar el comando ftp. Si usamos un software como puede ser gftp, nos aseguraremos de indicar como host al que conectar localhost, y como puerto, el puerto local al que hemos mapeado el puerto 21 remoto. Si por contra usamos directamente el comando ftp:

```
[icarus@mordor]$ ftp localhost puerto
```

En este caso indicamos como de costumbre localhost como dirección, y puerto el puerto al que hicimos el mapeo mediante ssh. En este caso particular del FTP, es necesario mencionar que es imprescindible el uso del modo pasivo, usemos el cliente que usemos, ya que todas las transferencias las haremos por el túnel, que comprende un sólo puerto remoto, de modo que no podemos usar el modo port al necesitar éste abrir conexiones en diversos puertos.

Del mismo modo que en el ejemplo con POP3 anterior, podemos realizar un script de shell que nos facilite la tarea de conectar de forma remota a nuestro servidor de FTP, a través de la conexión ssh, y que nos abra por ejemplo gFTP:

```
#!/bin/bash
gftp &
ssh -L 10021:dirección_servidor:21 nuestro_usuario@dirección_servidor
```

Enlaces Interesantes

Pá gina de manual de ssh: http://www.physik.uni-regensburg.de/edv/unix/ux_saccess/man/ssh.html
Pá gina de manual de ftp: http://www.nevis.columbia.edu/cgi-bin/man.sh?man=ftp
Ximian Evolution: http://www.ximian.com/products/evolution/
gFTP: http://gftp.seul.org/

Licencia

Este documento público puede ser distribuido y/o traducido libremente, siempre y cuando en el documento resultante se incluyan los datos del autor indicados en la cabecera, esta sección de licencia, y la url del documento original:
http://pantuflo.escet.urjc.es/~japecre/trabajos/independientes/docs/tunneling.php?idioma=es