

La página de login del hotel se puede sortear con la ayuda de algunos trucos de entunelamiento

# VISIÓN DE TÚNEL

Aleksandar Jovanovic, 1294

Los expertos llevan años diciendo que el bloqueo de los puertos conocidos no es ninguna medida de seguridad real, pero la creencia sobrevive a pesar de sus esfuerzos. Este simple experimento, llevado a cabo por un inquisitivo guerrero de la carretera, contiene una útil moraleja para todo aquel al que le haya surgido la necesidad de tomar algún atajo en cuanto a seguridad de redes. **POR TOBIAS EGGENDORFER**

Las personas que pasan tiempo en la carretera con su portátil, como yo mismo hago, probablemente ya estén familiarizadas con los precios completamente abusivos que cobran algunos hoteles en concepto de acceso a Internet. El uso de la WLAN a veces llega a suponer hasta una cuarta parte de la factura del hotel, algo realmente molesto, especialmente cuando tiene que salir del bolsillo de uno.

Una buena tarde, hace tiempo, mientras me alojaba en la rama de Munich de una conocida cadena de hoteles, mi perplejidad se vio desplazada por una curiosidad imperiosa. Estos sistemas de acceso, pensé, no pueden ser tan sofisticados. Después de todo, el método es bastante sencillo. Hasta que nuestra dirección MAC no es aprobada por el sistema, todas las peticiones web se redirigen a una página de inicio.

La empresa que gestiona el soporte técnico del hotel usa un método bastante primitivo para bloquear los puertos IMAP, IMAPS y POP. Al probar este sistema con un servidor de IMAP, me percaté de que el cliente realmente establecía una conexión con el sistema de destino.

Sin embargo, tan pronto como se establece la conexión, los datos de la respuesta no alcanzaban el punto de origen. Este com-

portamiento hace que la mayoría de los clientes de correo abandonen desesperados.

Mientras me hallaba investigando el sistema, tuve un golpe de suerte: Aunque no estaba registrado, mi cliente de mensajería instantánea se conectó y pude intercambiar mensajes con mis contactos.

¿Se trataba de un pequeño lapso o de una vulnerabilidad del sistema? Mi cliente de MI no usa el puerto 5190, que es el que se usa con la mayoría de los protocolos de MI. En lugar de ése utiliza el 443. A veces ayuda configurar los protocolos de MI de modo que usen el puerto 443 para esquivar los molestos cortafuegos en los viajes.

HTTP suele estar abierto, independientemente de lo restrictiva que sea la conexión, suponiendo que nos hayamos registrado correctamente. Pero en el hotel, ¡ni siquiera me había registrado! Tuve entonces la sospecha de que el puerto 443 de HTTPS no estaba siendo enrutado a través del proxy, sino que en vez de eso, disfrutaba de acceso directo. Pronto pude confirmar mis sospechas: Al usar HTTPS no se me presentaba la página de inicio.

## Conviene Estar Preparados

Al volver a casa, decidí armarme para mi próxima estancia en hotel. Necesitaría un

proxy HTTPS ubicado en un servidor en algún punto de Internet y que aceptase tráfico en el puerto 443. En Perl es fácil de escribir mediante el módulo de CPAN `HTTP::Proxy`.

Una vez instalado el módulo con CPAN,

```
perl -MCPAN -e 'install HTTP::Proxy'
```

seguí el código de ejemplo de `HTTP::Proxy` [1] para crear el mini proxy mostrado en el Listado 1. Aún con todo, esta solución sim-

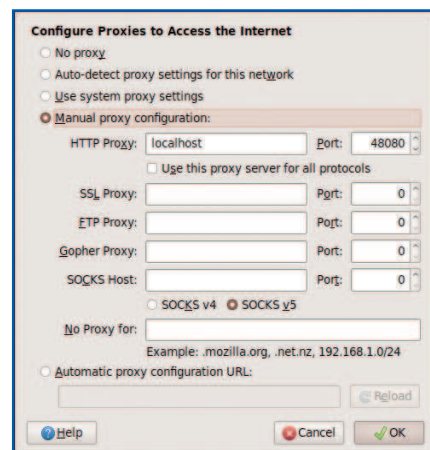


Figura 1: Una vez excavado el túnel, los guerreros de la carretera han de introducir el proxy HTTP.

# Learn

about the latest **KDE** and **Free Desktop Technology** during a two day conference

# Be inspired

by prominent **KDE** developers during 5 days of workshops and coding sessions

# Connect

with key contributors from the **Free Desktop** community and companies

# Enjoy

the unique atmosphere at one of the largest international **Free Software** events

# Join us

for Akademy 2010 and register for free at

<http://akademy.kde.org/>



Organised by KDE e.V. and COSS



**Tampere**  
Finland

Picture of Tampere copyright:  
City of Tampere/Sami Helenius



ple presentaba un par de problemas. Para que el programa pudiera abrir el puerto 443, debía ejecutarse con privilegios de root. Por razones de seguridad, no es buena idea en instalaciones permanentes; mi aplicación de servidor casero sería un blanco fácil al alcance de cualquier máquina con acceso a Internet. Además, la máquina sólo haría de proxy HTTP (una restricción innecesaria).

Aún así, con un simple

```
./http_proxy.pl &
```

bastaría para hacer las pruebas de forma local. Introduje en el navegador la dirección IP de mi servidor a modo de proxy HTTP. Los navegadores más competentes nos permiten especificar diferentes proxies para los distintos protocolos, pero en este caso sólo tenemos que configurar el proxy HTTP; los campos correspondientes al resto de protocolos, especialmente el de HTTPS, deben quedar vacíos.

## Excavando un Túnel

Los experimentos con el proxy fueron bien, pero aún buscaba un método superior que me permitiera trabajar otros protocolos aparte de HTTP. La forma de hacerlo estaba clara: necesitaría un túnel SSH en el que el portátil actuase como cliente, estableciese una conexión por SSH a un servidor de Internet y redirigiese un par de puertos bajo demanda.

Para que SSH escuchase en el puerto de HTTPS además de en el puerto 22, añadí una línea

```
Port 443
```

al archivo `/etc/ssh/sshd_config` y reinicié el demonio con:

```
kill -HUP `cat /var/run/sshd.pid`
```

Se puede configurar SSHD para que acepte conexiones SSH en varios puertos simultáneamente. En mi caso fue útil porque compartiría mi túnel de acceso SSH con un par

### Listado 1: `http_proxy.pl`

```
01 #!/usr/bin/perl
02 use HTTP::Proxy;
03 my $proxy = HTTP::Proxy->new(
04   port => 443);
04 $proxy->start;
```

de colegas, pero sin permitirles acceder a la manipulación de puertos planeada.

Ahora necesitaba mover el proxy del puerto 443 a, digamos, el puerto sugerido en el código de ejemplo, el puerto 3128, que el servidor puede usar sin privilegios de root:

```
ssh tunnel.example.com -p 443
-L48080:localhost:3128
-f './http_proxy.pl &
```

Con esta línea le decimos a SSH que se conecte al puerto 443 de la máquina especificada y que una vez allí ejecute el script Perl. SSH establecerá además un túnel cifrado desde el puerto local 48080 a localhost, como se ve desde el servidor remoto de SSH, en el puerto 3128. La opción `-f` pone el demonio de SSH en segundo plano antes de ejecutar el comando y después de que haya sido introducida la contraseña, mientras que `&` enviaría a SSH a segundo plano inmediatamente. Ahora ya podemos poner como proxy HTTP en el navegador la dirección localhost 48080 (Figura 1).

## En Camino de Nuevo

De vuelta en el hotel de Munich, sólo me faltaba probar el tinglado. Como esperaba, el cliente de SSH y el servidor conectaron sin problemas a través del puerto 443, creando un túnel que traspasaba todas las restricciones, permitiéndome navegar por Internet de manera gratuita.

Evidentemente, se puede ampliar este método para acceder al correo:

```
ssh tunnel.example.com
-p 443 -L48080:localhost:3128
-L60110:pop.example.com:110
-L60993:imap.example.com:993
-L60025:smtp.example.com:25
-f './http_proxy.pl &
```

El uso de un túnel SSH para recibir el correo a través de protocolos desprotegidos, como POP3, no deja de ser útil. Cualquier persona que viaje con frecuencia debería hacerlo. También se puede proteger la autenticación SMTP mediante el túnel SSH. Al mismo tiempo, se puede evitar el bloqueador de spam que instalan algunos proveedores para los servidores SMTP externos.

## Alternativas

Como hemos podido ver, para mis experimentos se ha usado el método del servidor

de SSH, pero podría haberse elegido cualquier otra cosa. Por ejemplo, con

```
ssh tunnel.example.com -p 443
-X -f
'/usr/local/mozilla/mozilla &
```

ejecutaríamos Mozilla en la máquina remota para navegar apoyándonos en un servidor externo. El problema es que el volumen de tráfico necesario hace que la experiencia no sea tan agradable. Redirigiendo al puerto 443 los protocolos utilizados por VNC o NX [2] conseguiremos un método un poco más rápido.

También podríamos tener un servidor OpenVPN escuchando en el puerto TCP 443 de casa. OpenVPN es capaz de entunelar una red completa a través de un puerto TCP or UDP, por lo que podemos conectar el portátil a la red de casa. OpenVPN puede además servir de ayuda redirigiendo todo el tráfico de red sobre VPN [3].

## ¿Qué Pueden Hacer los Proveedores?

¿Cómo podría haber evitado el proveedor este problema? Hasta donde yo sé, el sistema hace uso intensivo de proxies que distinguen entre usuarios de pago y usuarios no autorizados a través de sus direcciones IP o MAC.

Con un simple cortafuegos se puede mejorar la situación, bloqueando todos los puertos a las máquinas de las personas que aún no han pagado. De este modo no se evitaría, sin embargo, que los huéspedes monitoricen el tráfico en busca de dirección IP y direcciones MAC acreditadas con el objeto de suplantar sus conexiones; pero es un problema del que ya adolece también el sistema actual. La complejidad y la seguridad no suelen ir a la par.

Inmediatamente después de completar mis pruebas, informé al hotel. El recepcionista prometió notificarlo al proveedor. ¡Quién lo iba a decir!, en mi siguiente visita a Munich, el problema estaba resuelto. ■

## RECURSOS

- [1] HTTP::Proxy: <http://search.cpan.org/~book/HTTP-Proxy-0.24/lib/HTTP/Proxy.pm>
- [2] "X Rápida – Servicios de Terminal con NX", por Markus Feilner, Linux Magazine, Número 35, pág. 21, [http://www.linux-magazine.es/issue/35/021-024\\_NXLM35.crop.pdf](http://www.linux-magazine.es/issue/35/021-024_NXLM35.crop.pdf)
- [3] OpenVPN HowTo: <http://openvpn.net/index.php/open-source/documentation/howto.html>