Práctica 1.3

Configuración de un proxy directo: TinyProxy

Grado en Ingeniería Informática

2021-2022

Práctica 1.3 – Configuración de un proxy directo: TinyProxy

Grado en Ingeniería Informática

Objetivos

Esta práctica está pensada para que el estudiante aprenda a desplegar un proxy directo.

Índice

- Instalación, documentación y configuración
- Configuración del firewall
- Proxy configurado en cliente
- Proxy transparente
- Seguridad del firewall de inspección de estado y del proxy directo

Calificación

Esta práctica se califica con 3 puntos máximo y hace media con el resto de prácticas de la asignatura

Equipo

Nombre y Apellidos	Rol
Notkero Gómez FernándezF	*Programador/a*
	Analista

La misión principal del *Programador/a* es escribir todos los comandos de la práctica en el orden correcto para conseguir un buen aprovechamiento de los objetivos de la misma. Además, es el/la encargado/a de contestar el test asociado a la práctica.

La misión principal de *Analista* es tomar nota del trabajo que está haciendo su pareja de prácticas e intentar buscar la respuesta a todas las preguntas que se plantean en la práctica. Además, se encarga de auditar cada uno de los intentos de aprobar el test de la práctica marcando los fallos previos y buscando en sus notas la respuesta correcta.

Este cuadernillo de prácticas podrá ser requerido por los profesores y profesoras de la asignatura en cualquier momento a lo largo del curso para comprobar y/o recalificar el aprovechamiento del laboratorio.

AVISO LEGAL

i. Toda la información proporcionada en esta práctica es solo para fines educativos. Los profesores/as no son responsables en ningún caso del uso indebido de esta información por parte de los estudiantes de la asignatura de Redes y Seguridad II.

ii. Esta práctica está pensada dentro del diseño docente de la asignatura Redes y Seguridad II donde además del material aquí expuesto se imparte a los estudiantes conceptos de leyes y de responsabilidad. No está pensada para que sea realizada fuera del contexto de la asignatura, por lo que los profesores/as no nos hacemos responsables de la difusión y/o utilización indebida por parte de terceros que no tengan ninguna relación con la asignatura.

iii. Esta práctica está diseñada para conocer herramientas de auditoría y aprender sobre la manera en que actúan posibles atacantes.

iv. Los estudiantes pueden probar lo expuesto en esta práctica fuera del laboratorio de la asignatura, aunque siempre han de hacerlo sobre una red de su propiedad y bajo su propio riesgo. Realizar intentos de escaneo de puertos o simulaciones de ataques (sin permiso) sobre redes que no le pertenecen es ilegal.

v. Para poder utilizar el material y la información de esta práctica fuera del entorno educativo para el que fue creada, los estudiantes deberán firmar un acuerdo específico de auditoría de seguridad con la persona responsable de los equipos donde se va a utilizar. En ese acuerdo deben aparecer claramente las pruebas específicas que se van a realizar, pudiéndose referenciar específicamente este material. En ningún otro caso fuera del estrictamente educativo permitimos la referencia directa o indirecta a este material.

vi. Consulte las leyes de su país antes de acceder o utilizar estos materiales. En el contexto en el que se imparte la asignatura de Redes y Seguridad II es importante tener presente el artículo 197 de la Ley Orgánica 1/2015: "El que [...] **vulnerando las medidas de seguridad** establecidas para impedirlo, y **sin estar debidamente autorizado, acceda o facilite a otro el acceso al conjunto o una parte de un sistema de información** o se mantenga en él en contra de la voluntad de quien tenga el legítimo derecho a excluirlo [...]" "El que [...] **sin estar debidamente autorizado, intercepte transmisiones no públicas de datos informáticos** que se produzcan desde, hacia o dentro de un sistema de información [...]"

Madrid, a 15 de febrero de 2020

Configuración de la red

En la práctica 1.3 vamos a usar la estructura de red de la figura 1.

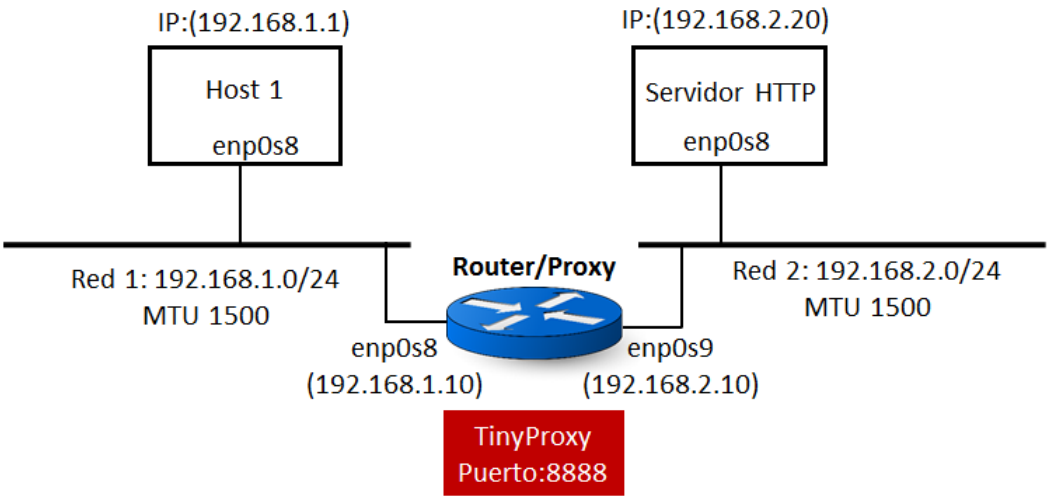

Figura 1

A continuación, vamos a describir los **pasos a seguir para configurar nuestro entorno de trabajo**:

> Recuerda: si has decidido **reutilizar las máquinas de la práctica 1.2 (**Host 1, Router 1 y Servidor), para configurar dichas máquinas tienes que empezar directamente en el **paso 4**
>
> Si no quieres usar las máquinas de la práctica 1.2 tienes que realizar **3 clonaciones** de la máquina virtual rys1920.ova (usuario:usuario/contraseña:usuariorys): Router 1, Host 1 y Servidor HTTP (recuerda **reinicializar las direcciones MAC** y realizar **clonación enlazada**).

1. Comprueba que la máquina tiene los adaptadores de red que se necesitan habilitados y conectados a la red correspondiente (por ejemplo, Host 1 debe tener el adaptador 1 conectado a NAT y el adaptador 2 conectado a la red interna Red 1).
2. Arranca la máquina y configura sus interfaces de red. Recuerda que vamos a hacer que la configuración sea persistente por lo que tendremos que modificar el fichero /etc/network/interfaces. Abrir el fichero con un editor (como sudo) y, en el caso de Host 1, escribir (después de la última línea escrita):

```
auto enp0s8
iface enp0s8 inet static
address 192.168.1.1
netmask 255.255.255.0
```

3. Graba y sal del editor. Después ejecuta en la ventana de comandos:

```
$ sudo /etc/init.d/networking reload
```

Si la configuración del interfaz ha fallado, nos dirá a qué ha sido debido el error. Si todo ha ido bien, ejecutar:

```
$ sudo /etc/init.d/networking restart
```

Si la máquina no ha cogido bien la configuración de red, reiníciala.

4. Configura la tabla de encaminamiento. Por ejemplo, para Host 1:

   `$ sudo route add -net 192.168.2.0/24 gw 192.168.1.10` (Router 1 es el encaminador hacia la Red 2)

   NOTA: Si la máquina que estás configurando es reutilizada y guardaste el estado probablemente ya lo tengas configurado, compruébalo con `$ sudo route -n`

5. D**esactiva** el **network-manager** en la máquina virtual**:**

   ```
   $sudo service network-manager stop
   ```

6. **Desactiva** el interfaz conectado a **NAT en la máquina virtual**:

   ```
   $ sudo ifconfig enp0s3 down
   ```

7. Si la máquina que estás configurando es reutilizada asegúrate que no tiene ninguna regla iptables y que la política por defecto de todas sus cadenas es ACCEPT.

   Para mostrar las reglas:

   ```
   $ sudo iptables -L -v
   ```

   Para quitar las reglas se usa la opción -F:

   `$ sudo iptables -F,` (vacía todas las cadenas de una vez)

   Para poner política por defecto ACCEPT:

   ```
   $ sudo iptables -P FORWARD ACCEPT
   $ sudo iptables -P INPUT ACCEPT
   $ sudo iptables -P OUTPUT ACCEPT
   ```

8. Repite los mismos pasos con Servidor HTTP y Router 1 (para esta máquina no hay que definir ningún encaminamiento). No te olvides de activar el *forwarding* en Router 1:

   ```
   $ sudo sysctl -w net.ipv4.ip_forward=1
   ```

9. Comprueba que todas las máquinas están correctamente conectadas, haciendo, por ejemplo, un ping de Host 1 a Servidor HTTP.

10. **Arranca el servidor** en la máquina Servidor HTTP:

    ```
    $ sudo python -m SimpleHTTPServer 80
    ```

No cierres la ventana del terminal donde has arrancado el servidor o dejará de funcionar

Si necesitas ejecutar comandos en esta máquina tienes que abrir otra terminal

Documentación y configuración

En esta práctica vamos a usar un proxy que trabaja en la capa de aplicación y, por tanto, será específico de un determinado protocolo. En nuestro caso, necesitamos un proxy HTTP y hemos elegido Tinyproxy (https://tinyproxy.github.io/), por haber sido diseñado para ser rápido y ligero. Proporciona filtrado de URLs, monitorización de conexiones, control de acceso, inserción y borrado de cabeceras HTTP...

Ejercicio 1: Configuración de Tinyproxy.

1. Consulta la página del manual y revisa su configuración en /etc/tinyproxy/tinyproxy.conf.

   Contesta las siguientes preguntas:

¿Sobre qué puerto trabaja Tinyproxy?: 8888

¿Qué función tiene la línea Listen? Si tienes varias interfaces te permite unirte solo a una.

¿Qué función tiene la línea Bind? Indica la interfaz usada para conexiones de salida.

¿Qué ficheros están activos por defecto? ¿Cuál es su función?

Archivos de error para cada error típico, DefaultErrorFile para cuando no hay archivo HTML, StatFile para saber el estado del servidor, LogFile para saber donde se loggea la información, PidFile para saber el PID del hilo principal.

¿Qué función tiene la línea Allow?

Personalizar los controles de autorización, la acción por defecto es DENY, cuando se indica pasa a ser ALLOW.

¿Qué función tiene "Anonymus Cookie"?

Para que se active el proxy anónimo y se permitan las cookies de manera anónima cuando se requiere.

Si has comprendido el significado de la línea Allow en el archivo de configuración del Tinyproxy entenderás que tienes que modificar la configuración por defecto para que se ajuste a la IP de tu red, es decir, tienes que indicar que la red que tiene permiso para acceder a Tinyproxy es Red 1.

2. Edita el archivo de configuración y añade la Red 1 a la lista de redes permitidas (es necesario hacerlo con sudo). Apunta cómo ha quedado tu configuración (se pueden obviar las líneas comentadas):

```
# Allow: Customization of authorization controls. If there are any
# access control keywords then the default action is to DENY. Otherwise,
# the default action is ALLOW.
#
# The order of the controls are important. All incoming connections are
```

```
# tested against the controls based on order.
#
Allow 127.0.0.1
#Allow 192.168.0.0/16 172.16.0.0/12 10.0.0.0/8
#Allow 192.168.1.0/24
```

3. Ahora asegúrate de que Tinyproxy lee correctamente el fichero de configuración que has modificado:

```
$ sudo /etc/init.d/tinyproxy reload
```

4. Arranca Tinyproxy con la nueva configuración:

```
$ sudo /etc/init.d/tinyproxy restart
```

5. Asegúrate de que el proxy está funcionando y listo para aceptar conexiones, consultando el archivo de log de Tinyproxy:

```
$ sudo tail -n 25 /var/log/tinyproxy/tinyproxy.log
```

Apunta aquí lo que ves en el archivo de log:

¿En qué parte del fichero de configuración se indica que el proceso tinyproxy va a crear 10 hijos? ¿Se podría modificar este valor?

Configuración del Firewall en Router 1

En la práctica 1.1 se explicó que la configuración más habitual de un firewall es una política de descarte por defecto junto a las reglas que definen el tráfico que está permitido. En esta práctica necesitamos configurar el firewall de Router 1 para que **todo el tráfico HTTP entre los hosts de Red 1 (Host 1) y el Servidor HTTP pasen por el proxy**, que estará instalado en Router 1. Además, **solo se permitirá pasar por el router el tráfico que haya sido iniciado en la Red 1**, es decir, no se aceptará ningún paquete que venga de Red 2, salvo que pertenezca a una conexión ya establecida.

Ejercicio 2: Reglas Iptables para la configuración de un firewall de inspección de estado y para un proxy.

1. Comprobamos que la política de iptables que tenemos por defecto es aceptar todo:

```
$ sudo iptables –L -v
```

2. Cambiamos la política por defecto de iptables en todas las cadenas para que descarte cualquier paquete:

```
$ sudo iptables -P INPUT DROP
$ sudo iptables -P OUTPUT DROP
$ sudo iptables -P FORWARD DROP
```

Con esta configuración, ningún paquete puede llegar ni salir de Router 1 ni atravesarlo.

3. Comprueba ejecutando `$ ping 192.168.2.20` y `$ wget 192.168.2.20`, que Host1 no puede realizar ninguna conexión con el servidor.

4. Primero vamos a introducir una serie de reglas en la cadena FORWARD que **permitan establecer cualquier tipo de conexión desde cualquier máquina de Red 1 con el exterior, salvo conexiones HTTP porque estas deberán ir a través del proxy.** Para ello, hay que introducir las siguientes reglas:

Cadena FORWARD:

- Reenviar paquetes recibidos por la interfaz interna con IP origen Red 1.

```
sudo iptables -A FORWARD -s 192.168.1.0 -i enp0s8 -j ACCEPT
```

- Reenviar paquetes que tengan conexiones establecidas.

```
sudo iptables -A FORWARD -m state --state ESTABLISHED -j ACCEPT
```

Con estas dos reglas deberías poder conectarte a servidor. Compruébalo y a continuación añade la siguiente regla:

- No reenviar paquetes recibidos por la interfaz interna que son conexiones al servidor HTTP.

¿Por qué se tiene que introducir esta regla?

Porque si no podrían conectarse al servidor HTTP sin pasar por el proxy.

¿Importa la posición en la que se introduzca esta regla? Razona la respuesta.

Sí, debe escribirse antes que la primera regla porque si no se aceptarían antes de llegar a esta.

Escribe la regla

```
sudo iptables -A FORWARD -i enp0s8 -p tcp --dport http -m state --state NEW -j DROP
```

5. Con la configuración que acabamos de hacer en la cadena FORWARD, se permiten cualquier tipo de conexión desde cualquier máquina de Red 1 con el exterior y se descartan los paquetes recibidos por la interfaz interna con puerto destino HTTP. **Solo se debe permitir enviar estos paquetes si lo hacen a través del proxy.** Para ello hay que añadir una serie de reglas en el firewall **de Router 1,** en las cadenas INPUT y OUTPUT, para que se permitan:
   - Las conexiones HTTP, iniciadas desde Red 1 y dirigidas al **proxy**
   - Las conexiones HTTP, enviadas por el **proxy** al servidor

Cadena INPUT:

Introduce las siguientes reglas:

- Aceptar recibir paquetes por la interfaz interna (`enp0s8`) con IP origen Red 1 con puerto destino 8888 (donde escucha el proxy).

```
sudo iptables -A INPUT -i enp0s8 -s 192.168.1.0 --dport 8888 -j ACCEPT
```

¿Por qué especificas que la IP origen sea de Red 1?

Porque solo nos interesa que se puedan conectar hosts de esta red, no de otras.

- Aceptar recibir paquetes de conexiones establecidas.

```
sudo iptables -A INPUT -m state --state ESTABLISHED -j ACCEPT
```

¿Se necesita añadir una regla acepte los mensajes de inicio de conexión (estado NEW) que envía host 1?

No, porque ya se aceptan por la norma indicada justo antes, donde se aceptan los input por el puerto 8888 con origen en la red 1.

Cadena OUTPUT:

Introduce las siguientes reglas:

- Aceptar enviar paquetes por la interfaz externa (`enp0s9`) con puerto destino HTTP.

```
sudo iptables -A OUTPUT -o enp0s9 -p tcp --dport 80 -j ACCEPT
```

- Aceptar enviar paquetes de conexiones establecidas.

```
sudo iptables -A OUTPUT -m state --state ESTABLISHED -j ACCEPT
```

Para especificar conexiones establecidas en una regla de iptables utiliza la opción:
-m state - - state ESTABLISHED

6. Comprueba que desde Host 1 no puedes conectarte al Servidor HTTP (con el navegador o ejecutando el comando `$ wget 192.168.2.20`)

¿Por qué no puede conectarse el Host 1 al Servidor HTTP, aunque Tinyproxy esté instalado y el tráfico dirigido al puerto 8888 de Router 1 permitido?

Porque se está impidiendo el acceso por el puerto 80.

¿Puedes realizar un ping desde Host1 a servidor?

No, debe realizarse la conexión a través del proxy.

¿Puedes realizar un ping desde servidor a Host1?

No, por la misma razón.

Proxy configurado en cliente

Para que todas las conexiones web de Red 1 con el exterior pasen a través del proxy, es necesario redirigir el tráfico HTTP de las máquinas de Red 1 al proxy, y para ello una opción es configurar cada uno de los clientes web de las máquinas de Red 1 para que redirijan este tráfico al proxy.

Ejercicio 3: Acceso al servidor web configurando el cliente web de Host 1.

Para entender cómo se realiza la comunicación entre Host 1 y el servidor utilizando un proxy, primero vamos a analizar los paquetes TCP que se intercambian a través del Router 1 antes de configurar en el cliente (Host 1) el proxy:

1. Arranca Wireshark en Router 1. Configúralo para que escuche en la interfaz `enp0s8` y filtra los paquetes para ver únicamente los de tipo TCP.
2. Intenta conectarte con el servidor desde Host 1 (con el navegador o ejecutando el comando `$ wget 192.168.2.20`) y observa que al Router 1 solo llegan los mensajes SYN de Host 1. Captura ahora por la interfaz `enp0s9`, vuelve a intentar conectarte al servidor desde Host 1 y comprueba que no se ve ningún mensaje, ¿por qué?

Porque el proxy capa las solicitudes de conexión que se hagan por el puerto 80.

Ahora vamos a **configurar en el host que los mensajes dirigidos el puerto http se envíen al proxy** (puerto 8888):

3. Abre el navegador en Host 1 y **configúralo para que todos los mensajes HTTP se envíen al Router 1 a través del puerto 8888, donde escucha el proxy**. Para ello en el navegador hay que pulsar en *Preferences* (ver Figura 2) y después en la pestaña *Settings* de *Network Settings* (al final de Preferences). Se desplegará la ventana de *Connection Settings*, en ella elegir *Manual proxy configuration* y en *HTTP proxy* escribir 192.168.1.10 y en *Port* 8888.

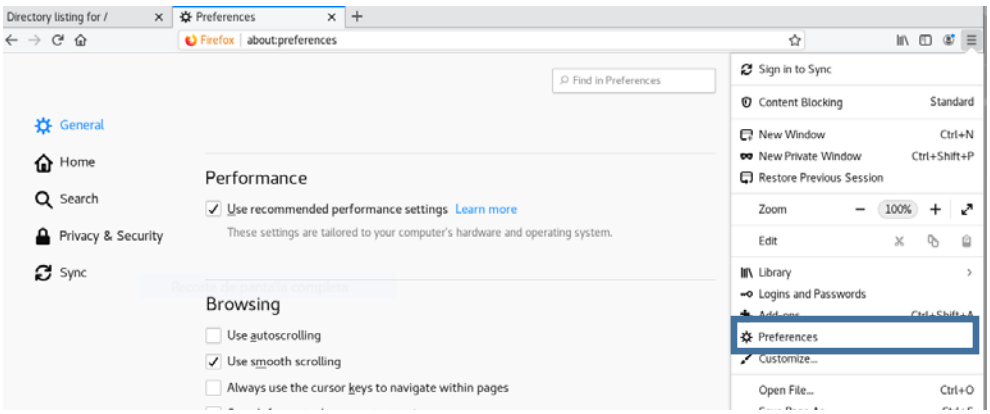

Figura 2

La conexión web también se puede hacer con el comando wget desde el Terminal y sin usar el navegador ni configurarlo. Si lo prefieres hacer así, hay que especificar que use el porxy deberás ejecutar:

```
$ wget 192.168.2.20 -e use_proxy=yes -e http_proxy=192.168.1.10:8888
```

4. Reinicia la captura de Wireshark (Capture → Restart) y vuelve a capturar el tráfico por la interfaz `enp0s8`.

5. Comprueba que ahora sí puedes acceder al servidor desde Host 1 (con el navegador o con el comando `wget`) y observa la captura de Wireshark. **Identifica los 3 primeros mensajes de establecimiento de conexión TCP** y contesta las siguientes preguntas:

¿Qué IP origen aparece?

¿Qué IP destino?

¿Qué puerto destino?

¿Por qué no aparece la IP 192.168.2.20? y sin embargo sí que estás conectado con el servidor?

6. **Busca el primer mensaje en que aparece la IP del Servidor HTTP** y contesta las siguientes preguntas:

¿Qué tipo de mensaje es?:

¿Qué IP y puerto destino aparece en la cabecera IP?

7. Comprueba el fichero de registro de Tinyproxy (en Router 1):

```
$ sudo tail -f /var/log/tinyproxy/tinyproxy.log
```

8. Cierra la conexión desde Host 1, reinicia la captura de Wireshark para que ahora escuche en la interfaz `enp0s9`, vuelve a conectarte al servidor desde Host 1 (con el navegador o con el comando `wget`) y observa el tráfico que sale de **Router 1 hacia el Servidor HTTP**.

¿Qué IP origen y destino aparecen en los mensajes?

¿Qué puertos?

¿A qué máquina contesta el servidor?

Observa que ahora en el mensaje que contiene GET en el campo datos ya no aparece la dirección del Servidor HTTP.

Proxy transparente

Para no tener que configurar los clientes de las máquinas de la Red 1, se puede usar un proxy transparente. En este caso, el navegador de Host 1 creerá que se está comunicando directamente con el servidor, pero todos los mensajes con destino al servidor el firewall los redireccionará al puerto 8888 de Router 1, por lo que pasarán a través del proxy.

Ejercicio 4: Acceso al servidor web a través de un proxy transparente.

Para configurar el proxy de forma que sea transparente al cliente hay que introducir en las iptables una nueva regla que redirija las peticiones web al proxy. Esta regla se tiene que añadir en la tabla NAT de Router 1. los siguientes pasos:

1. Añade en Router 1 esta regla que redirige las peticiones web al proxy:

```
$ sudo iptables -t nat -A PREROUTING -i enp0s8 -p tcp --dport 80 -j REDIRECT
  --to-port 8888
```

2. Apaga el proxy (recuerda que lo hemos arrancado en algún punto de la práctica) para comprobar que la conexión se está realizando a través del proxy:

```
$ sudo /etc/init.d/tinyproxy stop
```

3. Vuelve a **configurar el navegador en Host 1 para que no use proxy** (sigue las instrucciones indicadas en el paso 3 del ejercicio 3 pero ahora elige la opción *No proxy*). A continuación, intenta volver a conectarte al Servidor HTTP. ¿Puedes conectarte ahora? ¿Por qué? (aparte de lo obvio, que es tener el proxy apagado)

3. Configura de nuevo el Wireshark en Router 1 para capturar tráfico por la interfaz `enp0s8.`
4. Arranca de nuevo el proxy:

```
$ sudo /etc/init.d/tinyproxy restart
```

5. Desde Host 1 vuelve a intentar conectarte al Servidor HTTP (con el navegador o ejecutando el comando `$ wget 192.168.2.20`). En el Wireshark de Router 1, observa el tráfico en `enp0s8` correspondiente al proxy transparente. ¿Qué diferencias observas con respecto a las IPs y puertos que aparecían cuando el proxy estaba configurado en el cliente? ¿Por qué esta diferencia?

6. Configura el Wireshark para que capture el tráfico por la interfaz `enp0s9`. Desde Host 1 vuelve a conectarte al servidor (con el navegador o ejecutando el comando `$ wget 192.168.2.20`) y observa el tráfico capturado correspondiente al proxy transparente. ¿Hay diferencias con respecto a la configuración del proxy en el cliente? ¿Por qué?

Seguridad del firewall de inspección de estado y del proxy directo

Vamos a comprobar que tanto el firewall de inspección de estado como el proxy están bien configurados. Tal y como hemos definido las reglas de Iptables, hemos conseguido que ninguna máquina del exterior pueda iniciar una comunicación con ninguna máquina de la Red 1. Para comprobarlo vamos a intentar atacar la Red 1 desde el Servidor HTTP. Para poder atacar la Red 1 deberíamos conocer direcciones IPs de esa red y puertos abiertos (primera fase de un ataque).

Ejercicio 5: Comprobación de la seguridad que ofrece el firewall-proxy a la Red 1.

Prueba a realizar cualquiera de los escaneos de la práctica 1.1 (había algunos directamente asociados al protocolo tcp). Si has configurado bien el firewall no debería funcionar ninguno.

NOTA: el escaneo tiene que ser desde el servidor al Host 1 porque el proxy está protegiendo la red 1

Apunta los escaneos que has probado e indica si han funcionado:

ANOTACIONES