Práctica 1.1. Ataques a redes y protección mediante firewalls I

Redes y Seguridad II

Práctica 1.1 - Ataques a redes y protección mediante firewalls I

Objetivos

Esta práctica está pensada para que el estudiante adquiera habilidades básicas sobre firewalls y comprenda algunas técnicas de ataque, como el escaneo de redes y puertos.

Índice

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
	Programador/a
	Analista

La misión principal del *Programador* es escribir todos los comandos indicados en la práctica en el orden correcto para conseguir un buen aprovechamiento de los objetivos buscados en la misma.

La misión principal del *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, es el encargado de contestar el test asociado a la práctica.

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 2 de febrero de 2020

Configuración de la red

En esta práctica vamos a implementar, con máquinas virtuales, la estructura de la red que se puede observar en la Figura 1. Para ello, vamos a realizar **2 clonaciones** de la máquina virtual **rys22.ova** (usuario:usuario/contraseña:usuariorys): Router 1 y Host 2 (recuerda **reinicializar las direcciones MAC** y realizar **clonación enlazada**) y **una clonación** de la máquina **consola_rys22.ova** (usuario:usuario/contraseña:usuariorys): Host 1.

Si en el entorno donde estás realizando la práctica **ya existen las máquinas** Host 1 y Host 2, no necesitarás hacer la clonación, sino que **puedes reutilizarlas**.

No queremos que las máquinas **Host 1, Host 2, Router 1 y Servidor HTTP** se puedan conectar con el exterior, por lo tanto, será necesario **deshabilitar el adaptador de red**. Ir a VirtualBox y sobre cada máquina pulsar con el botón derecho. Elegir la opción Configuración->Red; en Adaptador 1, pulsar sobre Avanzadas y desmarcar la opción Cable conectado.

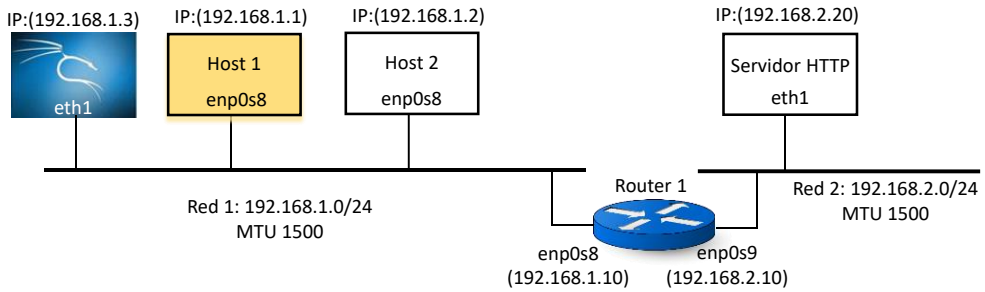


Figura 1

Importa la máquina virtual **Kali** (usuario:kali/contraseña:kali) y la máquina **MetasploitLinux** (usuario:msfadmin/contraseña:msfadmin), que será nuestro Servidor HTTP. Comprueba que ambas máquinas tienen el adaptador 2 conectado a una red interna, la máquina Kali a Red 1 y la máquina Servidor HTTP a Red 2.

Recuerda que las máquinas virtuales las puedes encontrar en la pestaña Material Prácticas del CV o en el laboratorio en las carpetas RS2 (Equipo/mnt/DiscoVMs/RS2). En el laboratorio, la máquina MetasploitLinux se encuentra en la carpeta SER (Equipo/mnt/DiscoVMs/SER).

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

Recuerda que, si has decidido **reutilizar las máquinas Host 1 y Host 2,** para configurarlas comenzarías directamente en el **paso 4**.

1. Comprueba que las máquinas tienen 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 las máquinas y configura sus interfaces de red. En la máquina **Kali** y en el **Servidor HTTP** el **adaptador de red 2** se llama **eth1**, en lugar de enp0s8. 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 de texto (usando 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 restart
```

4. Configura la tabla de encaminamiento de Host 1:

```
$ sudo ip route add 192.168.2.0/24 via 192.168.1.10
```

5. Haz en Kali $ man arpspoof para comprobar si está ese software instalado. Si lo está salta al paso 6, si no, tendrás que descargar en la máquina de Kali el paquete `dsniff` (este paquete contiene el comando `arpspoof`)

```
$ sudo apt-get update
$ sudo apt-get install dsniff
```

6. Repite los mismos pasos con Host 2, Kali (la interfaz es eth1), Servidor HTTP (la interfaz es eth1) y Router 1 (para esta máquina no hay que definir ningún router, ya que pertenece a las dos redes). No te olvides de activar el *forwarding* en Router 1:

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

> **Nota:** El servidor y Kali tienen el teclado americano. Podéis **cambiar el teclado a español** con:
> `$ sudo loadkeys es`

7. En el caso de la máquina Kali, desactiva el adaptador conectado a NAT, para ello ir a VirtualBox y sobre la máquina de Kali pulsar con el botón derecho. Elegir la opción Configuración->Red; en Adaptador 1, pulsar sobre Avanzadas y desmarcar la opción Cable conectado.

8. Comprueba que todas las máquinas están correctamente conectadas, haciendo, por ejemplo, un ping de Host1 a Host2 y un ping de Host 1 a Servidor HTTP.

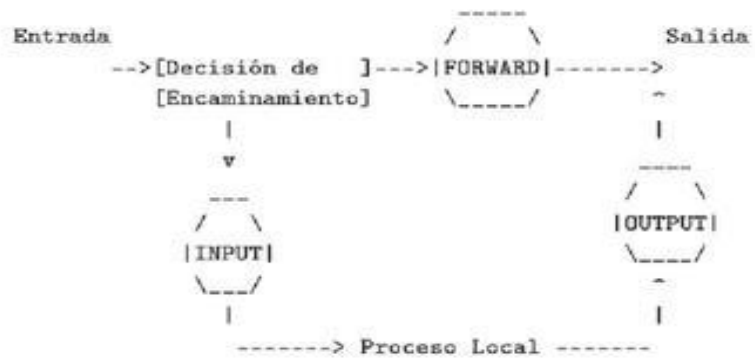
Iptables: algunas reglas básicas

Iptables es una herramienta, integrada en el kernel de Linux, con la que se puede implementar un firewall de filtrado de paquetes, que permitirá restringir o filtrar el tráfico que entra o sale de un sistema. Además, si se usan los módulos *state* o *conntrack*, iptables puede funcionar como un firewall de inspección de estado, que guarda información sobre el estado de una conexión y permite saber si un paquete pertenece o no a una conexión activa. Iptables permite crear reglas que analizarán los paquetes de datos que entran, salen o pasan por nuestra máquina, y en función de las condiciones establecidas, tomará la decisión de qué hacer con el paquete.

Iptables ofrece varias tablas. En esta práctica trabajaremos con la tabla FILTER, para el filtrado de paquetes. No obstante, más adelante utilizaremos también la tabla NAT.

La tabla FILTER de Iptables tiene definidas tres CADENAS:

a. `INPUT`, para paquetes entrantes cuyo destino es la propia máquina
b. `OUTPUT`, para paquetes salientes cuyo origen es la propia máquina
c. `FORWARD`, para paquetes entrantes pero cuyo destino es otra máquina y pasan a través de esta



Un firewall de **filtrado de paquetes** se configura definiendo una serie de reglas que establecen unas acciones a realizar sobre los paquetes que analizan. Estas reglas se construyen a partir de condiciones sobre los campos de las cabeceras IP, TCP, UDP ..., por lo que al analizar los paquetes solo se examinarán estas cabeceras.

Las principales acciones son:

a. `DROP`: descartar (borrar el paquete, como si nunca hubiera llegado)
b. `ACCEPT`: dejar pasar el paquete
c. `REJECT`: rechazar el paquete enviando un mensaje de error. Se utiliza si se quiere poner de manifiesto que existe un firewall protegiendo la conexión.

Generalmente, los firewalls se configuran para trabajar con una **política de DROP por defecto**. Esto quiere decir que, si un paquete cumple alguna de las reglas, se le dejará pasar. Y si no, se descartará. Las reglas se analizan en orden y si una se cumple ya no se siguen comprobando las demás.

Opciones básicas:

`-P` : establecer política por defecto
`-A` : añade una regla a una cadena de forma temporal (cuando reinicies la máquina, la regla se habrá perdido)
`-D` : borra una regla
`-j` : indica la acción a realizar
`-i` : especifica la interfaz de entrada
`-o` : especifica la interfaz de salida
`-p` : especifica el protocolo
`-s` : especifica la IP fuente
`-d` : especifica la IP destino
`--dport` : especifica el puerto destino (tiene que ir siempre precedido de –p TCP o –p UDP)

Ejercicio 1: Ejemplo de construcción de una regla con iptables.

¿Qué dice la siguiente regla de iptables?

```
iptables -A INPUT -i enp0s8 -p tcp --dport 80 -s 192.168.1.1 -j ACCEPT
```

Configurando Iptables

Ejercicio 2: Configuración de iptables en Host 2.

Primero, vamos a consultar el contenido de la tabla FILTER de iptables en Host 2:

```
$ sudo iptables –L -v
```

¿Qué cadenas hay? ¿Cuántas reglas hay en el firewall? ¿Qué política por defecto tiene definida el firewall?

Añade en Host 2 una regla con `iptables -A`:

```
$ sudo iptables -A INPUT -s 192.168.1.1 -p ICMP -j DROP
```

¿Qué dice esta regla?

Comprueba que la regla se ha añadido correctamente (`sudo iptables –L –v`, puede tardar un poco en ejecutarse el comando) y realiza un ping desde Host 1 a Host 2. ¿Qué sucede? ¿Por qué?

Las reglas introducidas con el comando iptables se perderán al reiniciar la máquina virtual. Sin embargo, también se puede borrar una regla sin reiniciar la máquina usando la opción –D.

Borra la regla anterior (en Host 2) y comprueba que se pueden volver a recibir paquetes ICMP desde Host 1:

```
$ sudo iptables -D INPUT -s 192.168.1.1 -p ICMP -j DROP
```

Ejercicio 3: Configuración de iptables en Router 1.

Primero, vamos a comprobar que nos podemos conectar al Servidor HTTP (conexión web) desde Host 1, para ello desde la consola de Host 1 **ejecutar** `wget 192.168.2.20`.

Si no podéis conectaros puede que se haya desconfigurado alguna interfaz de red en alguna de las máquinas o que no estén bien las tablas de encaminamiento, corregid el problema.

Si la conexión con el servidor funciona, añadir una **regla en el firewall de Router 1 que impida el *forwarding* de paquetes TCP entrantes desde la Red 1**:

```
$ sudo iptables -A FORWARD -i enp0s8 -p TCP -j DROP
```

Comprueba que las **máquinas de Red 1** ya **no se pueden conectar con el Servidor HTTP con wget, pero en cambio sí pueden realizar un ping**. ¿Por qué?

Sin embargo, la configuración normal no sería esta. Desde el punto de vista de la seguridad lo normal es que el usuario pueda acceder a la web, pero no atacar el servidor (mediante ping flood, por ejemplo). ¿Qué tendría que cambiar en la regla anterior para que sea así?

Para vaciar cadenas se usa la opción -F:

- `$ sudo iptables –F OUTPUT,` vacía la cadena OUTPUT
- `$ sudo iptables –F,` para vaciar todas las cadenas de Router 1 de una vez.

Vacía las cadenas de Router 1 y comprueba ahora, otra vez, que ya puedes conectarte con wget al Servidor HTTP.

Escaneo de redes con Nmap

El escaneo de redes es una parte de la auditoría de seguridad que sirve para obtener información del sistema (máquinas activas, puertos abiertos, sistemas operativos, servicios activos …). También forma parte de la fase de reconocimiento de un ataque a una red, cuyo objetivo es recopilar toda la información que se pueda sobre dicha red.

El objetivo de todos estos sondeos es enviar mensajes y sacar conclusiones en función de la respuesta recibida, que depende del estado de la máquina/puertos, de la especificación del protocolo y de su implementación. Esto último es muy importante porque el mismo mensaje puede obtener resultados diferentes en diferentes SO, o lo que es lo mismo, algunos SO pueden ser vulnerables a ciertos escaneos y otros no.

Los motivos para realizar pruebas de escaneo de redes son muy diversos y, dependiendo de ellos, será necesario usar distintos protocolos en diferentes capas de la pila TCP/IP, aunque lo más habitual es trabajar en las capas 3 (protocolos IP e ICMP) y 4 (protocolos TCP y UDP).

Nmap es una herramienta de exploración de redes que ofrece innumerables opciones, que pueden adaptarse a las necesidades de una auditoría. Permite identificar cuáles son las máquinas que hay en

una red, qué puertos tienen abiertos, qué sistema operativo o servicios (incluidas versiones) tienen instalados estas máquinas, si hay algún tipo de firewall instalado en la máquina, … Los sistemas objetivo se pueden especificar con una secuencia de nombres de hosts, una secuencia de direcciones IP, una lista (192.168.1.1,2), un rango (192.168.1.1-2), una dirección de red (192.168.1.0/24) o comodines (192.168.1.*). También se pueden limitar los puertos a explorar con la opción -p, seguida de una lista o rango de puertos.

Nmap comenzó como un analizador de puertos eficiente, y aunque ha aumentado su funcionalidad a través de los años, esta sigue siendo su función principal. La sencilla orden **`nmap <objetivo>`** analiza más de 1660 puertos TCP del equipo `<objetivo>`. Aunque muchos analizadores de puertos han agrupado tradicionalmente los puertos en dos estados: abierto o cerrado, Nmap es mucho más descriptivo. Los puertos se dividen en seis estados distintos: abierto, cerrado, filtrado, no filtrado, abierto|filtrado, o cerrado|filtrado.

Consulta el manual de nmap ejecutando el comando `man nmap` en un Terminal o consultando la página https://nmap.org/. Apunta las opciones que consideres más interesantes.

En esta práctica, vamos a probar sobre la arquitectura de red de la Figura 1 distintas posibilidades de uso de Nmap.

Lo primero que vamos a hacer es un reconocimiento de hosts o *Host Discovery* dentro de una red. De la lista de posibles IPs de una red, queremos saber cuáles están activas o cuáles pueden ser interesantes (por ejemplo, si pertenecen a servidores, etc.). Para saber si un host está activo y correctamente conectado, a una administradora de sistemas le podría bastar con utilizar un ping. Sin embargo, una auditora, que está realizando una prueba de penetración (pen test), puede estar más interesada en comprobar si es posible evadir un firewall.

Al descubrimiento de hosts a veces también se le llama *ping scan*, pero va mucho más allá del envío de paquetes ICMP Echo Request, ya que también pueden utilizarse diferentes combinaciones de mensajes TCP SYN/ACK, UDP, SCTP INIT y distintos sondeos con ICMP.

En la mayoría de las redes solamente un pequeño porcentaje de las IPs están activas simultáneamente.

== TIPS ==

Uno de los primeros pasos en cualquier misión de reconocimiento de red es el de reducir un conjunto (muchas veces enorme) de posibles direcciones IP a una lista de equipos activos o interesantes. Analizar cada puerto de cada una de las direcciones IP es lento y normalmente innecesario.

Una de las opciones más utilizada es `-sP` (Escaner ping) o `-sn` (en versiones más modernas de Nmap). La opción `-sP` le indica a Nmap que únicamente realice descubrimiento de sistemas mediante un escaneo ping, y que luego emita un listado de los equipos que respondieron al mismo. No se realizan más escaneos (como un análisis de puertos o detección del sistema operativo). Permite un

reconocimiento poco intrusivo de la red objetivo. Saber cuántos equipos están activos es de mayor valor que el listado de cada una de las IP y nombres proporcionado por el escaneo.

La opción `-sn` envía una solicitud ICMP Echo Request y un paquete TCP al puerto 80 (http) y al puerto 443 (https) por omisión.

Cuando un usuario sin privilegios ejecuta Nmap se envía un paquete SYN (utilizando la llamada connect()) al puerto 80 del objetivo.

Cuando un usuario privilegiado intenta analizar objetivos en la red Ethernet local se utilizan solicitudes ARP (-PR) a no ser que se especifique la opción --send-ip.

Escaneo de redes

Ejercicio 4: Escaneo de Red 1 y Red 2 desde Kali

Abre el Wireshark en Kali y configúralo para que muestre los paquetes capturados por la interfaz `eth1`.

Escanea desde Kali la Red 1 (192.168.1.0) para ver qué hosts están activos utilizando:

```
$ sudo nmap -sn 192.168.1.0/24
```

¿Se ha podido realizar el escaneo? ¿Qué información has obtenido? ¿Coincide con los hosts que tienes conectados a la red?

Observa en el Wireshark los paquetes capturados (desde la ejecución del comando nmap), ¿qué tipo de mensajes se han usado? ¿y cuántos mensajes? En base a la información obtenida explica exactamente qué es lo que hace Kali para conseguir la información buscada.

Haz un Capture → Restart en el Wireshark. Escanea ahora la Red 2 (192.168.2.0) con la opción `-sn`.

```
$ sudo nmap -sn 192.168.2.0/24
```

¿Qué información has obtenido? ¿Qué información has obtenido de los hosts en la Red 1 que no obtienes en este último escaneo? ¿Por qué?

Observando los paquetes capturados en Wireshark (desde la ejecución del último comando nmap), ¿qué tipo de mensajes se han usado para detectar los hosts activos en esta red?

La mayoría de los tipos de escaneo disponibles solo los puede llevar a cabo un usuario privilegiado. Esto es debido a que envían y reciben paquetes en crudo, lo que hace necesario tener acceso como administrador (root).

Escaneo de puertos

A continuación, vamos a hacer un escaneo de puertos. Los distintos sondeos consisten en enviar distintos mensajes TCP (activando distintas marcas o flags) y observar los mensajes que devuelve el sistema objetivo (o los firewalls que están delante de ellos). Por ejemplo:

- Escaneo Null (`-sN`): No fija ningún bit (la cabecera de flags TCP es `0`)
- Escaneo FIN (`-sF`): Solo fija el flag TCP FIN.
- Escaneo Xmas (`-sX`): fija los bits de FIN, PSH, y URG flags, iluminando el paquete como si fuera un árbol de Navidad.

Estos tres tipos de escaneos aprovechan una indefinición en la RFC de TCP que diferencia los puertos abiertos de los cerrados. En este documento se dice que si el puerto destino está CERRADO se enviará un segmento con el flag de RST activo en respuesta a un segmento entrante que no contenga un RST. Cuando se escanean sistemas que cumplen con el texto de esta RFC, cualquier paquete (sin los bits SYN, RST, o ACK activos) generará el envío de un segmento de RST si el puerto está cerrado y no enviará nada si el puerto está abierto. Siempre y cuando no se incluyan ninguno de estos tres bits, cualquier combinación de los otros tres (FIN, PSH, y URG) será válida para conocer el estado del puerto.

Ejercicio 5: Escaneo de puertos del Servidor HTTP desde Kali

Desde Kali, obtén información de los puertos del Servidor HTTP con el **escaneo de puertos FIN** (activa el flag FIN en un mensaje TCP). ¿Qué información has obtenido? ¿En qué estado está el puerto 80?

```
$ sudo nmap -sF 192.168.2.20
```

== TIPS ==

La ventaja fundamental de este tipo de escaneos es que pueden atravesar algunos firewalls que no hagan inspección de estado o encaminadores que hagan filtrado de paquetes. Otra ventaja es que este tipo de escaneos son algo más sigilosos que, incluso, un escaneo SYN. Sin embargo, la mayoría de los productos IDS pueden configurarse para detectarlos. El problema es que no todos los sistemas siguen el estándar RFC 793 al pie de la letra. Algunos sistemas envían respuestas RST independientemente de si el puerto está o no cerrado. Esto hace que la mayoría de los puertos se marquen como cerrados. Algunos sistemas operativos muy utilizados que hacen esto son Microsoft Windows, muchos dispositivos Cisco, BSDI, e IBM OS/400. Este escaneo no funciona contra sistemas basados en UNIX. Otro problema de estos escaneos es que no se pueden distinguir los puertos *abiertos* de algunos puertos *filtrados*, lo que resulta en la respuesta *abierto|filtrado*.

Repite el escaneo haciendo ahora un escaneo SYN. ¿Qué información has obtenido? ¿En qué estado está el puerto 80? Explica, en base a lo explicado en la práctica y a lo explicado en las clases de teoría, a qué se debe esta diferencia en el estado del puerto 80 en los dos escaneos.

Configuración de Iptables para evitar escaneos con Nmap

A continuación, vamos a pensar cómo impedir estos escaneos mediante Iptables. Entre los comandos que podemos utilizar existen algunos específicos para el protocolo TCP. Para obtener ayuda sobre una extensión, escribe -h o --help después de la opción que se usa para cargarla, por ejemplo:

```
$ sudo iptables -p tcp --help
```

Las extensiones TCP se cargan de forma automática y se pueden utilizar las siguientes opciones (siempre precedidas de la opción `-p tcp`):

`--tcp-flags,` seguido por dos cadenas de indicadores (flags), permite filtrar dependiendo de ciertos indicadores de TCP. La primera cadena es la máscara, es decir, una lista de los indicadores que se desea examinar. La segunda cadena especifica los indicadores que deben estar activos. Por ejemplo:

```
iptables -A INPUT --protocol tcp --tcp-flags ALL SYN,ACK -j DROP
```

indica que deben ser examinados todos los indicadores («ALL»), pero solo deben estar activos SYN y ACK para que el paquete se descarte. Hay otro argumento llamado «NONE», que significa "ningún indicador".

`--syn,` es equivalente a `--tcp-flags SYN,RST,ACK SYN`

`--source-port` (o **`-sport`**), seguido por un puerto o rango de puertos TCP. Estos pueden estar representados por su nombre, tal como viene en /etc/services, o por un número. El rango se puede representar por dos nombres de puerto separados por un -, o por dos números siguiendo el formato primer_puerto:último_puerto.

`--destination-port` (o **`-dport`**), es lo mismo que lo anterior, solo que especifican el puerto de destino en lugar del de origen.

`--tcp-option,` seguido por un número, se ajusta a paquetes con una opción TCP igual a ese número. Un paquete que no tenga una cabecera TCP completa, será descartado automáticamente si se intentan examinar sus opciones TCP.

Ejercicio 6: Teniendo en cuenta las reglas anteriormente descritas, **¿Cómo podrías evitar que un escaneo -sF proveniente del exterior consiga atravesar el Router 1 y entrar en la Red 2?** Implementa tu solución y comprueba su funcionamiento repitiendo el escaneo y comparando la información con la obtenida la primera vez que lo realizaste.

Si la máquina de Kali perteneciese a la Red 2, ¿funcionaría el escaneo de los puertos del Servidor HTTP con el **escaneo de puertos FIN** después de introducir esta regla en el firewall?

Si la respuesta es afirmativa, ¿qué regla de iptables habría que introducir y en qué máquina habría que hacerlo para evitar que un atacante interno escaneara los puertos del Servidor HTTP? ¿Valdría esta solución también para evitar el escaneo proveniente del exterior? Comprobar que la nueva regla funciona correctamente.

== TIPS ==

La cantidad de posibles escaneos o ataques es innumerable, y por tanto también la cantidad de reglas que habría que escribir para mitigarlos.

Por eso se aconseja que el firewall tenga como **política por defecto la de descartar paquetes** y se añadan reglas para las comunicaciones permitidas, como, por ejemplo, realizar consultas a un servidor DNS, conectarse a un servidor http externo …

Ejercicio 7: Por último, vamos a intentar **detectar** desde Kali qué **sistema operativo** tiene el Servidor HTTP. Borra las reglas de iptables que hayas introducido anteriormente en el servidor y en Router 1 y después ejecuta:

```
$ sudo nmap -O 192.168.2.20
```

Ante ciertas solicitudes cada sistema operativo responde de una determinada forma. Nmap posee una base de datos con la información que devolverá cada sistema operativo en el fichero /usr/share/nmap/nmap-os-db. Cuando se ejecuta nmap con la opción –O se manda una serie de paquetes a la máquina objetivo y su respuesta permite crear una huella (*fingerprint*) del sistema que se compara con las contenidas en la base de datos. Si hay alguna coincidencia se ha podido detectar el sistema operativo. Más información en http://nmap.org/book/osdetect.html.

Copia la salida obtenida al tratar de detectar el sistema operativo del Servidor HTTP.

Ejecuta en el Servidor HTTP el comando `$ sudo uname –a`, para ver si la versión del kernel detectada coincide.

ARP Cache poisoning con ARP Spoofing

Vamos a simular un ataque desde el interior de la red, donde la máquina atacante (Kali) se va a hacer pasar por el router que hace de gateway de la red, capturando el tráfico de Host 2 que entra y sale de Red 1.

Para ello, en primer lugar, la máquina atacante va a realizar un envenenamiento de la tabla ARP (ARP-cache poisoining) de Host 2: realizando un ARP spoofing va a anunciar a Host 2 que su dirección MAC está asociada a la IP de Router 1. De esta forma todos los paquetes dirigidos de Host 2 a Router 1 pasarán por Kali y podrá espiar la comunicación desde el Host 2 al exterior. Después, la máquina de Kali también va a realizar un envenenamiento de la tabla ARP del Router 1 anunciándole que su dirección MAC está asociada a la IP de Host 2. De esta manera la máquina de Kali actúa como un man-in-the-middle

(Intermediario) en la comunicación entre Host 2 y Router 1 pudiendo, escuchar todo el tráfico intercambiado entre estas dos máquinas.

Para este apartado de la práctica se puede apagar el Host 1

La siguiente imagen es para recordar la estructura de red que estamos utilizando e ilustra lo que queremos reproducir.

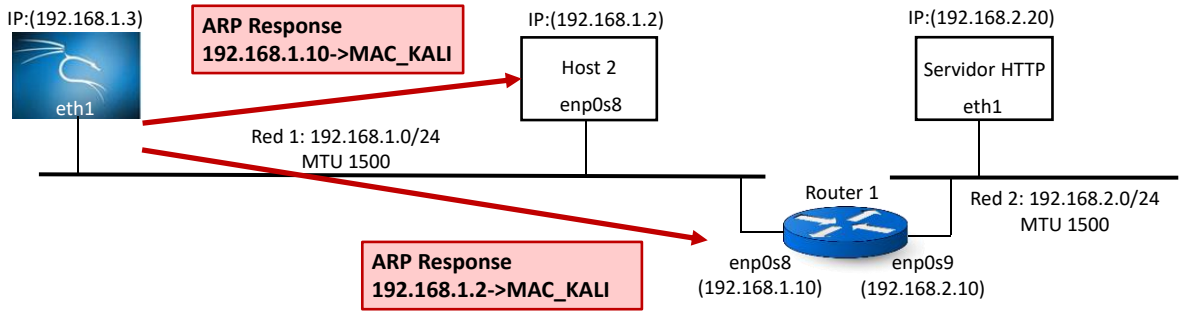


Figura 2

Ejercicio 8: Ataque ARP cache poisoning

Usando el comando `ip address`, averigua y apunta las direcciones MAC de Router 1, Host 2 y la máquina Kali.

```
Router 1-enp0s8:

Host 2-enp0s8:

Kali -eth1:
```

Para poder realizar y comprender este ataque vamos a seguir los siguientes pasos:

1. Abre el Wireshark en la máquina de Kali.
2. Comprueba el contenido de la tabla ARP de Host 2 (`$ ip neigh show`). Si hay alguna entrada, bórrala con `$ sudo ip neigh del <IP_maquina> dev <Interfaz>.`
3. Realiza un ping de Host 2 al Servidor HTTP que envíe 8 mensajes con el contenido "ABA":

   ```
   $ ping -c 8 -p ABA 192.168.2.20
   ```

4. Comprueba el contenido de la tabla ARP de Host 2:

   ```
   $ ip neigh show
   ```

   La dirección MAC asociada a la IP de Router 1 debería ser la correcta.

5. Comprueba en el Wireshark que la máquina Kali solamente ha recibido el ARP *Request (broadcast)* preguntando por la dirección MAC de Router 1 y no puede escuchar ni la respuesta ni el contenido del ping de Host 2 al Servidor HTTP.
6. Borra la entrada de la tabla ARP de Host 2 que se ha creado.

7. Ejecuta la siguiente orden en la máquina de Kali:

   ```
   $ sudo arpspoof -i eth1 -t 192.168.1.2 -r 192.168.1.10
   ```

   Para que el ataque funcione el atacante debe **mantener esta orden en ejecución** para que no se borre la entrada correspondiente de las tablas ARP de la máquina o máquinas víctima.

   Comprobar el manual de arpspoof (`man arpspoof`) para entender las distintas opciones del comando.

-i:

-t:

-r:

¿Cuál de las IPs del comando se considera de la máquina víctima?

¿Cuál de las IPs del comando pertenece a la máquina que debería ser la destinataria de los paquetes?

¿Qué mensajes está enviando esta orden?

Ahora repite el ping desde Host 2 al Servidor HTTP. ¿Qué tráfico capturas con el Wireshark en la máquina de Kali?

¿Por qué sí puedes ver ahora los mensajes entre Host 2 y el Servidor HTTP?

¿Funciona el ping? Si la respuesta es negativa, explicad a qué puede ser debido.

Vuelve a hacer el ping y comprueba las tablas ARP de Host 2 y Router 1. ¿Con qué MAC se vinculan las IPs de estas máquinas en cada tabla?

BONUS TRACK

Los primeros mensajes ICMP Echo Request e ICMP Echo Reply del PING llegan a destino, pero después dejan de hacerlo. ¿A qué crees que puede ser debido?

Para poder contestar a esta pregunta puedes abrir el Wireshark en Router 1 (interfaz enp0s8), borrar las tablas ARP de Host 1 y Router 1, lanzar el comando arpspoof y hacer el ping desde Host 1.

7. Detén el comando de arpspoof (con ctrl+C) y activa el reenvío de paquetes en la máquina Kali:

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

8. Vuelve a ejecutar el comando arpspoof:

   ```
   $ sudo arpspoof -i eth1 -t 192.168.1.2 -r 192.168.1.10
   ```

   Repite el ping de Host 2 al Servidor HTTP. ¿Funciona? ¿Qué tráfico capturas con el Wireshark en la máquina de Kali? ¿Por qué?

   ¿Se puede mitigar este tipo de ataques con una regla de Iptables? ¿Por qué?

ANOTACIONES