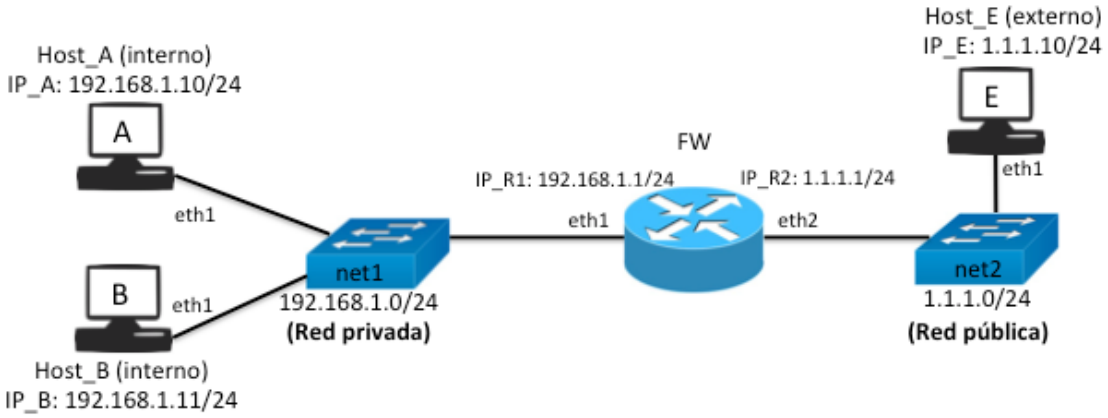
Seguridad en Redes

Práctica 3.3. Cortafuegos

Preparación del entorno

Vamos a usar dos redes internas (`net1` y `net2`) y 4 MVs (`FW`, `Host_A`, `Host_B` y `Host_E`) tal y como se muestra en la figura. La red `net1` emulará una red privada y la `net2` emulará una red pública (Internet):



Importa una MV usando el archivo `SER.ova`, haz tres clonaciones enlazadas y configura los siguientes adaptadores de red:

- Todas las máquinas tendrán el adaptador 1 (`eth0`) conectada a NAT.
- `Host_A` y `Host_B` están conectadas a `net1` a través del adaptador 2 (`eth1`).
- `Host_E` está conectada a `net2` a través del adaptador 2 (`eth1`).
- `FW` está conectado a ambas redes: adaptador 2 (`eth1`) a `int` y adaptador 3 (`eth2`) a `ext`.

Configura `FW`:

```
sudo ifdown eth0
sudo ip link set dev eth1 up
sudo ip link set dev eth2 up
sudo ip addr add 192.168.1.1/24 broadcast + dev eth1
sudo ip addr add 1.1.1.1/24 broadcast + dev eth2
sudo sysctl -w net.ipv4.ip_forward=1
```

Configura `Host_A`:

```
sudo apt-get update
sudo apt-get install ssh hping3
sudo ifdown eth0
sudo ip link set dev eth1 up
sudo ip addr add 192.168.1.10/24 broadcast + dev eth1
sudo ip route add 1.1.1.0/24 via 192.168.1.1
```

Configura `Host_B`:

```
sudo apt-get update
sudo apt-get install ssh hping3
sudo ifdown eth0
sudo ip link set dev eth1 up
sudo ip addr add 192.168.1.11/24 broadcast + dev eth1
sudo ip route add 1.1.1.0/24 via 192.168.1.1
```

Configura `Host_E`:

```
sudo apt-get update
sudo apt-get install ssh hping3
sudo ifdown eth0
sudo ip link set dev eth1 up
sudo ip addr add 1.1.1.10/24 broadcast + dev eth1
sudo ip route add 192.168.1.0/24 via 1.1.1.1
```

Arranca un servidor *web* simple en `Host_A`, `Host_B` y `Host_E`:

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

Filtrado de paquetes

Consulta la página de manual del comando `iptables` o, por ejemplo, https://help.ubuntu.com/community/IptablesHowTo

Establece en el `FW` la política por defecto de las tres cadenas de la tabla `FILTER` (`INPUT`, `OUTPUT`, `FORWARD`) a descartar (`DROP`)

Una vez definidas esta políticas, prueba que no es posible establecer ningún tipo de comunicación (`ping`, `http`, `ssh`, etc.) entre las máquinas internas y la externa, o viceversa.

Permite el acceso al interfaz *loopback* en el `FW`:

```
$ sudo iptables -A INPUT -i lo -j ACCEPT
$ sudo iptables -A OUTPUT -o lo -j ACCEPT
```

A continuación, define en el `FW` las reglas de filtrado que se indican a continuación, en el orden indicado. Todas las reglas se deben añadir a la cadena `FORWARD`.

1. Permitir todos los paquetes pertenecientes a conexiones en estado `ESTABLISHED` o `RELATED`

2. Permitir `ping` de la red interna a la externa.

   Para ello es necesario aceptar los paquetes `ICMP` de tipo `echo request` que entran por la interfaz interfaz red interna (`eth1`). Observa que no es necesario permitir explícitamente las respuestas `echo reply` que entran por la interfaz externa (`eth2`), ya que a éstas se les aplica el estado `RELATED`.

Una vez definida esta regla comprueba que es posible realizar un `ping` desde las máquinas internas (`Host_A` y `Host_B`) hacia la máquina externa (`Host_E`), pero no al revés.

3. Permitir todas las conexiones `ssh` nuevas, tanto entrantes como salientes

   Para ello es necesario aceptar todos los paquetes `tcp` con estado `NEW` dirigidos al puerto destino `22` (`ssh`).

   Una vez definida esta regla comprueba que es posible realizar una conexión `ssh` desde las máquinas internas hacia la máquina externa, y también desde la máquina externa a las máquinas internas.

   Observa en el `FW` la tabla de estado de conexiones en el fichero `/proc/net/nf_conntrack`.

4. Suponiendo que el `Host_A` actúa como servidor web público, permitir conexiones nuevas dirigidas al servidor `http` de dicho host.

   Para ello es necesario aceptar todos los paquetes `tcp` con estado `NEW` dirigidos a la IP destino del `Host_A` (`192.168.1.10`) y al puerto destino `80` (`http`).

   Una vez definida esta regla comprueba que es posible realizar una conexión `http` desde el `Host_E` al `Host_A` (usa en `Host_E` el comando `wget 192.168.1.10`). Comprueba también que no es posible realizar una conexión `http` desde el `Host_E` al `Host_B`.

   Observa en el `FW` la tabla de estado de conexiones en el fichero `/proc/net/nf_conntrack`.

5. Regla *antispoofing* 1: prohibir que entren paquetes desde la red externa con una dirección origen perteneciente a la red interna.

   Para ello es necesario rechazar todos los paquetes que entren por la interfaz externa (`eth2`) y que tengan una IP origen perteneciente a la red interna (`192.168.1.0/24`).

   Intenta establecer una conexión sobre el puerto 80 del `Host_A`, desde el `Host_E`, suplantando la identidad de una máquina interna, ejecutando el siguiente comando en el `Host_E`:

   ```
   # hping3 -p 80 -S --spoof 192.168.1.11 192.168.1.10
   ```

   Problema: si ejecutamos `wireshark` en el `Host_A`, veremos que los paquetes con suplantación de IP siguen entrando

   ¿Por qué sucede esto?: debido al orden de las reglas. Comprueba el orden de las reglas de filtrado con el comando `iptables -L -v` y observa que la regla que permite conexiones nuevas al servidor web del `Host_A` está antes que la regla *antispoofing* y por tanto es la que se aplica.

   Solución: poner las reglas *antispoofing* en primer lugar. Para añadir la regla en una posición concreta, en lugar de añadir con la opción `-A FORWARD`, usamos la opción de

insertar regla en una posición determinada: `-I FORWARD <posicion>`.

Borra la regla *antispoofing* escrita anteriormente (opción `-D FORWARD`) y escribirla de nuevo colocándola en la posición número 3 (opción `-I FORWARD 3`), justo detrás de las reglas que permiten el acceso a la interfaz de *loopback*. Comprueba el orden de las reglas de filtrado con el comando `iptables -L -v`

Una vez establecida la regla antispoofing al principio de la lista de reglas, ejecuta de nuevo el siguiente comando en comando en el `Host_E`:

```
# hping3 -p 80 -S --spoof 192.168.1.11 192.168.1.10
```

Ejecuta `wireshark` en el `Host_A`, y comprueba que ahora los paquetes con la IP suplantada no llegan a esta máquina.

6. Regla *antispoofing* 2: prohibir que salgan paquetes hacia la red externa con una dirección origen que no pertenece a la red interna.

   Para ello se insertará una regla en la posición 4 que rechace los paquetes que entran por la interfaz interna (`eth1`) con una dirección IP origen distinta de `192.168.1.0/24`. Para ello es necesario usar el operador de negación (`!`), con siguiente sintaxis: `! -s 192.168.1.0/24`.

   Una vez definida esta regla comprueba que no es posible enviar un `ping` del `Host_A` al `Host_E`, suplantando la dirección IP de una máquina externa. Para ello se puede usar el siguiente comando en el `Host_A`:

   ```
   # hping3 --icmp -icmptype 8 --spoof 1.1.1.1 1.1.1.10
   ```

7. Reglas para registrar (*log*) las acciones del firewall, aplicado a las reglas antispoofing.

   Para ello, es necesario repetir las dos reglas *antispoofing* anteriores, pero usando la acción `-j LOG`, en lugar de `-j DROP` (se necesita una regla para registrar, y otra, con los mismos criterios para descartar). Las reglas se escribirán siguiendo las siguientes recomendaciones:
   - Se creará un `LOG` de nivel 4 (*Warning*) usando la opción `--log-level warning`
   - Se añadirá el prefijo "*IP Spoofing*" a los mensajes de `LOG` usando la opción `--log-prefix "IP Spoofing"`
   - Para que las reglas `LOG` tengan efecto se deben colocar antes de las reglas `DROP` (colocaremos las reglas `LOG` en las posiciones 3 y 4, respectivamente)

   Una vez establecidas las relas `LOG`, realiza IP spoofing desde el `Host_E` hacia el `Host_A` (utiliza el comando `hping3` usado anteriormente) y comprueba los mensages de `LOG` que se registran en el `FW`. Estos mensages se guardan en el archivo `/var/log/messages`.

   Muchos administradores definen una nueva cadena, `LOGDROP`, con una regla para registrar y otra para descartar todos los paquetes, de forma que sólo tienen que escribir una regla con la acción `-j LOGDROP` para registrar y descartar. La nueva cadena se puede definir mediante las siguientes órdenes:

```
# iptables -N LOGDROP
# iptables -A LOGDROP -j LOG --log-level warning
# iptables -A LOGDROP -j DROP
```

Entrega #1. Una vez definidas todas las reglas de filtrado anteriores, vuelca el listado de reglas de iptables en un archivo de texto (comando: `# iptables-save > iptables-rules.txt`) y entrega dicho archivo.

NAT y port forwarding

Usando la misma configuración de red, vamos a suponer que la red interna es privada y la red externa es pública.

Antes de configurar las reglas de NAT y *port forwarding*, borra en el `FW` todas las reglas definidas anteriormente, mediante la orden:

```
# iptables -F
```

Comprueba que el `FW` tiene definidas las políticas por defecto de las tres cadenas de la tabla `FILTER` (`INPUT, OUTPUT, FORWARD`) a descartar (`DROP`).

Permite el acceso al interfaz *loopback* en el `FW`:

```
$ sudo iptables -A INPUT -i lo -j ACCEPT
$ sudo iptables -A OUTPUT -o lo -j ACCEPT
```

A continuación, define en el `FW` las reglas NAT y las reglas de filtrado que se indican a continuación:

1. Traducir la IP privada a la IP pública del `FW` (`1.1.1.1`) para todos los paquetes que salen de la red privada (interna) hacia la red pública (externa).

   Para ello mediante es necesario definir una regla de la tabla `nat` de tipo `MASQUERADE`. Esta regla se aplicará a todos los paquetes que salen por la interfaz de red pública del `FW` (`eth2`).

   Adicionalmente, será necesario añadir dos reglas de filtrado adicionales a la cadena `FORWARD` de la tabla `filter`:
   - Una regla para permitir todas las conexiones nuevas (estado `NEW`) establecidas desde la red interna (es decir, aquellas que entran por la interfaz `eth1` del `FW`)
   - Otra regla para permitir todos los paquetes pertenecientes a conexiones en estado `ESTABLISHED` o `RELATED`

   Comprueba que es posible conectarse desde las máquinas internas (`Host_A` y `Host_B`) a la máquina externa (`Host_E`), usando `ping`, `ssh` o estableciendo una conexión al servidor web de la máquina externa. Usando `wireshark` en las máquinas internas y en la máquina externa, comprueba que las direcciones IP privadas de la red interna se

traducen a la IP pública del FW cuando los paquetes salen a la red externa.

Observa en el FW la tabla de estado de conexiones en el fichero /proc/net/nf_conntrack.

2. Crear un servidor web virtual en el Host_A mediante *port forwarding*.

   Para ello es necesario añadir una regla de la tabla nat de tipo DNAT para los paquetes que entren por la interfaz externa (eth2) y que vayan dirigidos al puerto 80 del FW, de manera que éstos se traducirán a la IP y al puerto 80 del Host_A (acción -j DNAT --to 192.168.1.10:80).

   Adicionalmente, será necesario añadir una regla de filtrado adicional a la cadena FORWARD de la tabla filter para permitir todas las conexiones nuevas (estado NEW) que vayan dirigidas al puerto 80 del Host_A.

   Comprueba que es posible establecer una conexión web desde el host externo (Host_E) hasta la IP pública del FW (1.1.1.1). El FW redirigirá esta conexión al puerto 80 del Host_A. Usando wireshark en el Host_E y en el Host_A, comprueba que dirección IP pública del FW se traduce a la IP privada del Host_A cuando los paquetes entran a la red interna.

   Observa en el FW la tabla de estado de conexiones en el fichero /proc/net/nf_conntrack.

3. Crear un servidor ssh virtual en el Host_B mediante *port forwarding*.

   Para ello es necesario añadir una regla de la tabla nat de tipo DNAT para los paquetes que entren por la interfaz externa (eth2) y que vayan dirigidos al puerto 22 del FW, de manera que éstos se traducirán a la IP y al puerto 22 del Host_B (acción -j DNAT --to 192.168.1.11:22).

   Adicionalmente, será necesario añadir una regla de filtrado adicional a la cadena FORWARD de la tabla filter para permitir todas las conexiones nuevas (estado NEW) que vayan dirigidas al puerto 22 del Host_B.

   Comprueba que es posible establecer una conexión ssh desde el host externo (Host_E) hasta la IP pública del FW (1.1.1.1). El FW redirigirá esta conexión al puerto 22 del Host_B. Usando wireshark en el Host_E y en el Host_B, comprueba que dirección IP pública del FW se traduce a la IP privada del Host_B cuando los paquetes entran a la red interna.

   Observa en el FW la tabla de estado de conexiones en el fichero /proc/net/nf_conntrack.

Entrega #2. Copia y entrega la salida del comando anterior

Entrega #3. Una vez definidas todas las reglas de NAT, port forwarding y filtrado anteriores, vuelca el listado de reglas de iptables en un archivo de texto (comando: `# iptables-save > iptables-rules.txt`) y entrega dicho archivo.