CÓMO FUNCIONA IPTABLES

**Contenido**

[1 Pasos previos 3](#_heading=h.gjdgxs)

[1.1 Activar el bit de forwarding 3](#_heading=h.30j0zll)

[1.2 Lanzamiento de iptables al inicio del sistema 3](#_heading=h.1fob9te)

[2 Reglas 3](#_heading=h.3znysh7)

[3 Práctica del curso 3](#_heading=h.2et92p0)

[3.1 Esquema 3](#_heading=h.tyjcwt)

[3.2 Script 4](#_heading=h.3dy6vkm)

[3.2.1 Cuestiones a tener en cuenta 4](#_heading=h.1t3h5sf)

[3.2.2 Escribir el script 5](#_heading=h.4d34og8)

[3.2.3 Ejecutamos el script. 6](#_heading=h.2s8eyo1)

[3.3 Procesamiento NETFILTER 7](#_heading=h.17dp8vu)

[3.3.1 El esquema 7](#_heading=h.3rdcrjn)

[3.3.2 Ejemplos 7](#_heading=h.lnxbz9)

[3.4 Mejoras en el script 14](#_heading=h.35nkun2)

[3.4.1 Proxy 14](#_heading=h.1ksv4uv)

[4 Bibliografia 15](#_heading=h.44sinio)

# Pasos previos

## Activar el bit de forwarding

Para que un equipo LINUX enrute, hay que activar el bit de forwarding. Si no está activado, dos equipos de dos redes diferentes conectadas directamente al firewall no deberían verse. En esta práctica aparece para ello una orden en el propio script:

echo "1">/proc/sys/net/ipv4/ip\_forward

## Lanzamiento de iptables al inicio del sistema

1. Crear un script con las reglas en ***/etc****.*
2. Asignar permiso de ejecución a ese script.
3. En el archivo ***/etc/rc.local***, incluir una línea con la ruta absoluta del script.

# Reglas

Iptables -I 🡪 \*insert\*. Inserta una regla al principio de IPTables (se suele usar para probar una regla de forma rápida, luego se borra y ya está).

Iptables -A -> \*append\*. Añade una regla al final de la cadena elegida (se usa en los scripts, porque lo que queremos es que se vayan guardando las reglas en el orden en el que las escribimos en el script).

Cuando queramos incluir un puerto en una regla, podremos poner tanto el número de puerto como el nombre del servicio al que hace referencia (ese nombre deberá aparecer asociado a ese servicio en /etc/services).

Creado de regla: *iptables –I input –m state –state NEW –j DROP*

Borrado de regla: *iptables –D input –m state –state NEW –j DROP*

# Práctica del curso

## Esquema



Una máquina, por muchas interfaces que tenga, sólo debe tener configurada una Default Gateway. En nuestro caso, en la máquina Firewall, la única interfaz que tendrá una Default Gateway definida será la que conecte con Internet (10.9.52.33).

## Script

### Cuestiones a tener en cuenta

Conviene que la política por defecto sea DROP por seguridad y que luego aceptemos lo que nos interese.

¿Por qué DROP y no REJECT? Porque con REJECT se manda una respuesta al emisor y eso puede dar pistas a un “agresor”.

### Escribir el script

#Ojo, hay que dar permisos de ejecución a este script

#Mejora posible: utilizar las interfaces además de las IPs

#La IP pública “simulada” es la 10.9.52.33. Cada uno tendremos una diferente

#Vamos a limpiar todas las tablas de IPTABLES

IPTABLES=/sbin/iptables

$IPTABLES -P OUTPUT ACCEPT

$IPTABLES -P INPUT ACCEPT

$IPTABLES -P FORWARD ACCEPT

cat /proc/net/ip\_tables\_names | while read table; do

test "X$table" = "Xmangle" && continue

$IPTABLES -t $table -L -n | while read c chain rest; do

if test "X$c" = "XChain" ; then

$IPTABLES -t $table -F $chain

fi

done

$IPTABLES -t $table -X

done

#Hasta aquí la limpieza

#Definición de variables

RED\_INTERNA=192.168.2.0/24

RED\_DMZ=192.168.3.0/24

IP\_PUBLICA=10.9.52.33

#Activamos la opción forward

echo "1">/proc/sys/net/ipv4/ip\_forward

#Ponemos la política en todo prohibido : DROP

iptables -P INPUT DROP

iptables -P OUTPUT DROP

iptables -P FORWARD DROP

#NAT saliente para la red interna. Sale a Internet con la IP pública del FW

iptables -t nat -A POSTROUTING -s 192.168.2.0/24 ! -d 192.168.3.0/24 -j SNAT --to-source $IP\_PUBLICA

#NAT saliente para la red DMZ. Sale a Internet con la IP pública del FW

iptables -t nat -A POSTROUTING -s 192.168.3.0/24 ! -d 192.168.2.0/24 -j SNAT --to-source $IP\_PUBLICA

#NAT entrante para que las conexiones http a la IP pública se manden al servidor de la DMZ

iptables -t nat -A PREROUTING -s 0/0 -d $IP\_PUBLICA -p tcp --dport 80 -j DNAT --to 192.168.3.2

#NAT entrante para que las conexiones ssh a la IP pública se manden al servidor de la DMZ

iptables -t nat -A PREROUTING -s 0/0 -d $IP\_PUBLIKOA -p tcp --dport 22 -j DNAT --to 192.168.3.2

#Permitimos todo tipo de tráfico por el interface loopback. Hay que tener en cuenta que aplicaciones locales en ocasiones hacen peticios a servicios de red de la máquina local (esto tiene que ver, por ejemplo, con X11)

iptables -A INPUT -i lo -j ACCEPT

iptables -A FORWARD -i lo -j ACCEPT

iptables -A OUTPUT -o lo -j ACCEPT

#Permitimos conexiones ya establecidas y relacionadas en las 3 cadenas

iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

#Permitimos navegar (puerto 80) desde la red interna a la DMZ

iptables -A FORWARD -s 192.168.2.0/24 -d 192.168.3.2 -p tcp --dport 80 -m state --state NEW -j ACCEPT

#Permitimos navegar a Internet desde la red interna

iptables -A FORWARD -s 192.168.2.0/24 ! -d 192.168.3.0/24 -p tcp --dport 80 -m state --state NEW -j ACCEPT

#Permitimos que el DNS privado haga peticiones al DNS definido en el campo 'forwarders' del archivo "/etc/bind/named.conf.options" del

iptables -A FORWARD -s 192.168.2.0/24 ! -d 192.168.3.0/24 -p udp --dport 53 -m state --state NEW -j ACCEPT

#Permitimos navegar a Internet desde la red DMZ

iptables -A FORWARD -s 192.168.3.0/24 ! -d 192.168.2.0/24 -p tcp --dport 80 -m state --state NEW -j ACCEPT

#Permitimos que el propio firewall se conecte a cualquier cosa

iptables -A OUTPUT -s $IP\_PUBLICA -d 0/0 -m state --state NEW -j ACCEPT

iptables -A OUTPUT -s 192.168.2.1 -d 0/0 -m state --state NEW -j ACCEPT

iptables -A OUTPUT -s 192.168.3.1 -d 0/0 -m state --state NEW -j ACCEPT

#Permito ssh desde Internet al servidor de la DMZ

iptables -A FORWARD -s 0/0 -d 192.168.3.2 -p tcp --dport 22 -m state --state NEW -j ACCEPT

#Permitimos conexiones al servidor WEB desde Internet

iptables -A FORWARD -p tcp --dport 80 -m state --state NEW -j ACCEPT

#Permitimos conexiones ssh (puerto 22) desde la red interna y DMZ al FW

iptables -A INPUT -s $RED\_DMZ -d 192.168.3.1 -p tcp --dport 22 -m state --state NEW -j ACCEPT

iptables -A INPUT -s $RED\_INTERNA -d 192.168.2.1 -p tcp --dport 22 -m state --state NEW -j ACCEPT

#Permitimos PING desde la red interna y DMZ al FW

iptables -A INPUT -s $RED\_DMZ -d 192.168.3.1 -p icmp -m state --state NEW -j ACCEPT

iptables -A INPUT -s $RED\_INTERNA -d 192.168.2.1 -p icmp -m state --state NEW -j ACCEPT

#Permitimos PING desde la red interna a la pata de la DMZ del FW

iptables -A INPUT -s $RED\_INTERNA -d 192.168.3.1 -p icmp -m state --state NEW -j ACCEPT

#Permitimos PING desde la DMZ a la pata interna del FW (ojo, es para probar, no conviene permitir ningún tráfico desde la DMZ hacia la LAN)

#iptables -A INPUT -s 192.168.3.0/24 -d 192.168.2.1 -p icmp -m state --state NEW -j ACCEPT

#Permitimos PINg desde Internet a la IP pública

iptables -A INPUT -s 0/0 -d $IP\_PUBLICA -p icmp -m state --state NEW -j ACCEPT

#Permitimos ping desde la red interna y la DMz a cualquier sitio (excluído desde la DMZ a la LAN)

iptables -A FORWARD -s $RED\_INTERNA -d 0/0 -p icmp -m state --state NEW -j ACCEPT

iptables -A FORWARD -s $RED\_DMZ ! -d $RED\_INTERNA -p icmp -m state --state NEW -j ACCEPT

### Ejecutamos el script.

Le tenemos que dar permisos de ejecución al archivo. 750 es suficiente, por ejemplo.

Una vez ejecutado, las cadenas quedan ordenadas de la siguiente forma:

#### TABLA FILTER

Cadena INPUT

1. iptables -A INPUT -i lo -j ACCEPT
2. iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
3. iptables -A INPUT -s $RED\_DMZ -d 192.168.3.1 -p tcp --dport 22 -m state --state NEW -j ACCEPT
4. iptables -A INPUT -s $RED\_INTERNA -d 192.168.2.1 -p tcp --dport 22 -m state --state NEW -j ACCEPT
5. iptables -A INPUT -s $RED\_DMZ -d 192.168.3.1 -p icmp -m state --state NEW -j ACCEPT
6. iptables -A INPUT -s $RED\_INTERNA -d 192.168.2.1 -p icmp -m state --state NEW -j ACCEPT
7. iptables -A INPUT -s $RED\_INTERNA -d 192.168.3.1 -p icmp -m state --state NEW -j ACCEPT
8. iptables -A INPUT -s 0/0 -d $IP\_PUBLICA -p icmp -m state --state NEW -j ACCEPT

Cadena OUTPUT

1. iptables -A OUTPUT -o lo -j ACCEPT
2. iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
3. iptables -A OUTPUT -s $IP\_PUBLICA -d 0/0 -m state --state NEW -j ACCEPT
4. iptables -A OUTPUT -s 192.168.2.1 -d 0/0 -m state --state NEW -j ACCEPT
5. iptables -A OUTPUT -s 192.168.3.1 -d 0/0 -m state --state NEW -j ACCEPT

Cadena FORWARD

1. iptables -A FORWARD -i lo -j ACCEPT
2. iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
3. iptables -A FORWARD -s 192.168.2.0/24 -d 192.168.3.2 -p tcp --dport 80 -m state --state NEW -j ACCEPT
4. iptables -A FORWARD -s 192.168.2.0/24 ! -d 192.168.3.0/24 -p tcp --dport 80 -m state --state NEW -j ACCEPT
5. iptables -A FORWARD -s 192.168.2.0/24 ! -d 192.168.3.0/24 -p udp --dport 53 -m state --state NEW -j ACCEPT
6. iptables -A FORWARD -s 192.168.3.0/24 ! -d 192.168.2.0/24 -p tcp --dport 80 -m state --state NEW -j ACCEPT
7. iptables -A FORWARD -s 0/0 -d 192.168.3.2 -p tcp --dport 22 -m state --state NEW -j ACCEPT
8. iptables -A FORWARD -s $RED\_INTERNA -d 0/0 -p icmp -m state --state NEW -j ACCEPT
9. iptables -A FORWARD -s $RED\_DMZ ! -d $RED\_INTERNA -p icmp -m state --state NEW -j ACCEPT

#### Tabla NAT

Cadena PREROUTING

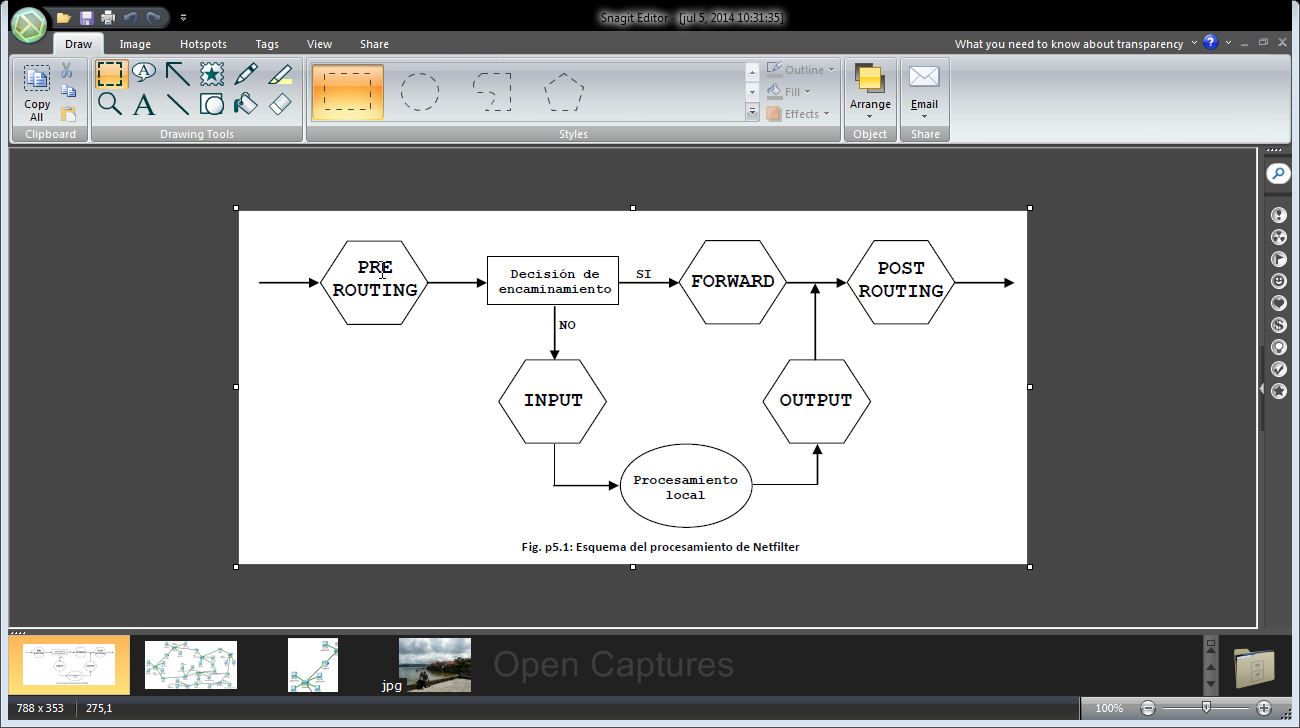
1. iptables -t nat -A PREROUTING -s 0/0 -d $IP\_PUBLICA -p tcp --dport 80 -j DNAT --to 192.168.3.2
2. iptables -t nat -A PREROUTING -s 0/0 -d $IP\_PUBLIKOA -p tcp --dport 22 -j DNAT --to 192.168.3.2

Cadena POSTROUTING

1. iptables -t nat -A POSTROUTING -s 192.168.2.0/24 ! -d 192.168.3.0/24 -j SNAT --to-source $IP\_PUBLICA
2. iptables -t nat -A POSTROUTING -s 192.168.3.0/24 ! -d 192.168.2.0/24 -j SNAT --to-source $IP\_PUBLICA

## Procesamiento NETFILTER

### El esquema



Este esquema se refiere exclusivamente a lo que ocurre en la máquina Firewall. Cuando se habla de procesamiento local, se está hablando de que la máquina Firewall procesa el paquete (bien porque es de entrada hacia la propia máquina o bien porque es de salida desde ella). No, por lo tanto, procesamiento a nivel de la red local (a veces lleva a engaño).

### Ejemplos

Hay que tener en cuenta que para que se lleve a cabo la acción de una regla, todas las condiciones de esa regla se tienen que cumplir (todas, un “y” y no un “o”). La acción que se lleva a cabo aparece después de “-j”.

En estos ejemplos sólo vamos a analizar los establecimientos de la conexión (paquetes con flag “syn” activado, “Soy A y quiero empezar una conexión contigo: B”. Estado NEW en IPTABLES). En TCP, después del anterior, se manda un paquete con los bits “syn” y “ack” activados (“Soy B y estoy de acuerdo en empezar la conexión con A”) como respuesta al paquete de establecimiento previo (. Y también otro paquete final con el bit “ack” activado desde el origen al destino (Soy A. “Vale, B, empecemos”). Estos dos últimos tipos de paquetes paquetes se aceptan siempre gracias a las reglas 2, 10 y 15. (Estado Established en IPTABLES o incluso Related – aunque éste se usa muy casualmente, con conexiones ya abiertas de FTP-).

Es muy importante también tener en cuenta que los paquetes que viajan dentro de una misma red o subred no pasan por el Firewall.

#### Petición ssh desde la red LAN hacia el Firewall

Para que funcione, el firewall tiene que tener el servicio openssh instalado.

Supongamos que desde el terminal del PC de la red LAN con IP 192.168.2.2 se escribe: ‘ssh [cliente@192.168.2.1](about:blank)’. El paquete generado se tiene que aceptar.

| Información del paquete | **Protocolo** | **IP de origen** | **IP de destino** | **Puerto de destino** |
| --- | --- | --- | --- | --- |
| TCP | 192.168.2.2 | 192.168.2.1 | 22 |

Estos son los pasos que el paquete seguirá en el esquema:

1. ¿Se trata de un paquete generado en el propio Firewall?

No; por lo tanto, va al principio.

1. **Cadena PREROUTING.**

Se analiza la regla 23. ¿Se cumplen todas las condiciones?

* 1. El paquete viene desde cualquier sitio SÍ.
  2. El objetivo es la IP pública NO.
  3. Es un paquete TCP SÍ.
  4. El puerto de destino es el 80 NO.

Como no se cumplen todas las condiciones, se pasa el paquete a la siguiente regla de la cadena. Se analiza ahora, por lo tanto, la regla 24. ¿Se cumplen todas las condiciones?

1. El paquete viene desde cualquier sitio SÍ.
2. El objetivo es la IP pública NO.
3. Es un paquete TCP SÍ.
4. El puerto de destino es el 22 SÍ.

Como no se cumplen todas las condiciones y como no hay más reglas en la cadena PREROUTING, el paquete seguirá adelante en el esquema. Ahora, tendrá que tomar la decisión de encaminamiento: ¿la IP de destino (192.168.2.1) es una IP del propio Firewall? Sí. Entonces, el paquete irá a la cadena *INPUT*.

1. **Cadena INPUT**

Se van analizando las reglas desde la 1 en adelante en orden. En cuanto se cumplan todas las condiciones de una, se ejecuta la acción que haya en ella y no se analizan más reglas en esa cadena. La primera cuyas condiciones (todas) se cumplen es la 4. ¿Cuál es la acción? ACEPTAR. Luego, ese paquete el sistema lo acepta y sigue su camino en el esquema, llegando al PROCESAMIENTO LOCAL.

.

Por lo tanto, como queríamos, el paquete se ha aceptado, permitiéndose la comunicación.

#### Petición ssh desde Internet hacia la máquina que tiene el Firewall

Supongamos que alguien desde un PC de la red del instituto –nuestro Internet simulado- con IP 10.9.53.201 escribe en su terminal: ‘ssh [cliente@10.9.52.33](mailto:cliente@10.9.52.33)’. El paquete generado se tiene que aceptar.

| Información del paquete | **Protocolo** | **IP de origen** | **IP de destino** | **Puerto de destino** |
| --- | --- | --- | --- | --- |
| TCP | 10.9.53.201 | 10.9.52.33 | 22 |

Estos son los pasos que el paquete seguirá en el esquema:

1. ¿Se trata de un paquete generado en el propio Firewall?

No; por lo tanto, va al principio.

1. **Cadena PREROUTING.**

Se analiza la regla 23. ¿Se cumplen todas las condiciones?

* 1. El paquete viene desde cualquier sitio SÍ.
  2. El objetivo es la IP pública SÍ.
  3. Es un paquete TCP SÍ.
  4. El puerto de destino es el 80 NO.

Como no se cumplen todas las condiciones, se pasa el paquete a la siguiente regla de la cadena. Se analiza ahora, por lo tanto, la regla 24. ¿Se cumplen todas las condiciones?

1. El paquete viene desde cualquier sitio SÍ.
2. El objetivo es la IP pública SÍ.
3. Es un paquete TCP SÍ.
4. El puerto de destino es el 22 SÍ.

Como se cumplen todas las condiciones, se ejecutará la acción de la regla 24: DNAT. Así, la dirección IP de destino se cambiará, quedando así la información del paquete:

| Información del paquete | **Protocolo** | **IP de origen** | **IP de destino** | **Puerto de destino** |
| --- | --- | --- | --- | --- |
| TCP | 10.9.53.201 | 192.168.3.2 | 22 |

Ahora, tendrá que tomar la decisión de encaminamiento: ¿la IP de destino (192.168.3.2) es una de las IPs del propio Firewall? No. Entonces, el paquete irá a la cadena *FORWARD*.

1. **Cadena *FORWARD***

Se van analizando las reglas desde la 14 en adelante en orden. No se cumplen todas las condiciones de ninguna entre la 14 y la 19. Sí, en cambio, las de la 20. Es por ello por lo que se ejecutará la acción de la regla 20: ACEPTAR. En cuanto se cumpla una, se ejecuta la acción que haya en ella y no se analizan más reglas en esa cadena. La primera que se cumple es la 4. ¿Cuál es la acción? Aceptar. Luego, ese paquete el sistema lo acepta y sigue su camino en el esquema, llegando a la cadena *POSTROUTING*.

1. **Cadena POSTROUTING**

No se cumple la regla 25 ni la 26. El paquete saldrá del esquema NETFILTER.

Por lo tanto, como queríamos, el paquete se ha aceptado, permitiéndose la comunicación.

#### PING desde la red DMZ hacia la red LAN

Supongamos que desde el terminal del servidor Apache de la red DMZ con IP 192.168.3.2 se escribe: ‘ping 192.168.2.2’. El paquete generado se tiene que denegar.

| Información del paquete | **Protocolo** | **IP de origen** | **IP de destino** | **Puerto de destino** |
| --- | --- | --- | --- | --- |
| ICMP | 192.168.3.2 | 192.168.2.2 | --- |

Estos son los pasos que el paquete seguirá en el esquema:

1. ¿Se trata de un paquete generado en el propio Firewall?

No; por lo tanto, va al principio.

1. **Cadena PREROUTING.**

Se analiza la regla 23. ¿Se cumplen todas las condiciones?

* 1. El paquete viene desde cualquier sitio SÍ.
  2. El objetivo es la IP pública NO.
  3. Es un paquete TCP NO.
  4. El puerto de destino es el 80 NO.

Como no se cumplen todas las condiciones, se pasa el paquete a la siguiente regla de la cadena.

Se analiza ahora, por lo tanto, la regla 24. ¿Se cumplen todas las condiciones?

1. El paquete viene desde cualquier sitio SÍ.
2. El objetivo es la IP pública NO.
3. Es un paquete TCP NO.
4. El puerto de destino es el 22 NO.

Como no se cumplen todas las condiciones y como no hay más reglas en la cadena PREROUTING, el paquete seguirá adelante en el esquema. Ahora, tendrá que tomar la decisión de encaminamiento: ¿la IP de destino (192.168.2.2) es una IP del propio Firewall? No. Entonces, el paquete irá a la cadena *FORWARD*.

1. **Cadena *FORWARD***

Se van analizando las reglas desde la 14 en adelante en orden. No se cumplen todas las condiciones de ninguna regla. ATENCIÓN. La regla 22 acepta todo el tráfico ICMP desde la DMZ hacia cualquier destino excepto hacia la red LAN. No hay, por lo tanto, ninguna acción a ejecutar. ¿Qué se hará entonces? Ejecutar la acción que indica la política por defecto; es decir: DROP. Luego, ese paquete el sistema lo acepta y sigue su camino en el esquema, llegando a la cadena *POSTROUTING*.

1. **Cadena POSTROUTING**

No se cumple la regla 25 ni la 26. El paquete saldrá del esquema NETFILTER.

Por lo tanto, como queríamos, el paquete se ha denegado, no permitiéndose la comunicación.

#### Navegación hacia la DMZ desde Internet

Supongamos que alguien desde el PC con IP 10.9.53.201 de la red del instituto –nuestro Internet simulado- escribe <http://10.9.52.33> en el navegador. El paquete generado se tiene que aceptar, y permitir la navegación.

| Información del paquete | **Protocolo** | **IP de origen** | **IP de destino** | **Puerto de destino** |
| --- | --- | --- | --- | --- |
| TCP | 10.9.53.201 | 10.9.52.33 | 80 |

Estos son los pasos que el paquete seguirá en el esquema:

1. ¿Se trata de un paquete generado en el propio Firewall?

No; por lo tanto, va al principio.

1. **Cadena PREROUTING.**

Se analiza la regla 23. ¿Se cumplen todas las condiciones?

* 1. El paquete viene desde cualquier sitio SÍ.
  2. El objetivo es la IP pública SÍ.
  3. Es un paquete TCP SÍ.
  4. El puerto de destino es el 80 SÍ.

Como se cumplen todas las condiciones, se ejecutará la acción de la regla 23: DNAT. Así, la dirección IP de destino se cambiará, quedando así la información del paquete:

| Información del paquete | **Protocolo** | **IP de origen** | **IP de destino** | **Puerto de destino** |
| --- | --- | --- | --- | --- |
| TCP | 10.9.53.201 | 192.168.3.2 | 80 |

Ahora, tendrá que tomar la decisión de encaminamiento: ¿la IP de destino (192.168.3.2) es una de las IPs del propio Firewall? No. Entonces, el paquete irá a la cadena *FORWARD*.

1. **Cadena *FORWARD***

Se van analizando las reglas desde la 14 en adelante en orden. No se cumplen todas las condiciones de la 14 ni de la 15. Sí, en cambio, las de la 16. Es por ello por lo que se ejecutará la acción de la regla 16: ACEPTAR. Luego, ese paquete el sistema lo acepta y sigue su camino en el esquema, llegando a la cadena *POSTROUTING*.

1. **Cadena POSTROUTING**

No se cumple la regla 25 ni la 26. El paquete saldrá del esquema NETFILTER.

Por lo tanto, como queríamos, el paquete se ha aceptado, permitiéndose la comunicación.

#### Petición DNS hacia el exterior

Supongamos que en el navegador del PC-LAN con IP 192.168.2.2 escribimos <http://www.google.es>. Como ese PC tiene como servidor DNS el 192.168.2.3, le consultará a él. Como éste no resuelve nombres públicos, lo enviará al servidor DNS del centro —porque así lo tenemos configurado en el archivo *“/etc/bind/named.conf.options*” del servidor DNS —: 10.9.55.1. Por lo tanto, tenemos un paquete UDP con IP de origen 192.168.2.3 e IP de destino 10.9.55.1 (que es la IP del servidor DNS de la LAN del centro, pero en nuestro caso simula la IP de un servidor DNS público cualquiera de Internet). El paquete generado se tiene que aceptar.

| Información del paquete | **Protocolo** | **IP de origen** | **IP de destino** | **Puerto de destino** |
| --- | --- | --- | --- | --- |
| UDP | 192.168.2.3 | 10.9.55.1 | 53 |

Estos son los pasos que el paquete seguirá en el esquema:

1. ¿Se trata de un paquete generado en el propio Firewall?

No; por lo tanto, va al principio.

1. **Cadena PREROUTING.**

Se analiza la regla 23. ¿Se cumplen todas las condiciones?

* 1. El paquete viene desde cualquier sitio SÍ.
  2. El objetivo es la IP pública NO.
  3. Es un paquete TCP NO.
  4. El puerto de destino es el 80 NO.

Como no se cumplen todas las condiciones, se pasa el paquete a la siguiente regla de la cadena.

Se analiza ahora, por lo tanto, la regla 24. ¿Se cumplen todas las condiciones?

1. El paquete viene desde cualquier sitio SÍ.
2. El objetivo es la IP pública NO.
3. Es un paquete TCP NO.
4. El puerto de destino es el 22 NO.

Como no se cumplen todas las condiciones y como no hay más reglas en la cadena PREROUTING, el paquete seguirá adelante en el esquema. Ahora, tendrá que tomar la decisión de encaminamiento: ¿la IP de destino (10.9.55.1) es una IP del propio Firewall? No. Entonces, el paquete irá a la cadena FORWARD.

1. **Cadena *FORWARD***

Se van analizando las reglas desde la 14 en adelante en orden. No se cumplen todas las condiciones de las reglas 14-17. Sí, en cambio, las de la 18. Es por ello por lo que se ejecutará la acción de la regla 18: ACEPTAR. Luego, ese paquete el sistema lo acepta y sigue su camino en el esquema, llegando a la cadena *POSTROUTING*.

1. **Cadena POSTROUTING**

Se analiza la regla 25. ¿Se cumplen todas las condiciones?

* 1. El paquete viene de la red 192.168.2.0/24 SÍ
  2. El objetivo no es una IP de la red DMZ. SÍ

Como se cumplen todas las condiciones, se ejecutará la acción de la regla 25: SNAT. Así, la dirección IP de origen se cambiará, quedando así la información del paquete:

| Información del paquete | **Protocolo** | **IP de origen** | **IP de destino** | **Puerto de destino** |
| --- | --- | --- | --- | --- |
| UDP | 10.9.52.33 | 10.9.55.1 | 53 |

A continuación, el paquete saldrá del sistema.

Por lo tanto, como queríamos, el paquete se ha aceptado, permitiéndose la comunicación.

#### Navegación hacia Internet desde el Firewall

Supongamos que en el navegador del Firewall escribimos <http://173.194.34.88> (una IP pública de Google). El paquete generado se tiene que aceptar, y permitir la navegación.

| Información del paquete | **Protocolo** | **IP de origen** | **IP de destino** | **Puerto de destino** |
| --- | --- | --- | --- | --- |
| TCP | 10.9.52.33 | 173.194.24.88 | 80 |

Estos son los pasos que el paquete seguirá en el esquema:

1. ¿Se trata de un paquete generado en el propio Firewall?

Sí; es un paquete de procesamiento local e irá a la cadena *OUTPUT*.

1. **Cadena OUTPUT.**

Se van analizando las reglas desde la 9 en adelante en orden. No se cumplen todas las condiciones de la 9 ni de la 10. Sí, en cambio, las de la 11. Es por ello por lo que se ejecutará la acción de la regla 11: ACEPTAR. Luego, ese paquete sigue su camino en el esquema, llegando a la cadena *POSTROUTING*.

1. **Cadena POSTROUTING**

No se cumple la regla 25 ni la 26. El paquete saldrá del esquema NETFILTER.

Por lo tanto, como queríamos, el paquete se ha aceptado, permitiéndose la comunicación.

#### Navegación hacia Internet desde la red LAN

Supongamos que en el navegador del PC-LAN con IP 192.168.2.2 escribimos <http://173.194.34.88> (una IP pública de Google). El paquete generado se tiene que aceptar, y permitir la navegación.

| Información del paquete | **Protocolo** | **IP de origen** | **IP de destino** | **Puerto de destino** |
| --- | --- | --- | --- | --- |
| TCP | 192.168.2.2 | 173.194.24.88 | 80 |

Estos son los pasos que el paquete seguirá en el esquema:

1. ¿Se trata de un paquete generado en el propio Firewall?

No; por lo tanto, va al principio.

1. **Cadena PREROUTING.**

Se analiza la regla 23. ¿Se cumplen todas las condiciones?

* 1. El paquete viene desde cualquier sitio SÍ.
  2. El objetivo es la IP pública NO.
  3. Es un paquete TCP SÍ.
  4. El puerto de destino es el 80 SÍ.

Como no se cumplen todas las condiciones, se pasa el paquete a la siguiente regla de la cadena.

Se analiza ahora, por lo tanto, la regla 24. ¿Se cumplen todas las condiciones?

1. El paquete viene desde cualquier sitio SÍ.
2. El objetivo es la IP pública NO.
3. Es un paquete TCP SÍ.
4. El puerto de destino es el 22 NO.

Como no se cumplen todas las condiciones y como no hay más reglas en la cadena PREROUTING, el paquete seguirá adelante en el esquema. Ahora, tendrá que tomar la decisión de encaminamiento: ¿la IP de destino (173.194.34.88) es una IP del propio Firewall? No. Entonces, el paquete irá a la cadena FORWARD.

1. **Cadena *FORWARD***

Se van analizando las reglas desde la 14 en adelante en orden. No se cumplen todas las condiciones de las reglas 14-16. Sí, en cambio, las de la 17. Es por ello por lo que se ejecutará la acción de la regla 17: ACEPTAR. Luego, sigue su camino en el esquema, llegando a la cadena *POSTROUTING*.

1. **Cadena POSTROUTING**

Se analiza la regla 25. ¿Se cumplen todas las condiciones?

* 1. El paquete viene de la red 192.168.2.0/24 SÍ
  2. El objetivo no es una IP de la red DMZ. SÍ

Como se cumplen todas las condiciones, se ejecutará la acción de la regla 25: SNAT. Así, la dirección IP de origen se cambiará, quedando así la información del paquete:

| Información del paquete | **Protocolo** | **IP de origen** | **IP de destino** | **Puerto de destino** |
| --- | --- | --- | --- | --- |
| UDP | 10.9.52.33 | 173.194.24.88 | 53 |

A continuación, el paquete saldrá del sistema.

Por lo tanto, como queríamos, el paquete se ha aceptado, permitiéndose la comunicación.

## Mejoras en el script

### Proxy

﻿Es posible que el propio cortafuegos deba realizar también las tareas de servidor proxy para permitir o denegar las páginas web consultadas tanto desde la red local como desde la DMZ. Para ello, podríamos configurar el servicio squid, y ponerlo a la escucha en el puerto 3128.

Para ello, deberemos redirigir las navegaciones desde la DMZ o desde la red local hacia Internet al puerto 3128. Tendríamos que añadir las siguientes líneas al script:

iptables -t nat -A PREROUTING –s 192.168.2.0 -p tcp --dport 80 -j REDIRECT --to-port 3128

iptables -t nat -A PREROUTING –s 192.168.3.0 -p tcp --dport 80 -j REDIRECT --to-port 3128

﻿Al hacer REDIRECT le estamos diciendo al cortafuegos que cuando le venga un paquete dirigido al puerto 80 lo redirija al 3128. Pero, ¿cómo se hace eso?

1. Al paquete se le tiene que cambiar la siguiente información:
2. La IP de destino será la 10.9.52.33, y no la IP pública de la página web a consultar.
3. El puerto de destino será el 3128, y no el 80.

Por lo tanto, al ser un paquete destinado al propio firewall, pasará a la cadena *INPUT*, y no a la cadena *FORWARD*. Una vez filtrado en la cadena *INPUT*, se realizará el procesado local.

1. Entonces, se creará un nuevo paquete:
2. La IP de destino será la IP pública de la página web que se quiere consultar.
3. El puerto de destino será el 80.

Al tratarse de un paquete creado en el propio firewall, se dirigirá a la cadena *OUTPUT* tras realizar el procesamiento local.

Debemos, pues, añadir al script las siguientes reglas:

iptables –A INPUT –i eth0 –p tcp --dport 3128 -j ACCEPT

iptables –A INPUT –i eth1 --dport 3128 -j ACCEPT

iptables –A OUTPUT –o eth2 –p tcp --dport 80 -j ACCEPT

# Bibliografia

COSTAS, Jesús: *Seguridad y Alta Disponibilidad*, Editorial Ra-Ma, Madril, 2011.

GUERRA, Ander: *Sareko Zerbitzuak*, Lanbide Ekimena, Gasteiz, 2012.

MOLINA, Francisco José: *Servicios e Red*, Editorial Ra-Ma, Madril, 2010.