Universidad Politécnica de Cartagena

Escuela Técnica Superior de Ingeniería de Telecomunicación

SEGURIDAD EN REDES

Práctica 5: Configuración de un firewall

Autores: Josemaría Malgosa Sanahuja
María Dolores Cano Baños

SEGURIDAD EN REDES

Configuración de un *firewall* PRÁCTICA 5

FIREWALL

Un firewall es el dispositivo de red diseñado para permitir/rechazar el tráfico de paquetes que circulan por él. Los firewalls se utilizan con frecuencia para evitar que los paquetes generados por usuarios de Internet no autorizados puedan acceder a redes corporativas (intranets). Todos los paquetes que entren o salgan de la intranet pasan a través del firewall, que examina cada paquete y bloquea aquellos que no cumplen los criterios de seguridad especificados. También es frecuente conectar al firewall a una tercera red, llamada "zona desmilitarizada" o DMZ, en la que se ubican los servidores de la organización que deben permanecer accesibles desde el exterior.

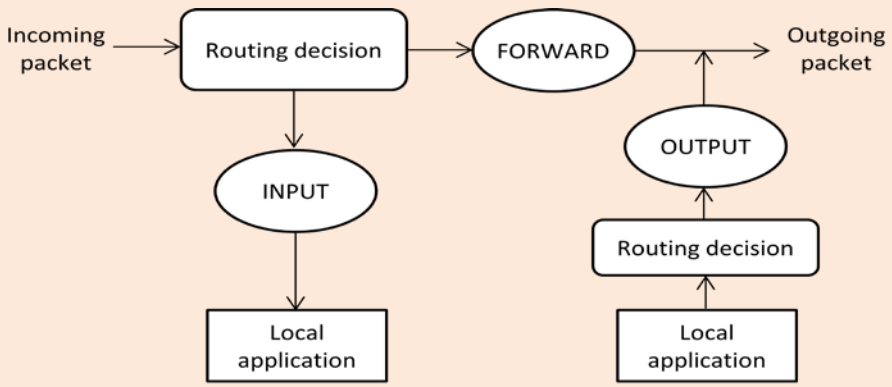
El proceso de filtrado se realiza mediante la inspección de las cabeceras de los paquetes. Si el *overhead* de un paquete coincide con el especificado por el conjunto de reglas del filtro, el paquete es aceptado/rechazado. Por tanto, cada paquete se filtra basándose en la información contenida en el propio paquete (interfaz de entrada y salida, dirección IP origen y destino, protocolo de transporte utilizado, puerto TCP/UDP origen o destino, etc.). Sin embargo, los firewalls también son capaces de procesar flujos de datos. Ahora, además de considerarse el *overhead* propio de cada paquete, también se analiza si el paquete pertenece a un flujo, es decir, a una conexión TCP/UDP establecida anteriormente.

Los firewalls pueden ser implementados en hardware o software. Actualmente, lo más habitual es utilizar una arquitectura hardware estándar controlada por un *kernel* de Linux compilado con la opción *Netfilter* activada. Con el comando *iptables* se pueden especificar las reglas para el filtrado de paquetes.

NETFILTER

Un paquete entra por un interfaz de red para ser finalmente entregado a una aplicación o viceversa, es generado por una aplicación para que finalmente salir por un interfaz de red determinado. También puede entrar por un interfaz de red y salir por otro sin pasar por ninguna aplicación (por ejemplo, cuando el ordenador actúa como un *router*). En todo este proceso, las distintas cabeceras del paquete son procesadas por el código del *kernel* de Linux. Si el *kernel* se ha compilado con la opción *Netfilter* habilitada, en algunos puntos determinados de ese código (llamados *chains*), se inserta una función (denominada *hook*) que por defecto, no hace nada. Sin embargo, a través del comando *iptables* se puede programar dicha función para que inspeccione el valor de los campos de las cabeceras del paquete y dependiendo del resultado de la inspección aplique las siguientes reglas (*rules*):

- Rechazar el paquete o aceptarlo para que siga siendo procesado por el *kernel* (tabla *filter*)
- Modificar algunos de los campos de la cabecera (tabla *mangle*)
- Modificar la dirección IP origen o destino (tabla *nat*)
- Trabajar a nivel de paquete y nunca a nivel de flujo (tabla *raw*)



En la figura se puede ver las *chains* INPUT, OUTPUT y FORWARD de la tabla *filter* asociadas con el nivel de red IP (que es la tabla más utilizada de las cuatro; en el portal de la asignatura hay un diagrama donde se muestra una descripción más detallada de las otras tablas, incluyendo también las tablas y cadenas propias del nivel de enlace).

En resumen, la cadena INPUT procesa todos los paquetes dirigidos hacia el propio ordenador, la OUTPUT procesa los paquetes generados por el propio ordenador y la FORWARD los paquetes en tránsito. Mencionar por último que las reglas del filtrado se ejecutan estrictamente en el orden especificado por el usuario. Cuando una regla se satisface, se dejan de evaluar las restantes reglas.

En el portal de la asignatura está disponible un tutorial muy completo y extenso sobre el funcionamiento de *iptables*. Las opciones más utilizadas del comando *iptables* son:

- Trabajar con la tabla *table* (-t *table*, si no se especifica se asume la tabla *filter*)
- Definir la política por defecto (-P).
- Lista de reglas en una cadena (-L)
- Vaciar las reglas de una cadena (-F)
- Crear una nueva cadena (-N)
- Añadir una regla a una cadena (-A)
- Borrar una regla en algún punto de la cadena, o la primera que coincida (-D).

1. ¿Cuál sería el resultado de ejecutar los siguientes comandos en un ordenador?

```
iptables -P INPUT DROP -> Impide las conexiones entrantes. POLITICA POR DEFECTO
iptables -P OUTPUT ACCEPT -> Acepta las conexiones salientes. POLITICA POR DEFECTO
iptables -A INPUT -m state –state ESTABLISHED,RELATED -j ACCEPT
-> Las conexiones entrantes que ya están establecidas se aceptan. Solo acepta las que haya generado previamente con un output
```

2. ¿Cuál sería el resultado de ejecutar los siguientes comandos en un ordenador con capacidad de *routing*?

```
iptables -P INPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -P OUTPUT ACCEPT
iptables -A INPUT -s 10.0.0.0/8 -j DROP
iptables -A INPUT -s 172.16.0.0/12 -j DROP
iptables -A INPUT -s 192.168.0.0/24 -j DROP
iptables -A OUTPUT -s 10.0.0.0/8 -j DROP
iptables -A OUTPUT -s 172.16.0.0/12 -j DROP
iptables -A OUTPUT -s 192.168.0.0/24 -j DROP
iptables -A FORWARD -s 10.0.0.0/8 -j DROP
iptables -A FORWARD -s 172.16.0.0/12 -j DROP
iptables -A FORWARD -s 192.168.0.0/24 -j DROP
```

```
Acepta todas las conexiones entrantes, de paquetes en tránsito y salientes.
Elimina los paquetes entrantes de 10.0.0.0/8
Elimina los paquetes entrantes de 172.16.0.0/12
Elimina los paquetes entrantes de 192.168.0.0/24
Elimina los paquetes salientes hacia 10.0.0.0/8
Elimina los paquetes salientes hacia 172.16.0.0/12
Elimina los paquetes salientes hacia 192.168.0.0/24
Elimina los paquetes en transito de 10.0.0.0/8
Elimina los paquetes en transito de 172.16.0.0/12
Elimina los paquetes en transito de 192.168.0.0/24
```

```
Impide que el router reciba, envíe o redireccione cualquier flujo de datos con IP privada.
```

3. ¿Cuál sería el resultado de ejecutar los siguientes grupos de comandos en el servidor de un laboratorio cuyos ordenadores tuvieran direccionamiento privado 192.168.2.xxx?

```
iptables -A INPUT -p tcp --dport 80 -s 192.168.2.0/24 -j ACCEPT
iptables –A INPUT –j DROP Solo las IP 192.168.2.0/24 se podrán conectar al puerto tcp 80

iptables -A INPUT -p tcp --dport 80 -m iprange --src-range 192.168.2.1-192.168.2.15 -j ACCEPT
iptables –A INPUT –j DROP Solo las IP 192.168.2.1 - 192.168.2.15 se podrán conectar al puerto 80

iptables -A INPUT -s 192.168.2.0/24 -j ACCEPT
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables –A INPUT –j DROP Solo las IP 192.168.2.0/24 se podrán conectar a cualquier puerto.
                          Cualquier IP se puede conectar SOLO al puerto 80
```

4. Suponiendo que el portal web del servidor del laboratorio fuera un servicio muy solicitado desde el exterior ¿cuál de los dos grupos de reglas es más adecuado para permitir el acceso desde el exterior solo al servidor web y a cualquier servicio desde la intranet?

```
iptables -A INPUT -s 192.168.2.0/24 -j ACCEPT
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables –A INPUT –j DROP
```

```
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A INPUT -s 192.168.2.0/24 -j ACCEPT
iptables –A INPUT –j DROP
```

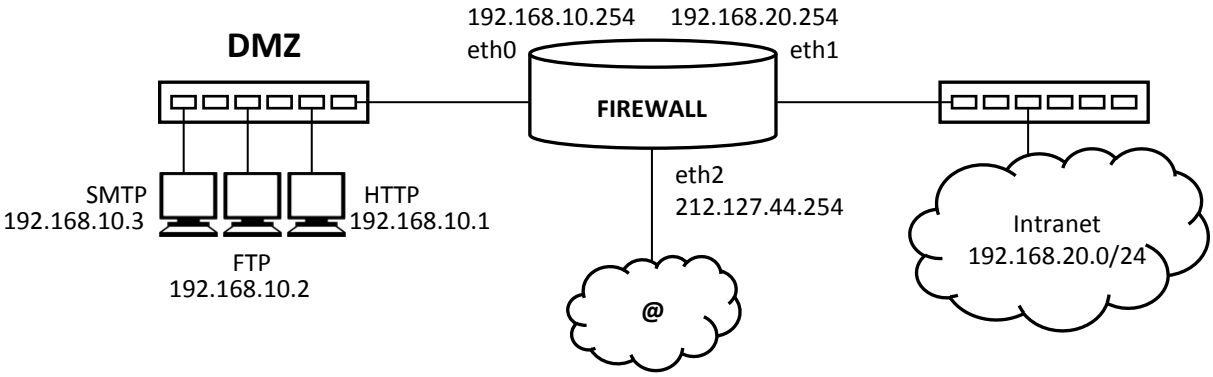
Porque va a recibir muchas más peticiones al servidor web. Cuanto más probable es una cosa, más arriba se pone

5. El comando `iptables -A OUTPUT -p tcp -d www.facebook.com -j DROP` filtra todo el tráfico generado por el propio ordenador dirigido a www.facebook.com. Sin embargo, no es aconsejable utilizar nombres de dominio en las reglas de *iptables*. Utilizando los comandos *host* y *whois* escribir la regla anterior pero especificando el rango de direcciones IP asociadas al dominio www.facebook.com

```
iptables -A OUTPUT -p tcp -m iprange --src-address X.X.X.X - X.X.X.X -j DROP
```

En la siguiente figura se muestra el esquema de un firewall utilizado en muchas organizaciones. Las restricciones de acceso son las siguientes:

- Los servidores en DMZ deben ser accesibles tanto desde la intranet como desde Internet
- A la intranet también se le permite acceder a cualquier servicio de Internet
- Cualquier otro flujo de paquetes debe ser descartado



El siguiente *script* muestra una posible realización del *firewall* (el script está disponible en el portal de la asignatura):

```
 1: echo "1" > /proc/sys/net/ipv4/ip_forward
 2:
 3: IPTABLES="/sbin/iptables"
 4:
 5: iDMZ="eth0"
 6: iLAN="eth1"
 7: iINET="eth2"
 8:
 9: iDMZ_IP="192.168.10.254"
10: iLAN_IP="192.168.20.254"
11: iINET_IP="212.127.44.254"
12:
13: HTTP_IP="192.168.10.1"
14: FTP_IP="192.168.10.2"
15: SMTP_IP="192.168.10.3"
16:
17: $IPTABLES -F
18: $IPTABLES -t nat -F
19: $IPTABLES -t mangle -F
20: $IPTABLES -t raw -F
21:
22: $IPTABLES -P FORWARD DROP
23:
24: $IPTABLES -N allowed
25: $IPTABLES -A allowed -p tcp --tcp-flags SYN,RST,ACK SYN -j ACCEPT
26: $IPTABLES -A allowed -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT
27: $IPTABLES -A allowed -p tcp -j DROP
28:
29: # INPUT chain
30: # nothing for now
31:
32: # OUTPUT chain
33: # nothing for now
34:
35: # FORWARD chain
```

```
36: $IPTABLES -A FORWARD -p tcp -o $iDMZ -d $HTTP_IP --dport 80 -j allowed
37: $IPTABLES -A FORWARD -p tcp -i $iDMZ --sport 80 -j ACCEPT
38:
39: $IPTABLES -A FORWARD -p tcp -o $iDMZ -d $FTP_IP --dport 21 -j allowed
40: $IPTABLES -A FORWARD -p tcp -i $iDMZ --sport 21 -j ACCEPT
41:
42: $IPTABLES -A FORWARD -p tcp -o $iDMZ -d $SMTP_IP --dport 25 -j allowed
43: $IPTABLES -A FORWARD -p tcp -i $iDMZ --sport 25 -j ACCEPT
44:
45: $IPTABLES -A FORWARD -i $iLAN -o $iINET -j ACCEPT
46: $IPTABLES -A FORWARD -i $iINET -o $iLAN -m state --state ESTABLISHED,RELATED -j ACCEPT
47:
48: $IPTABLES -t nat -A PREROUTING -p tcp -i $iINET -d $iINET_IP --dport 80 -j DNAT
    --to-destination $HTTP_IP
49: $IPTABLES -t nat -A PREROUTING -p tcp -i $iINET -d $iINET_IP --dport 21 -j DNAT
    --to-destination $FTP_IP
50: $IPTABLES -t nat -A PREROUTING -p tcp -i $iINET -d $iINET_IP --dport 25 -j DNAT
    --to-destination $SMTP_IP
51:
52: $IPTABLES -t nat -A POSTROUTING -s 192.168.10.0/24 -j SNAT --to-source $iINET_IP
53: $IPTABLES -t nat -A POSTROUTING -s 192.168.20.0/24 -j SNAT --to-source $iINET_IP
```

6. ¿Para qué sirve la instrucción 1?

   Para encender ip_forward, para permitir el enrutamiento

7. ¿Cuál es la diferencia entre las reglas *ACCEPT* y *allowed*?

ACCEPT acepta un paquete y ALLOWED es un conjunto de acciones que aceptará un paquete si y solo sí pertenece a una conexión establecida TCP previamente.
Fíjate que ALLOWED está definida con la opción -N, que indica que se está definiendo una nueva "chain".

8. ¿Cuál es la función de las instrucciones 45 y 46?

Acepta los paquetes que vayan de LAN y INTERNET, acepta los paquetes entrantes de INTERNET a LAN, si la conexión ha sido establecida antes

9. ¿Cuál es la función de las instrucciones 48, 49 y 50?

   Enrutar las conexiones al puerto 80,21,25 a su IP

10. ¿Se podría sustituir las reglas 52 y 53 por `iptables -t nat -A POSTROUTING -o $iINET -j SNAT --to-source $iINET_IP`?

    Sí

11. ¿Cuál es la diferencia entre estas dos instrucciones?

    SNAT: Cuando la dirección IP pública que sustituye a la IP origen es estática (SNAT también significa Static NAT).

    ```
    iptables -t nat -A POSTROUTING -o $iINET -j SNAT --to-source $iINET_IP

    iptables -t nat -A POSTROUTING -o $iINET -j MASQUERADE
    ```

    MASQUERADE: Cuando la dirección IP pública que sustituye a la IP origen es dinámica.

12. Si se sustituyeran las instrucciones 36 y 37 por las que se muestran a continuación ¿Seguiría funcionando el *firewall*? ¿Mejor o peor?

    ```
    $IPTABLES -A FORWARD -i $iDMZ -o $iINET -j ACCEPT
    $IPTABLES -A FORWARD -i $iINET -o $iDMZ -m state --state ESTABLISHED,RELATED -j ACCEPT
    $IPTABLES -A FORWARD -p tcp -o $iDMZ -d $HTTP_IP --dport 80 -j allowed
    ```

    No funciona, no dejará conectarse a los usuarios externos porque no pueden producir una conexión

13. Determinar el contenido de las cadenas INPUT y OUTPUT de tal forma que se puedan hacer conexiones al *firewall* (por ejemplo, para configurarlo) desde la intranet y DMZ pero nunca desde Internet. Además, desde el *firewall* se debe poder establecer cualquier tipo de conexión hacia Internet. Todos los paquetes que no satisfagan ninguna de estas reglas deben ser eliminados (política por defecto DROP)

```
#INPUT chain
$IPTABLES -P INPUT DROP
$IPTABLES -A INPUT -i $iDMZ -j ACCEPT
$IPTABLES -A INPUT -i $iLAN -j ACCEPT
$iptables -A INPUT -i $iINET -m state --state ESTABLISHED, RELATED -j ACCEPT
```

```
#OUTPUT CHAIN
$IPTABLES -P OUTPUT DROP
$iptables -A OUTPUT -O $iDMZ -m state --state ESTABLISHED, RELATED -j ACCEPT
$iptables -A OUTPUT -O $iLAN -m state --state ESTABLISHED, RELATED -j ACCEPT
$IPTABLES -A OUTPUT -O $iINET -j ACCEPT
```

14. Aunque se recomienda no utilizar nunca ordenadores con sistema operativo *Mircrosoft*, es posible que en la Intranet algún usuario no cumpla con dicha recomendación. El inconveniente más destacado que presentan estos sistemas es que para gestionar las redes MS-net (puertos UDP del 135 al 139) se ven obligados a inundar la red con paquetes *broadcast*. Escriba la regla para evitar que dichos paquetes saturen al *firewall*

    ```
    $IPTABLES -A INPUT -p udp --match multiport --dports 135:139 -j DROP
    ```

15. Modifique las cadenas INPUT/OUTPUT de tal forma que el *firewall* pueda aceptar y dar respuesta a paquetes ICMP del tipo 8 (*Echo*) y 11 (*Time Exceeded*)

```
iptables -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT
iptables -A INPUT -p icmp --icmp-type time-exceeded -j ACCEPT
## ** all our server to respond to pings ** ##
iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
```