Seguridad 1. Filtrado de paquetes utilizando Netfilter

Fecha de Entrega: 10/11/2021.

Objetivo: Desarrollar una experiencia práctica en seguridad de redes utilizando un firewall de uso extendido profesionalmente. Conocer las reglas básicas que permiten restringir o permitir el acceso a un servicio determinado en la red. Poner a prueba la aplicación de políticas de seguridad. Conocer buenas prácticas para "suavizar" el efecto de: ataques de denegación de servicio, de envío de spam y finalmente de intentos, por fuerza bruta, de login remoto. Categorías ISO: FCAP**S**.

Bibliografía

- STALLINGS, W. 2011. Capítulo 22: Firewalls. En *Cryptography and Network Security: Principles and Practice* (5th ed). Prentice Hall.
- HERTZOG, R. & MAS, R. 2015. Capítulo 14. Sección 2: "Firewall o el filtrado de paquetes". En *Debian 9. El manual del Administrador de Debian. Debian Stretch from Discovery to Mastery*. Freexian. http://l.github.io/debian-handbook/html/es-ES/sect.firewall-packet-filtering.html
- ZWICKY, E.; COOPER S. & CHAPMAN D. B.. 2000. *Building Internet Firewalls* (2nd ed). O'Reilly Media.
- EVANS, Julia - iptables. https://twitter.com/b0rk/status/1054056111626686465 - Última consulta: 24/10/2018

Experiencia de Laboratorio

1 Teniendo en cuenta las actividades que ha realizado en las prácticas de esta asignatura y de la asignatura Teleinformática y Redes. De las cuatro políticas básicas vistas en la teoría (las "4 P"), ¿cuál considera que es la política de firewall (filtrado de paquetes) aplicada en los equipos Linux? ¿cómo llegó a esa conclusión?

2 Valide su suposición ejecutando `iptables -L` Busque las líneas *Chain INPUT*, *Chain FORWARD* y *Chain OUTPUT* y tome nota de la política vigente.

3 Para el resto de esta experiencia de laboratorio y para evitar modificar su configuración personal se utilizará **una nueva versión** del laboratorio netkit-lab_webserver.tar.gz, por lo que recomendamos volver a descargarlo y reemplazar el laboratorio existente por el actual. Si no tiene instalado netkit siga las instrucciones provistas en: TP1 de TYR.

Para iniciar el laboratorio `cd ~/netkit/netkit-lab_webserver/` y luego `lstart` .

Este es un escenario muy sencillo que tiene un host que funciona como webserver llamado "server" con ip `10.0.0.1` y un host "client" con ip `10.0.0.2` que puede realizarle peticiones.

4 Haga un ping desde "client" hacia "server" y verifique la conectividad. Luego realice un escaneo al servidor mediante `nmap` . Finalmente, verifique las reglas existentes hasta el momento en el firewall de cada equipo con `iptables -L` .

5 Vamos a proceder ahora a configurar el firewall en el equipo "server". Una buena práctica, cuando se quiere configurar una política de seguridad prudente, es denegar primero todo aquello que no esta expresamente permitido. Para lograrlo se deben definir para cada cadena (INPUT, OUTPUT, FORWARD) una política por defecto. La acción recomendada para ello es "DROP". Con esto lo que lograremos es descartar cualquier paquete recibido y tampoco podremos enviar ni reenviar nada.

```
iptables -P INPUT DROP
iptables -P FORWARD DROP
```

```
iptables -P OUTPUT DROP
```

¿Ha llegado usted a la configuración más segura para un servidor web? ¿Que servicio de seguridad no se estaría cumpliendo?

6 Si en algún momento quiere deshacer todos los cambios efectuados en las reglas de iptables ejecute `iptables --flush` . De todas formas, las reglas de iptables se borran al reiniciar los equipos (la política por defecto debe ser cambiada manualmente con iptables -P como se vió en el punto 5)

7 Realice una captura desde su terminal en su equipo real. Para ello y mientras esta situado en el directorio del laboratorio, escriba el comando `vdump A > captura.pncap` . Verifique la configuración realizando un ping desde "client" hacia "server". ¿Llegan?. Finalice en su terminal la captura con `CTRL + C` y abra la misma con `wireshark captura.pncap` . ¿Que puede observar?

8 El "server" es después de todo un webserver ¿No?. Vamos a permitir que entonces pueda recibir peticiones HTTP de puerto 80 de cualquier IP.

```
iptables -A INPUT -i eth0 -p tcp --dport 80 -j ACCEPT
```

Con esta linea le decimos al firewall que vamos a agregar una regla (`-A`) para permitir (`-j ACCEPT`) el ingreso (cadena `INPUT`) de los paquetes que lleguen a la interfaz "eth0" (`-i eth0`), con protocolo "tcp" (`-p tcp`) y al puerto destino 80 (`--dport 80`).

9 Con el comando "wget" desde el host "client" solicite al "server" la página index. Para ello ejecute `wget http://10.0.0.1/` ¿Funcionó? Realice nuevamente una captura con `vdump A > captura.pncap` y verifique que sucede con wireshark.

10 ¡Claramente olvidamos permitir que el servidor pueda enviar paquetes! Para ello agregue la siguiente linea.

```
iptables -A OUTPUT -p tcp --sport 80 -j ACCEPT
```

Vuelva a verificar con wget. ¿La página llegó? ¿Si probamos hacer ping desde el cliente al servidor? ¿Llegan? ¿Deberían hacerlo?

11 Vamos a ser piadosos con el host "client" y vamos a permitir todos los mensajes del protocolo ICMP con este. Para ello en el "server" escribiremos:

```
iptables -A INPUT -s 10.0.0.2 -p icmp -j ACCEPT
iptables -A OUTPUT -d 10.0.0.2 -p icmp -j ACCEPT
```

Con esta linea lo que logramos es agregar una regla en la cadena INPUT para que se acepte todo paquete ICMP de la ip origen del client (`-s 10.0.0.2`) y que también pueda enviarle a esa IP (`-d 10.0.0.2`) paquetes de este protocolo (`-p icmp`). Verifique haciendo ping desde el cliente y desde el servidor.

12 Suponga que instalamos un nuevo servicio en el host "server" que escuchará en el puerto TCP 8000, pero no queremos que el host "client" se conecte a él. Así como tenemos configurado hasta ahora el firewall del servidor ¿Podría conectarse el host client al puerto 8000 del server? Vamos a verificar. Realice una captura con `vdump A > obvioQueNoLlega.pncap` . Luego desde el host "client" ejecute `nc 10.0.0.1 8000` para intentar conectarse al server al puerto 8000. Termine la captura. Observerla con wireshark.

13 Ahora vamos a cambiar las reglas para avisarle al host cliente que rechazamos sus intentos de conexión. Para ello haremos uso de la acción reject.

```
iptables -A INPUT -p tcp -s 10.0.0.2 --dport 8000 -j REJECT
```

Realice una captura nuevamente con el comando `vdump A > reject.pncap` y realice nuevamente desde el host client en intento de conexión `nc 10.0.0.1 8000`. ¿Qué observa en esta nueva captura a diferencia de la captura anterior?

14 Nos arrepentimos, ya no queremos avisarle al host "client" que rechazamos sus peticiones. Para ello agregamos la regla:

```
iptables -A INPUT -s 10.0.0.2 -p tcp --dport 8000 -j DROP
```

15 Verifique nuevamente haciendo desde el host client el intento de `nc 10.0.0.1 8000`. ¿Funcionó? ¿Nos dejó de avisar que nos rechaza la conexión? Claro que no, pero ¿por qué sigue avisando el server que rechaza la conexión si claramente ahora lo debería dropear directamente? Verifique las reglas existentes con `iptables -L`. Como puede verse, la regla está efectivamente definida. Sin embargo lo que sucede es que iptables respeta el orden en que se definen las reglas. Es decir, una vez que una regla hace "match" con el paquete entrante se aplica esa regla y se deja de verificar la tabla. En este caso la primera acción es "REJECT". La solucion en este caso será eliminar las ultimas dos reglas, ya que la política por defecto ya era dropear. Para ello usaremos `-D` para borrar ambas reglas, de la siguiente manera:

```
iptables -D INPUT -p tcp -s 10.0.0.2 --dport 8000 -j REJECT
iptables -D INPUT -s 10.0.0.2 -p tcp --dport 8000 -j DROP
```

14 Utilizando el comando `iptables -L` liste las reglas existentes para ver como ha quedado todo. Esta configuración ¿es *stateful* o *stateless*?

15 Las reglas definidas para aceptar y responder peticiones http al puerto 80, si bien funcionan, podrían ser aún más "seguras". Esto es por ejemplo por que la regla `iptables -A OUTPUT -p tcp --sport 80 -j ACCEPT` permitiría que nuestro servidor envie respuestas a cualquier IP destino sin importar si alguien hizo con nosotros o no una conexión previamente. Nuestro servidor, tal como descubrimos en el punto anterior, tiene un firewall sin estado por lo que ignora las conexiones previas y su relacion con el paquete que está examinando. Vamos a cambiar esto y para ello vamos a borrar las reglas que habiamos definido en el firewall del servidor de la siguiente manera:

```
iptables -D OUTPUT -p tcp --sport 80 -j ACCEPT
iptables -D INPUT -i eth0 -p tcp --dport 80 -j ACCEPT
```

Verificamos con wget desde el host "client" y comprobamos que nuevamente no podemos hacer peticiones. Ahora agregaremos la reglas:

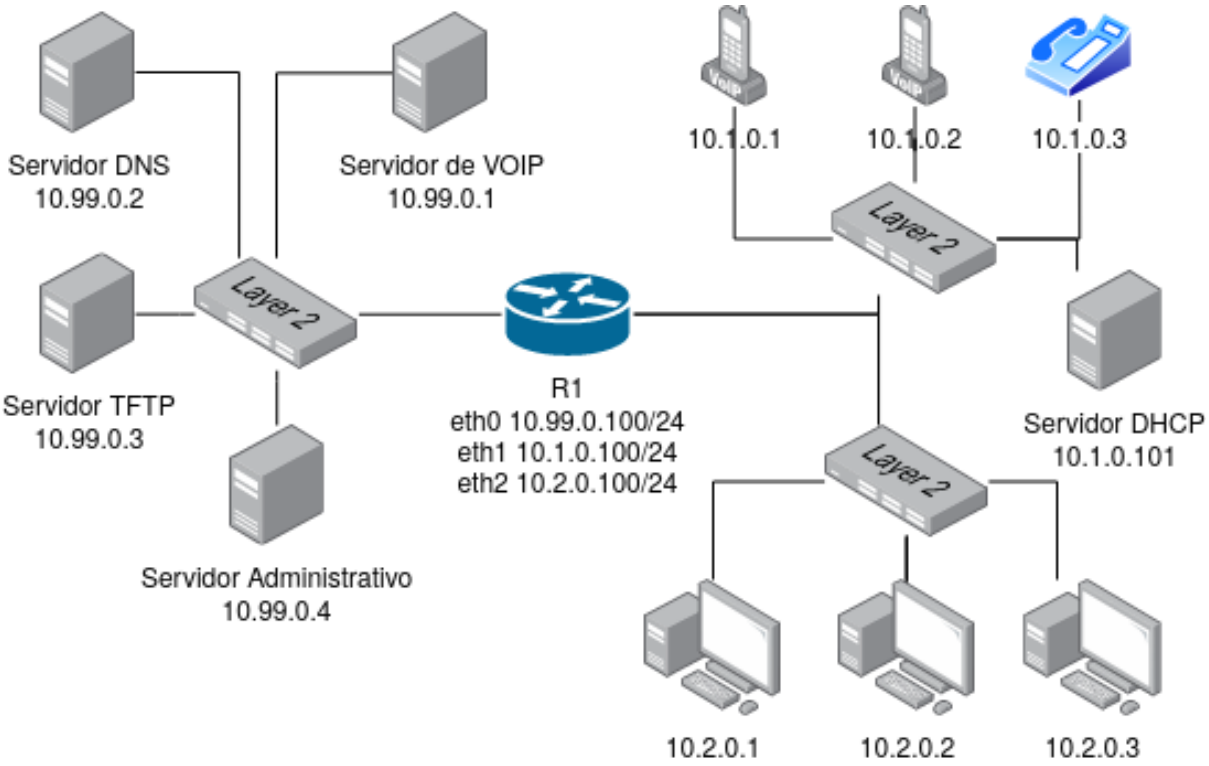
```
iptables -A INPUT -i eth0 -p tcp --dport 80 -m state --state NEW,ESTABLISHED -
j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp --sport 80 -m state --state ESTABLISHED -j ACCEPT
```

Estas reglas permiten que solo se acepten conexiones nuevas y establecidas y que envien paquetes solo a aquellas conexiones que ya fueron establecidas. De esta manera ahora el firewall necesita mantener información de las conexiones establecidas por lo que acabamos de convertirlo en "stateful".

16 Hasta ahora todas las reglas se han aplicado para filtrar los paquetes que se envían o reciben en un mismo host. ¿Qué cambios habría que efectuar en las reglas definidas para que funcionen en un router que opera como firewall?

Trabajo Práctico

1 Analice el siguiente escenario:



Se han dado las siguientes directivas acerca de cómo debe funcionar la red:

a. Los teléfonos deben poder comunicarse con el servidor de VOIP que utiliza SIP como protocolo de señalización. Nos informan además que uno de los teléfonos es de otra marca y que no soporta los mismos CODEC que los demás.

b. Los teléfonos VOIP toman su IP del servidor DHCP.

c. Los teléfonos VOIP toman su configuración del servidor TFTP.

d. Todos los equipos deben ser capaces de hacer consultas al servidor DNS.

e. Las computadoras deben poder comunicarse con el Servidor Administrativo que funciona en el puerto TCP 9896.

f. La computadora con ip 10.2.0.3 debe poder comunicarse con el servidor VOIP a la interfaz administrativa web que funciona con HTTPS.

g. El router tiene una interfaz de configuración ssh a la que debe poder acceder la PC 10.2.0.3. Todos los demás equipos deberían recibir un REJECT si intentan hacerlo.

h. El router debe poder contestar enviar y recibir mensajes ICMP.

Su tarea consiste en configurar, a conciencia, el firewall que se encuentra en el router. Para hacerlo debe completar la siguiente tabla de ACL. Considere que el firewall tiene por defecto la acción drop para todas sus cadenas.

Cadena	IP Origen	IP Destino	Protocolo	Port Origen	Port Destino	Acción	Comentario

Donde el valor de Cadena puede ser `INPUT` (Entrada), `OUTPUT` (Salida), `FORWARD` (Reenvío), entre otras, y el valor de la columna Acción puede ser `ACCEPT` (Permitir), `REJECT` (Rechazar) o `DROP` (Descartar), entre otras.

2 Dada la ACL que usted ha confeccionado, cree un script que configure mediante iptables el firewall de ese router. Para ello puede usar de guía el script de ejemplo que se encuentra en el Anexo I así también como pueden consultar el Anexo II que contiene más información acerca del funcionamiento de iptables. Además puede consultar los snippets presentes en la siguiente web http://www.thegeekstuff.com/2011/06/iptables-rules-examples.

Anexo I

```
#!/bin/sh
#############################################################################
# Script para  implementación de reglas de filtrado en firewall utilizando netfilter
# Interfaces:
# eth0: 172.210.97.200  - Red Interna de la Organización
# eth1: 172.210.96.200  - Red de Acceso público de la Organización
# eth2: 172.210.0.200   - Red de Interconexión a Proveedor de acceso a Internet
#############################################################################

#############################################################################
# Políticas por defecto
# Denegar todo el tráfico que no se habilite de manera específica
#############################################################################

iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP


#############################################################################
# Eliminar todas las reglas existentes
#############################################################################
iptables -F
iptables -X
iptables -Z


#############################################################################
#   Permitir al firewall enviar/recibir tráfico de servicios básicos
#############################################################################
## DNS
iptables -A OUTPUT -d 172.210.96.1 -p udp --dport 53 -j ACCEPT
iptables -A OUTPUT -d 172.210.96.1 -p tcp --dport 53 --syn -j ACCEPT
iptables -A INPUT -s 172.210.96.1 -p udp --sport 53 -j ACCEPT

## NTP
iptables -A OUTPUT -d 172.210.96.1-p udp --dport 123 -j ACCEPT
iptables -A INPUT -s 172.210.96.1 -p udp --sport 123 -j ACCEPT

## SMTP - Correo electrónico saliente
iptables -A OUTPUT -d 172.210.96.3 -p tcp --dport 25 --syn -j ACCEPT

## Aceptar paquetes de conexiones ya establecidas
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

## Aceptar conexiones locales
iptables -A INPUT -i lo -j ACCEPT
```

```
#########################################################################
#  Paquetes en tránsito (tráfico que se rutea)
#########################################################################

## Permitir paquetes pertenecientes a conexiones ya establecidas
iptables -A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT

#################################################################
#  a) Accedo desde red privada a servicios en servidores corporativos
#################################################################
## Acceso a servidor de correo electrónico
iptables -A FORWARD -i eth0 -s 172.210.97.0/24 -p tcp -d 172.210.96.3 \
         -m multiport --dports 25,110,143 --syn -j ACCEPT

## Acceso a servidor dns y ntp
iptables -A FORWARD -i eth0 -s 172.210.97.0/24 -d 172.210.96.1 -p udp --dport 53 -
j ACCEPT
iptables -A FORWARD -i eth0 -s 172.210.97.0/24 -d 172.210.96.1 -p tcp --dport 53 \
         --syn -j ACCEPT
iptables -A FORWARD -i eth0 -s 172.210.97.0/24 -d 172.210.96.1 -p udp --dport 123 -
j ACCEPT

## Acceso a servidor web
iptables -A FORWARD -i eth0 -s 172.210.97.0/24 -d 172.210.96.2 -p tcp \
         -m multiport --dports 80,443 --syn -j ACCEPT

#################################################################
#  b) Desde Internet es posible acceder al servidor web de la empresa.
#################################################################
iptables -A FORWARD -i eth2 -d 172.210.96.2 -p tcp -m multiport --dports 80,443 \
         --syn -j ACCEPT

#################################################################
# c) Desde Internet es posible enviar correo electrónico al servidor de correo
#    corporativo.
#################################################################
iptables -A FORWARD -i eth2 -d 172.210.96.3 -p tcp  --dport 25 --syn -j ACCEPT

#################################################################
# d) Desde Internet es posible realizar consultas DNS al servidor de nombres
#    de dominio de la organización.
#################################################################
iptables -A FORWARD -i eth2 -d 172.210.96.1 -p udp --dport 53 -j ACCEPT
iptables -A FORWARD -i eth2 -d 172.210.96.1 -p tcp --dport 53 --syn -j ACCEPT

#################################################################
# e) El servidor de nombres de dominio debe sincronizarse con otros servidores
#    NTP accesibles a través de Internet.
```

```
#####################################################################
iptables -A FORWARD -i eth1 -s 172.210.96.1 -p udp --dport 123 -j ACCEPT
```

Anexo II - Algo más de teoría

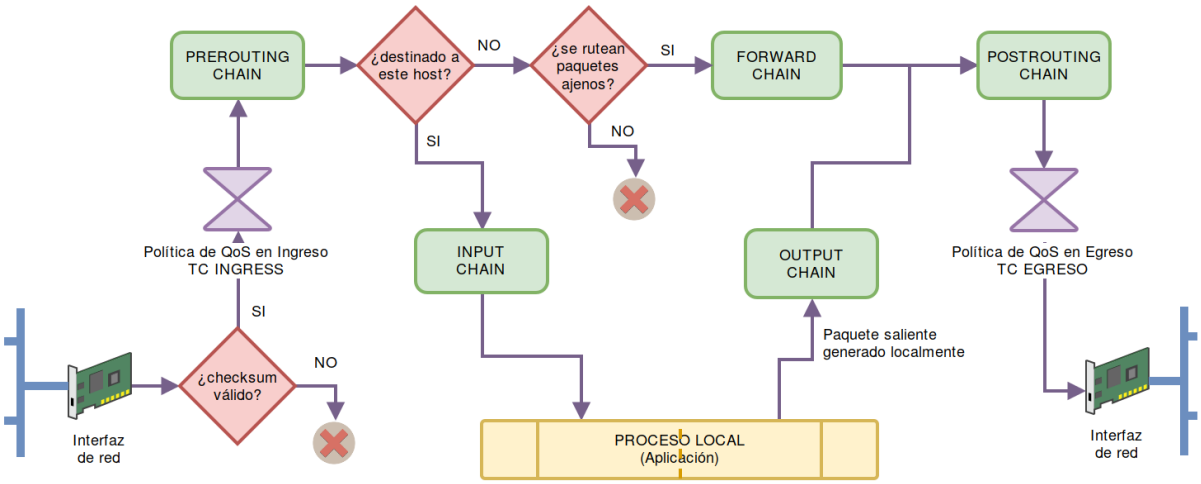


Figura 1: Esquema de funcionamiento de las interfaces de red de un equipo linux

Según la página de manual de *iptables*:

Iptables e **ip6tables** se utilizan para configurar, mantener e inspeccionar las tablas de reglas de filtro de paquetes IPv4 e IPv6 en el kernel de Linux. Se pueden definir varias **tablas** (TABLES) diferentes. Cada tabla contiene una serie de **cadenas** (CHAINS) incorporadas y también puede contener cadenas definidas por el usuario.

Cada cadena es una lista de **reglas** (RULES) que puede coincidir con un conjunto de paquetes. Cada regla especifica qué [acción] hacer con un paquete que coincide. Esto se llama **acción** u **objetivo** (TARGET), que puede ser un salto a una cadena definida por el usuario en la misma tabla.

Una regla de firewall especifica los criterios para un paquete y una acción. Si el paquete *no* coincide, se examina la siguiente regla de la cadena; si coincide, se ejecuta lo indicado en valor de acción, que puede ser un salto a otra cadena de reglas definida por el usuario, o uno de los objetivos descritos en `iptables-extensions` o bien (más frecuentemente) uno de los valores especiales `ACCEPT`, `DROP`, `RETURN`, `REJECT` o `LOG`.

`ACCEPT` significa dejar pasar el paquete. `DROP` significa descartar el paquete. `RETURN` significa dejar de leer esta cadena y reanudar en la siguiente regla de la cadena anterior. Si se llega al final de una cadena incorporada [sin que coincida ninguna regla], la acción que se toma es la especificada por la política por defecto de la cadena. `REJECT` descarta el paquete y remite al remitente un mensaje ICMP (icmp-port-unreachable) u otro definido por el usuario. `LOG` deja pasar el paquete pero registra su paso en el registro de actividad del kernel (`dmesg`). La lista completa de acciones disponibles puede consultarse en la sección TARGETS del manual de iptables y en la misma sección del manual de iptables-extensions.

Finalmente, las cadenas comúnmente utilizadas son:

- `INPUT` que almacena las reglas que se aplicarán sobre paquetes **destinados al firewall**. Es decir, la dirección IP *destino* que figura en el paquete es la del firewall.
- `OUTPUT`, que almacena las reglas que se aplicarán sobre paquetes **originados en el firewall**. Es decir, la dirección IP *origen* que figura en el paquete es la del firewall.
- `FORWARD`, que almacena las reglas que se aplicarán sobre paquetes **reenviados por el router/firewall**. Es decir, ninguna de las direcciones IP del paquete es la del firewall.

Las reglas se agregan/eliminan/listan mediante los comandos `iptables` e `ip6tables`, con la siguiente sintaxis:

```
iptables {-A|-D} CADENA REGLA-DE-MATCHEO -j ACCION
```

por ejemplo, para agregar una regla que rechace todos los paquetes IP destinados al puerto 80 del firewall, el comando a ejecutar es el siguiente:

```
iptables -A INPUT --destination IP-DEL-FIREWALL --dport 80 -j REJECT
```

El primer argumento indica el comando a ejecutar:

- `-L -v -n` listar las reglas en una cadena (verbose y sin resolver DNS)
- `-A` agregar la regla indicada a continuación
- `-D` borrar la regla indicada a continuación
- `-F` vaciar (limpiar) toda la lista de reglas de una cadena
- `-P` establecer la política por defecto de una cadena
- `-Z` reiniciar los contadores de paquetes de una cadena