SISTEMA OPERATIVO GNU/LINUX AVANZADO II

JOSE ARRIETA NARVAEZ
GUSTAVO CARO
JESUS GARCIA
NILXON VUELVAS

TALLER

FIREWALL

ING. LUIS GARCIAS

UNIVERSIDA DE CORDOBA
FACULTAD DE CIENCIAS BASICAS E INGENIERIAS
DEPARTAMENTO DE INGENIERIA DE SISTEMAS Y TELECOMUNICACIONES
MONTERIA-COLOMBIA
2010

1. Introducción

   Termilogia:

Paquete: estructura de datos con información que viaja a través de una red.

- ✔ El filtrado de paquetes es una herramienta para monitorear y controlar todas las conexiones que pasan a traves de una interfase de red.
- ✔ El filtrado de paquetes se utiliza para determinar (en base al encabezado del mismo, en la mayoría de los casos) qué hacer con un paquete de red.
- ✔ El filtrado de paquetes es una herramienta para monitorear y controlar todas las conexiones que pasan a traves de una interfase de red.
- ✔ El filtrado de paquetes se utiliza para determinar (en base al encabezado del mismo, en la mayoría de los casos) qué hacer con un paquete de red.

Conceptos
Reglas: Las reglas son órdenes escritas que intentan ser lo más detallistas posibles. Estas son las que determinan qué hacer con un paquete en base a su encabezado (de dónde viene, a dónde va, etc., .etc)
Cadenas: Las cadenas contiene reglas
Tablas: Las tablas contienen cadenas

Tablas y Cadenas
Tablas: Existen 3 (tres)
filter (se encarga del filtrado)
nat (se encarga de cambiar el encabezado del paquete)
Mangle (acá se pueden cambiar otros campos de los paquetes)
Cadenas: Cada tabla contiene cadenas propias y es posible crear cadenas personalizadas

Orden de verificación
Las reglas se van verificando una a una de arriba hacia abajo.
A diferencia de ipchains, en iptables no todos los paquetes atraviesan input.
Las tablas no son verificadas en orden.
El orden es determinado en base al origen y destino de cada paquete en particular.

¿Qué es un firewall?
Un firewall es un dispositivo que filtra el tráfico entre redes, como mínimo dos.
Puede ser un dispositivo físico o un software sobre un sistema operativo.
En general debemos verlo como una caja con DOS o mas interfaces de red en la que se establecen una reglas de filtrado con las que se decide si una conexión determinada puede establecerse o no.
Incluso puede ir más allá y realizar modificaciones sobre las comunicaciones, como el NAT.
Hoy en día es un hardware especifico con un sistema operativo o una IOS que filtra el tráfico TCP/UDP/ICMP/../IP y decide si un paquete pasa, se modifica, se convierte o se descarta.
Para que un firewall entre redes funcione como tal debe tener al menos dos tarjetas de red.

Conjunto de reglas en las que se examina el origen y destino de los paquetes del protocolo tcp/ip. En cuanto a protocolos es probable que sean capaces de filtrar muchos tipos de ellos, no solo los tcp, también los udp, los icmp, los gre y otros protocolos vinculados a vpns.
Bien, pues casi sin analogías, eso es un firewall en términos de sistemas computacionales.
Un cortafuegos, es un sistema informático, simple o compuesto que actúa como punto de conexión segura entre otros dos o más sistemas informáticos.

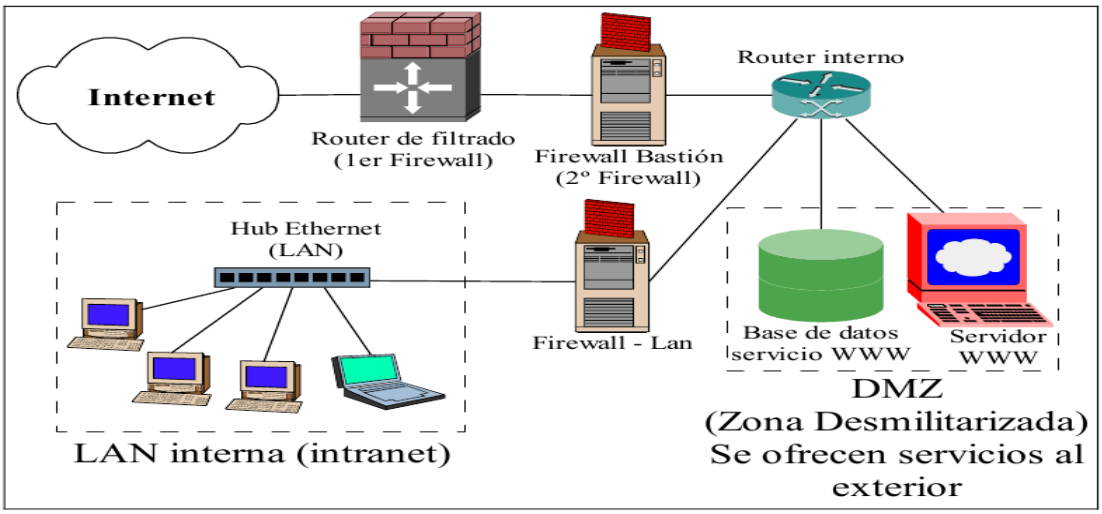


Figura 1 Sistema complejo con diversos Cortafuegos

Descendiendo en la abstracción, un cortafuego se sitúa entre dos o más redes con la intención de hacer cumplir unas determinadas directivas de seguridad sobre la comunicación entre ellas. Por ello, un cortafuego, puede ser desde un simple router, un PC2 antiguo o una subred de servidores SOLARIS.

2. Cortafuegos de filtrado de paquetes, IPFW

El documento presente, no pretende ser un estudio sobre la infinidad de familias y variedades de los firewalls, seremos menos abstractos, y nos centraremos en el estudio de un modelo de cortafuegos mas concreto, la familia de los firewalls de filtrado de paquetes sobre la pila de protocolos TCP/IP, los llamados IPFW.
Los IPFW funcionan como indican las dos primeras letras sobre paquetes IP, es decir, en el nivel de transporte y red de TCP/IP.

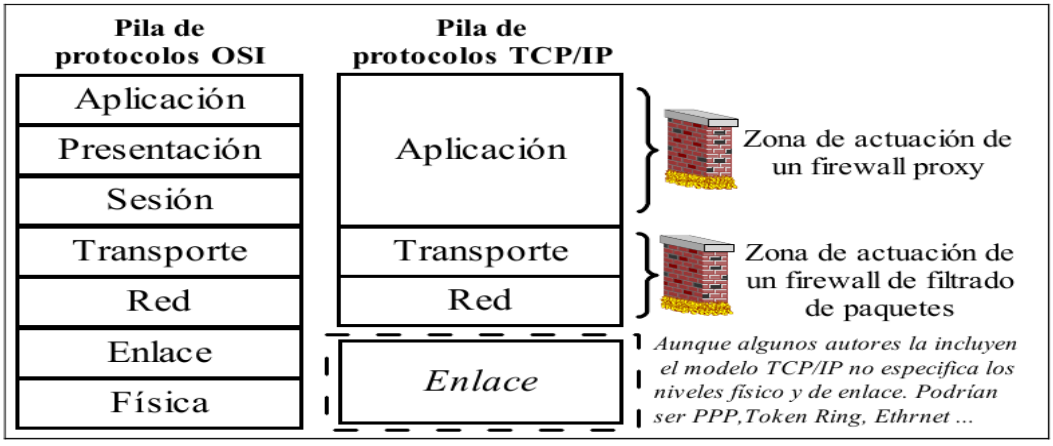


Figura 2 Niveles ISO/OSI vs TCP/IP

Los cortafuegos de filtrado de paquetes IP, suelen implementarse dentro del sistema operativo y funcionan en las capas de transporte y red, trabajando sobre la información de las cabeceras de los paquetes IP, es decir, que no analizarán el área de datos3, sino que únicamente utilizan la información que puede obtenerse de una cabecera IP.
Como ya habíamos dicho, habitualmente, un cortafuego se sitúa entre dos o más redes, lo que implica que tiene al menos dos interfaces de red. A pesar de que el cortafuegos separa dos redes cualquiera entre si, lo habitual, es que separe redes propietarias distintos. Es decir, aunque pueden utilizarse cortafuegos dentro de una misma red, filtrando las comunicaciones entres las distintas subredes internas, lo habitual, es que el cortafuego forme parte de una red propia4 y sea él quien vigile las comunicaciones con otra red o redes ajenas en las que no se confía5, por ejemplo y sobre todo Internet.

3. Introducción a iptables.

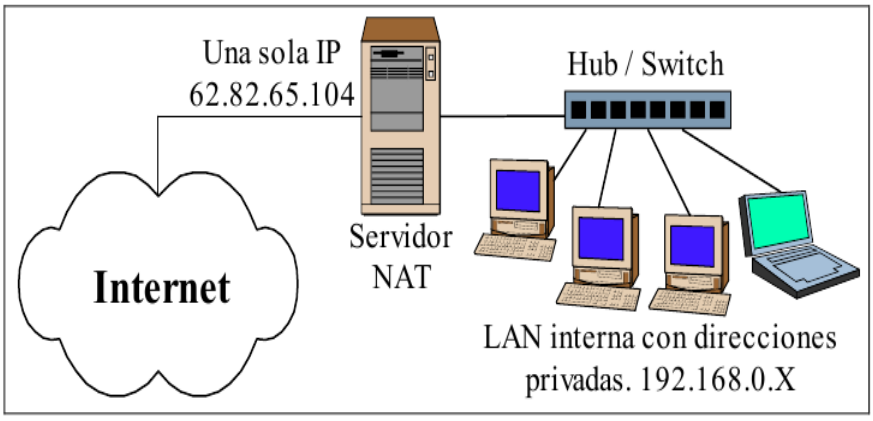
Bien, como ya se ha comentado, el objetivo del documento, es centrarse en el estudio de una herramienta concreta. Concretamente, el sistema a estudiar, será la herramienta de cortafuegos iptables sobre un sistema operativo Linux.
¿Por qué Linux? Como todo sistema operativo UNIX tiene entre sus objetivos la seguridad, además es FREEWARE.
De cualquier forma, en lo que a la configuración de IPTables concierne, las diferencias son prácticamente inexistentes entre los distintos Linux. Lo único necesario es que la distribución Linux elegida cuente con una versión de Kernel superior a la 2.4.
Iptables es como se conoce al módulo Netfilter, la herramienta estándar actual de cortafuegos bajo el sistema operativo estándar Linux de cortafuegos.

3.1. Un poco de historia.

La primera pila IP para Linux la desarrolló Ross Biro (NET-1). Al mismo tiempo, Orest Zborowski y Laurence Culthane trabajaban en la API sockets y en los controladores SLIP. Sin embargo, la verdadera integración de Linux a la red llegó de la mano de Alan Cox con NET-2 y NET-3. Esta última, es la actual pila de protocolos TCP/IP en la que además de Cox participaron muchos otros como Donnald Becker, etc…
El sistema operativo Linux, ha contado con herramientas de filtrado de paquetes (IPFW) incorporadas en su núcleo desde la versión del kernel 1.1.
Esta primera versión con filtrado, contaba con una adaptación de la herramienta ipfw del sistema operativo BSD llevada a cabo Alan Cox por el año 94.
El holandés Jos Vos junto a otras personas, mejoró el sistema de filtrado para las series 2.0 del kernel, e introdujo la utilidad de configuración ipfwadm.
A mediados de 1998, de la mano de Michael Neuling y Rusty Russell aparece la herramienta ipchains, incorporada en los kernels de la serie 2.2 y que todavía hoy es utilizada en gran parte de los sistemas Linux, aunque sólo se asegurará su compatibilidad en el núcleo hasta el año 2003. ¿Por qué solo hasta el 2003?
La respuesta llega a mediados de 1999. Cuando de nuevo Rusty Russell aparece en escena con una nueva herramienta de filtrado iptables. Como lo fue ipchains sobre ipfw, iptables es una modificación que permite la construcción de reglas más precisas y un mejor aprovechamiento de los recursos.

3.2. Novedades de iptables. NAT.

Además de realizar un mejor aprovechamiento de los recursos del sistema, la principal novedad del módulo netfilter-iptables, es la integración de las herramientas de filtrado (el cortafuegos propiamente dicho), de NAT y de manipulación (MANGLING). Aunque lo trataremos con más profundidad posteriormente, creo que debemos deternos un momento para explicar lo que es NAT. NAT es el acrónimo de Network Addres Translation. Lo que hace NAT básicamente es alterar las cabeceras de los paquetes IP, (principalmente las direcciones) y mantener un "registro de entrada y salida" de los paquetes modificados, para poder alterar los paquetes respuesta de igual forma. Las aplicaciones de NAT son muchísimas, aunque actualmente, la aplicación más conocida del NAT, es lo que se conoce como enmascaramiento o masquerading. El enmascaramiento, se utiliza principalmente para dos cosas:
- La conexión de varios equipos a través de una sola dirección IP. Como ejemplos, las pequeñas LAN domésticas, un CyberCafé, y en general cualquier red con más equipos que IPs. En lugar de instalar un servidor proxy y configurar las distintas aplicaciones para que hagan uso del mismo, se instala NAT, y no hay necesidad de configurar las aplicaciones, ya que al trabajar en un nivel más bajo de la pila, resulta totalmente transparente.
No debe confundirse el NAT con un "proxy", aunque tengan aplicaciones parecidas.
NAT no es un proxy, los proxies trabajan en un nivel superior de la pila de protocolos, bien en el nivel de TCP/UDP o incluso en el nivel de aplicación, lo que implica que los clientes deben configurarse para hacer uso del servicio. Por el contrario, NAT, al igual que el filtrado, trabaja en un nivel más bajo, a nivel IP, por lo que es "transparente". ¿Por qué se confunden muchas veces el NAT y un proxy? La respuesta puede deberse a que tanto el uso de un proxy como el uso de NAT se emplean actualmente11 para "compartir una conexión a Internet".



-El balanceo de carga. Para explicarlo, lo mejor es ofrecer un ejemplo. Imaginemos que tenemos una máquina en la que corre un servidor HTTP, una de las páginas que ofrece el servidor, tiene un número de visitas diarias tremendamente elevado, (por ejemplo www.millonesdevisitas.zzz). El servidor, está conectado a Internet con un ancho de banda suficientemente importante y está disponible para todo el mundo. El problema, es que dada la sobrecarga de peticiones a las que se ve sometido el servidor, la máquina se sobrecarga y las las páginas se sirven con mucha lentitud. ¿Cómo resolverlo? O compramos un servidor

más potente, o utilizamos NAT. La IP que tenía la máquina servidor y que estaba asociada a la URL www.millonesdevisitas.zzz, se le asignaría a una máquina que actuaría de radware (balanceador) y repartiría la carga entre varias máquinas (que formarían parte por ejemplo de una LAN Ethernet) cuyos servidores HTTP estuviesen configurados de forma similar (sirviesen todos las mismas páginas). La carga de los servidores quedaría repartida y además se facilitaría la redundancia de existencia, los servidores HTTP que alojan webs importantes o con un gran número de visitas utilizan este sistema de balanceo para que el tiempo de respuesta de las peticiones sea bajo y mantener el servicio si cae alguno de los servidores.

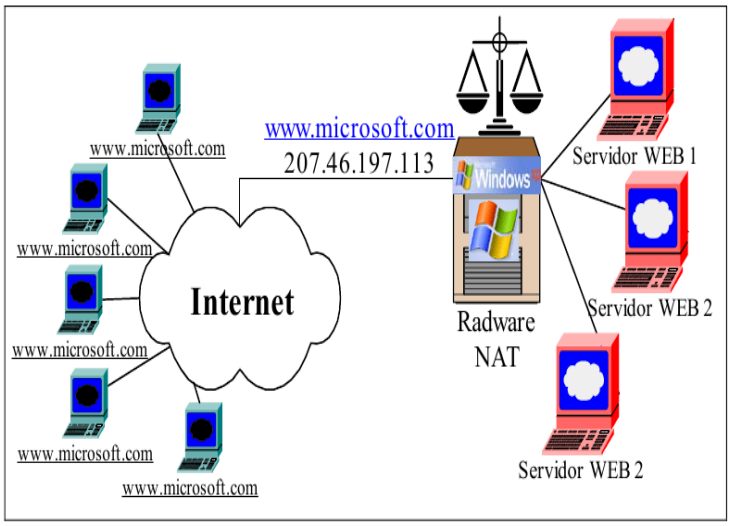


Figura 4 Ejemplo12 de un sistema de balanceo de carga HTTP ☺

El filtrado de paquetes y NAT están ampliamente ligados, ya que comos ya hemos ambos servicios constituyen el módulo Netfilter y se configuran haciendo uso de la herramienta iptables. Antes de iptables, la herramienta Linux utilizada para configurar y ofrecer servicios NAT era ipmasqadm. A diferencia de ipchains / iptables cuyo manejo es conceptualmente similar, entre ipmasqadm e iptables, existen diferencias de uso.

4. Instalación de iptables.

Lo primero que debemos hacer para instalar iptables, es conseguirlo. Podemos descargar iptables de http://www.netfilter.org/. En cualquier distribución actual se instala por defecto y el kernel “precompilado” soporta la configuración, a pesar de eso, vamos a describir brevemente el proceso necesario para una instalación.

4.1. Parámetros del kernel.

Como ya hemos dicho, para poder utilizar iptables, necesitamos un kernel superior a la versión 2.3.15. Para configurarlo correctamente deberemos recompilar el mismo (make config) e incluir distintas opciones dentro del kernel, o bien habilitar la posibilidad de que puedan cargarse posteriormente como módulos. En la figura ofrecida a continuación, están descritas brevemente las opciones más importantes:

CONFIG_PACKET: Permite a ciertos programas trabajar directamente con la interfaz de red.
CONFIG_NETFILTER: Opción necesaria para utilizar la máquina como cortafuegos y/o router.
CONFIG_IP_NF_FILTER: Habilita el uso de la tabla FILTER.
CONFIG_IP_NF_NAT: Habilita el uso de la tabla de NAT.
CONFIG_IP_NF_IPTABLES: Opción necesaria para utilizar iptables.
CONFIG_IP_NF_CONNTRACK: Opción necesaria para usar NAT y enmascaramiento (seguimiento de conexiones).
CONFIG_IP_NF_TARGET_MASQUERADE: opción necesaria para trabajar con NAT cuando la IP de conexión a Internet es dinámica.
CONFIG_IP_NF_FTP: Opción necesaria para poder hacer seguimiento de conexiones a servidores ftp a través del cortafuegos.
CONFIG_IP_NF_MATCH_LIMIT: Esta opción permite limitar en el tiempo el número de paquetes que casan con una cierta regla. Se utiliza para limitar ataques DOS por sobrecarga.
CONFIG_IP_NF_MATCH_MAC: Permite el uso de direcciones MAC (Ethernet) en las reglas de filtrado.
CONFIG_IP_NF_MATCH_STATE: Es una de las principales novedades respecto a ipchains, pues permite hacer filtrado a partir del estado de una conexión TCP (por ejemplo ESTABLISHED…).
CONFIG_IP_NF_MATCH_OWNER: Es un módulo recientemente incorporado a iptables, con el podemos filtrar paquetes en función del usuario que es dueño (UID).
CONFIG_IP_NF_TARGET_LOG: Permite registrar en ficheros .log el disparo de las reglas. Se utiliza para observar situaciones extrañas en la configuración o posibles ataques.

Figura 5 Distintas opciones del kernel.

Hay más opciones además de las citadas, por ejemplo, CONFIGIG_IP_NF_COMPAT_ IPCHAINS, habilita compatibilidad con reglas de ipchains. Por ello, se dice que la herramienta iptables es extensible, es decir, que tanto iptables como el kernel, están preparados para soportar la definición de nuevos criterios de evaluación en las condiciones de las reglas o nuevas acciones a efectuar con los paquetes. Estas extensiones, pueden ser implementadas por nosotros, o bien podemos descargar alguna extensión pública y utilizarla. Lo normal es optar por la segunda opción, aunque, los gurús comentan que los fuentes de iptables son bastante claros.

4.2. Instalando iptables.

Lo primero que debemos hacer tras descargar el paquete iptables, En primer lugar instalamos el paquete iptables:

```
#apt-get install -test iptables
```

Esto nos activará el servicio por defecto. Solo queda implementar el script de iptables con nuestras reglas y hacer que se carguen al inicio.
El script que yo tengo, sin entrar en detalles viene a ser algo asi, muy sencillo:

```
--------------------
#!/bin/bash
#-s Especifica una dirección de origen
#-d Especifica una dirección de destino
#-p Especifica un prototocolo
#-i Especifica un interface de entrada
#-o Especifica un interface de salida
#-j Especifica la acción a ejecutar sobre el paquete
#--sport Puerto de origen
#--dport Puerto de destino
```

```
#Borrar todas las reglas
iptables -F
#Politica general.Cerramos todo.Dejamos entrar y salir lo solicitado
iptables -P INPUT DROP
iptables -P OUTPUT ACCEPT
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
###OTRAS PROTECCIONES####
# Quitamos los pings.
/bin/echo "1" > /proc/sys/net/ipv4/icmp_echo_ignore_all
# No respondemos a los broadcast.
/bin/echo "1" > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
# Para evitar el spoofing nos aseguramos de que la dirección
# origen del paquete viene del sitio correcto.
for interface in /proc/sys/net/ipv4/conf/*/rp_filter; do
/bin/echo "1" > ${interface}
done
# Los ICMPs redirigidos que pueden alterar la tabla de rutas.
for interface in /proc/sys/net/ipv4/conf/*/accept_redirects; do
/bin/echo "0" > ${interface}
done
# No guardamos registros de los marcianos.
/bin/echo "1" > /proc/sys/net/ipv4/conf/all/log_martians
# Asegurar, aunque no tenga soporte el nucleo, q no hay forward.
/bin/echo "0" > /proc/sys/net/ipv4/ip_forward
###Reglas de los puertos####
# Permitimos que se conecten a nuestro servidor web.
#iptables -A INPUT -m state --state NEW -p TCP --dport 80 -j ACCEPT
#Abrimos ssh a la red.
#iptables -A INPUT -s 172.26.0.3 -p TCP --dport 22 -j ACCEPT
#iptables -A INPUT -s 172.26.0.4 -p TCP --dport 22 -j ACCEPT
#iptables -A INPUT -s 172.26.0.5 -p TCP --dport 22 -j ACCEPT
#iptables -A INPUT -p TCP --dport 22 -j ACCEPT
# Permitimos la comunicación con el servidor dns
iptables -A INPUT -p UDP --dport 53 -j ACCEPT
iptables -A INPUT -p TCP --dport 53 -j ACCEPT
#Permitimos uso de ftp.
#iptables -A INPUT -p TCP --dport 21 -j ACCEPT
#Permitimos acceso pop3.
#iptables -A INPUT -p TCP --dport 110 -j ACCEPT
# Permitimos uso de smtp
#iptables -A INPUT -p TCP --dport 25 -j ACCEPT
#Permitimos acceso imap.
#iptables -A INPUT -p TCP --dport 143 -j ACCEPT
#iptables -A INPUT -p UDP --dport 143 -j ACCEPT
#Permitimos todo el trafico de la LAN
iptables -A INPUT -s 172.26.0.2 -j ACCEPT
iptables -A INPUT -s 172.26.0.4 -j ACCEPT
iptables -A INPUT -s 172.26.0.5 -j ACCEPT
#Dejamos a localhost, para mysql, etc..
iptables -A INPUT -i lo -j ACCEPT
```

Logicamente las lineas van juntas. Solo se aplican las lineas no comentadas con # , las otras son para mi servidor.

el fichero se puede llamar por ejemplo firewall.sh , debemos darle permisos de ejecución:

#chmod +x firewall.sh

Ahora si aplicamos el script de este modo:

#sh firewall.sh

podemos ver la salida de iptables haciendo:

#iptables -L
que viene a ser algo asi:
---
root@platas:/home/herje # iptables -L
Chain INPUT (policy DROP)
target prot opt source destination
ACCEPT all -- anywhere anywhere state RELATED,ESTABLISHED
ACCEPT udp -- anywhere anywhere udp dpt:domain
ACCEPT tcp -- anywhere anywhere tcp dpt:domain
ACCEPT all -- glorioso anywhere
ACCEPT all -- minime anywhere
ACCEPT all -- placebo anywhere
ACCEPT all -- anywhere anywhere
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
ACCEPT all -- anywhere anywhere state RELATED,ESTABLISHED
root@platas:/home/herje #

Solo falta hacer que las reglas se carguen al inicio, para ello copiamos el script en /etc/init.d y lo ponemos por default:
#cp firewall.sh /etc/init.d/
#update-rc.d firewall.sh defaults

Y listo con esto tenemos nuestro firewall funcionando.

5. Estructura y funcionamiento de iptables.

Comentamos al principio del documento, que el módulo netfilter, integraba tres posibilidades en el manejo de los paquetes, cada una de esas posibilidades, se corresponde con una tabla donde se aplican las reglas. Con la opción iptables –t tabla, especificamos la tabla sobre la que queremos trabajar. Estas tablas son filter, nat y mangling. Veamos que podemos hacer sobre cada una de ellas:

-nat: La tabla nat se utiliza para configurar el protocolo de Network Address Translation. Cuando un flujo de paquetes (una conexión TCP) atraviesa la tabla, el primer paquete es admitido, el resto, son automáticamente identificados como parte del flujo de ese primer paquete y de manera automática se llevan a cabo sobre ellos las operaciones NAT o de enmascaramiento. Esta es la razón por la cual, no se lleva a cabo ningún tipo de filtrado en esta tabla. La tabla de nat tiene tres chains o cadenas sobre las que podemos añadir reglas. La cadena PREROUTING se utiliza para alterar los paquetes tan pronto llegan al cortafuegos (DNAT o NAT del destino). La cadena OUTPUT, se utiliza para alterar los paquetes generados localmente en el cortafuegos, antes de tomar ninguna decisión de enrutado. Finalmente tenemos la cadena POSTROUTING para alterar los paquetes que acaban de dejar el cortafuegos (SNAT o NAT en el origen).

- mangle: La tabla de mangling o “manipulación”, permite manipular otros elementos de los paquetes, como el TTL, el TOS, etc…, ha excepción del NAT, que se realiza en la otra tabla. La funcionalidad de esta tabla está en expansión y aunque potencialmente puede ser muy valiosa, no tiene demasiada utilidad (salvo a hackers).

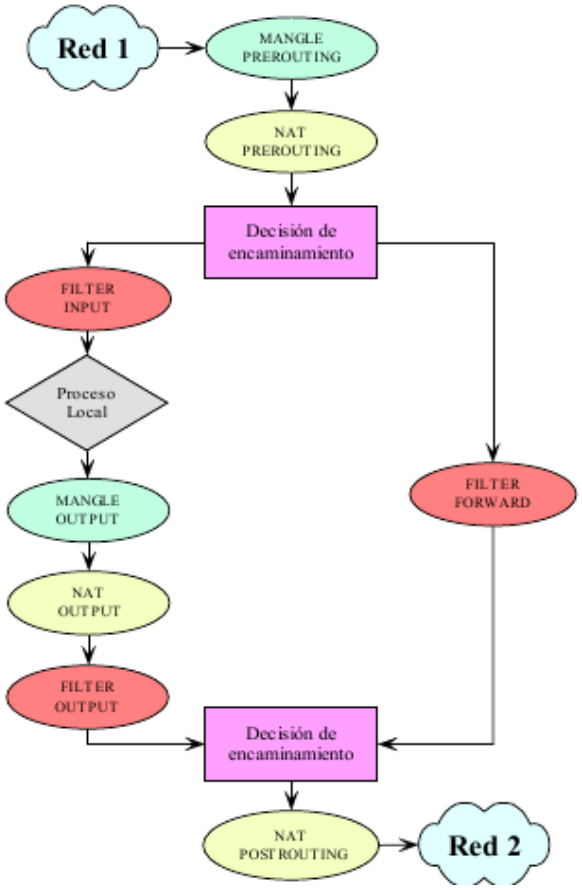
Consta de dos cadenas, PREROUTING y OUTPUT.

- filter: Esta es la reina de la casa. En la tabla filter, se llevan a cabo la funcionalidad principal

de iptables, el filtrado de paquetes. Como las anteriores, consta de varias chains predefinidas, en este caso INPUT, FORWARD y OUTPUT. La primera hace referencia a los paquetes entrantes cuyo destino es el propio cortafuegos. La segunda se emplea para decidir que hacer con los paquetes que llegan al cortafuegos y tienencomo destino otro host, así podemos decidir si encaminarlos o no. Por último, la cadena OUTPUT se utiliza para filtrar paquetes generados en el propio host con destinos externos.
Cuando un paquete entra en el cortafuegos, lo hace a través de alguna interfaz (tarjeta de red, MODEM…). El paquete se dirige al Kernel, entrando en las distintas cadenas de las tablas sólo si procede. En la figura que ofrecemos posteriormente, puede observarse el esquema general del procesado de paquetes en iptables. En la tabla siguiente podemos ver también ejemplos de las desventuras de los paquetes en su tránsito por el módulo netfilter del kernel.

Paso	Tabla	Cadena	Comentario
1			En el aire, por ejemplo Internet
2			Llega a la interfaz de red, por ejemplo eth0
3	MANGLE	PREROUTING	Si casa con alguna de las reglas de *mangling*, se actúa según indique la acción.
4	NAT	PREROUTING	Si casa con alguna de las reglas de *NAT*, se actúa según indique la acción.
5			Decisión de enrutado
6	FILTER	INPUT	Se procede a realizar el filtrado del tráfico con destino local.
7			Aplicación local (cliente o servidor)

Tabla 1 Pasos en el Kernel de un paquete externo con destino en el propio firewall.



Podemos comentar algunas curiosidades del esquema anterior. Partamos del supuesto de una conexión compartida a Internet. Supongamos que la RED1 es una red local con Ips privadas, y la RED2 es Internet, para que se accediese a servicios externos de Internet desde la RED1 (por ejemplo WWW), deberemos utilizar NAT, ¿Cómo funciona? Un paquete con origen en 192.168.0.2 (IP de un PC de la LAN), tiene como destino el puerto 80 de la IP 158.42.180.64 (www.redes.upv.es). El paquete llega a la interfaz interna del cortafuegos, pasa por la tabla de mangling sin recibir ninguna modificación, atraviesa la tabla de NAT prerouting sin modificarse, se observa que el destino es externo, por lo que se encamina a través de la tabla filter y de la cadena FORWARD, suponiendo que atraviesa todas sus reglas (es decir, que no se impide que los usuarios de la LAN accedan a servidores HTTP externos), se llega de nuevo a la decisión de encaminamiento y tras ello, el paquete atraviesa la tabla NAT y la cadena POSTROUTING, es aquí donde se modifica la dirección origen del paquete para que coincida con la IP pública del cortafuegos, a partir de este momento, los paquetes (flujo) de esa conexión,van marcados por la tabla NAT que lo gestiona de forma autónoma.

6. El comando iptables.

En este apartado, vamos a presentar el uso del comando iptables, que es la herramienta para crear las reglas de nuestro cortafuegos. A pesar de la descripción que vamos a ofrecer, como ocurre siempre en el mundo Linux, no hay que olvidar nuca la utilidad del comando man, si se desea un conocimiento exhaustivo de todas las opciones de la herramienta iptables lo mejor es consultar el manual. No sólo eso, además en el mundo Linux, funcionan las maravillosas HOWTOs, la HOWTO de iptables ha sido llevada a cabo por el propio Rusty

Russell, lo que es una garantía de su calidad. Así que, ante cualquier duda que surja, lo mejor es acudir a esas fuentes.

iptables -[ADC] chain rule-specification [options]
iptables -[RI] chain rulenum rule-specification [options]
iptables -D chain rulenum [options]
iptables -[LFZ] [chain] [options]
iptables -[NX] chain
iptables -P chain target [options]
iptables -E old-chain-name new-chain-name

Figura 8 Opciones de iptables a través del comando man.

Como ya habíamos dicho, iptables, funciona mediante tres tablas, a su vez, cada una de esas tablas, tiene definidas unas “chains” o cadenas. Cada una de estas chains se compone de una lista de reglas de filtrado. Cada regla no es más que un par condición/acción sobre atributos del paquete IP. El paquete, irá pasando secuencialmente por cada una de las reglas de la chain hasta encajar en el patrón de alguna de ellas. Cuando esto ocurra, el paquete se tratará según indique la acción de la regla con cuya condición el paquete ha hecho matching16. Si tras recorrer toda la lista, el paquete no encaja con ninguna de las reglas, se ejecutará la acción por defecto asociada a esa chain.
Ahora, podemos adentrarnos ligeramente en las opciones de la herramienta iptables.
Primero, veremos como manipular las chains. La utilización del comando, presenta siempre el siguiente patrón:
iptables [-t tabla] comando [match] [objetivos/saltos]
Las tablas son siempre una de las tres siguientes filter, nat, o forward. Si no se indica tabla alguna, por defecto, nos referimos a la tabla filter.
Para añadir y manipular cadenas utilizaremos siempre los comandos siguientes:
-iptables –N: Crear una nueva chain o cadena vacía (sin reglas).
-iptables –X: Elimina una chain que esté vacía (a excepción de las tres internas)
-iptables –F: Vacía una chain. Es decir, elimina todas las reglas de una chain.
-iptables –P: Cambia la política por defecto de una chain.
-iptables –L: Lista las reglas de una chain.
-iptables –Z: Pone a cero las variables de auditoría de una chain (número de paquetes, de bytes, etc…)

Con los comandos siguientes conseguimos manejar las reglas de esas cadenas:
-iptables –A: Inserta al final de una cadena una nueva regla.
-iptables –I: Inserta al comienzo de una chain una nueva regla.
-iptables –R: Reemplaza una regla de una chain.
-iptables –D: Elimina una regla de una chain (podemos indicar el orden o su condición).

Por último, sólo queda mostrar como podemos establecer las condiciones y acciones sobre cada regla, es decir como establecer las condiciones de match entre paquetes y reglas:
- iptables –s: Indica un dominio o IP (rango de IPs) de origen sobre el que se evalúa la condición de la regla.
- iptables –d: Indica un dominio o IP (rango de IPs) de destino el que se evalúa la condición de la regla.

- iptables –i (--in-interface): Indica la interfaz de entrada sobre la cual se evalúa la condición de la regla.
- iptables –o (--out-interface): Indica la interfaz de salida sobre la cual se evalúa la condición de la regla.
- iptables –p: Especifica el protocolo del datagrama que concordará con esta regla. Los nombres válidos de protocolos son tcp, udp, icmp, o un número, si se conoce el número del protocolo de IP. Cada protocolo lleva asociados sus propios modificadores a traves de las extensiones correspondientes:

- ✔ Extensiones de TCP:

–-sport: Especifica el puerto que debe utilizar el origen del datagrama para concordar con esta regla. Se pueden especificar los puertos en la forma de u rango, especificando los límites inferior y superior con los dos punto ":" como delimitador. Por ejemplo, 20:25 describe todos los puertos que van desde el 20 hasta el 25 incluyendo ambos. De nuevo, el signo "!" puede utilizarse para negar los valores.
--dport: Igual que la opción anterior pero para el puerto destino.
--tcp-flags: Especifica mediante una máscara si los bits indicadores de TCP del datagrama concuerden con ella. La máscara es una lista separada por comas de los indicadores que deben examinarse en la comprobación.(SYN, ACK, FIN, RST, URG, PSH, ALL o NONE).
-–syn: La regla casa con los datagramas cuyo bit SYN valga 1 y cuyos bits ACK y FIN valgan ambos 0. Esta opción es una abreviatura de: --tcp-flags SYN,RST,ACK SYN

- ✔ Extensiones UDP:

–-sport: Como en TCP, especifica el puerto que debe utilizar el origen del datagrama para concordar con esta regla.
--dport: Igual que la opción anterior pero para el puerto destino.

- ✔ Extensiones ICMP:

–-icmp-type: Puede especificarse el tipo de mesaje ICMP tanto por su número como por los siguientes identificadores: echo-request, echo- reply, source-quench, time-exceeded,network-unreachable, host-unreachable, Destination-unreachable, protocol-unreachable, y port unreachable.

- ✔ Extensiones MAC:

–-mac-source: Se especifica la dirección MAC (ethernet) Sólo tiene sentido en la cadena INPUT y FORWARD.
- iptables –f: Cuando se fracciona un datagrama porque supera el MTU de la red, podemos utilizar esta opción para especificar acciones sobre el segundo y restantes fragmentos del datagrama.

- – iptables ¡: Invierte el valor lógico de la condición de la regla.

7. Un caso real: Una Home-LAN.

El manejo de iptables es sencillo, aunque explicarlo mediante la simple narración puede hacerlo parecer más complicado. La mejor manera de entenderlo es a través de su aplicación a un caso real. Para ello, supongamos que vamos a configurar un cortafuegos de filtrado de paquetes para la red de la figura siguiente.

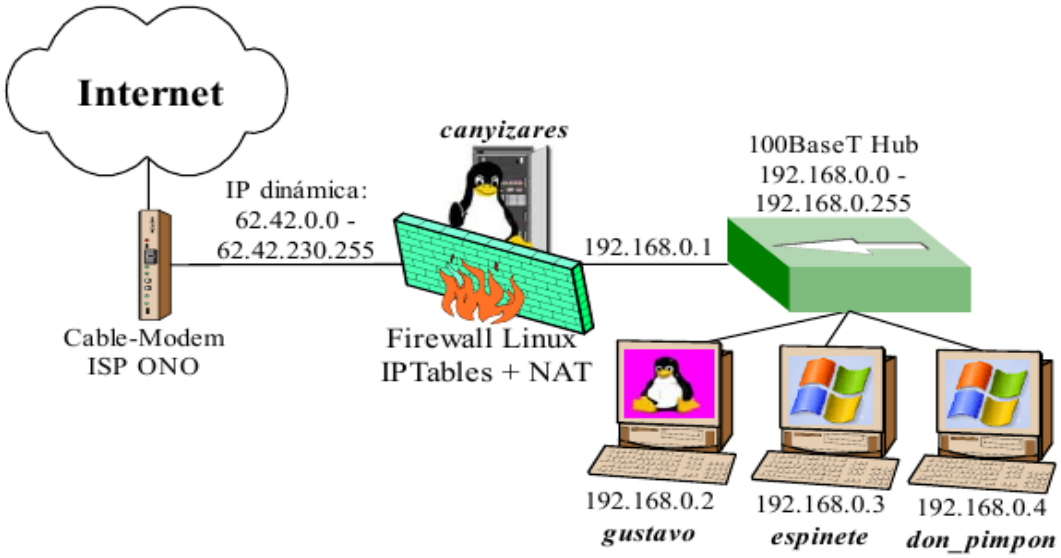


Figura 9 Home-LAN de ejemplo a configurar.

En principio, las tres maquinas de la red son simples estaciones de trabajo, que tal vez exportan algún directorio por SMB dentro de la propia red local. Por lo que debemos de preocuparnos principalmente de que tengan salida a Internet los principales servicios, clientes HTTP, FTP, etc. Por contra, en canyizares además de actuar de cortafuegos, queremos que corran servicios exportados tanto a la LAN como a Internet, por ejemplo un servidor HTTP, un servidor FTP y un servidor SSH para poder conectarnos remotamente a la máquina.
Además, deberemos resolver el problema del servicio DNS para la red local. La opción más sencilla, es configurar las máquinas de la red para que utilicen el mismo servidor DNS que la máquina cortafuegos (canyizares), es decir, los DNS del ISP (en este caso ONO). La otra alternativa, es ejecutar un servidor DNS en el cortafuegos y configurar las máquinas de la red local para que lo utilicen. Naturalmente, el servicio DNS que correrá en canyizares, será sólo un servicio de caché. Esta segunda configuración, es mas compleja, aunque ofrece ventajas respecto a la primera porque reduce el de peticiones tráfico DNS hacia Internet, ya que se resuelven localmente salvo cuando no están en caché, por esta misma causa, el tiempo medio de servicio en la resolución de un nombre de maquina desciende ligeramente.

Supondremos que la red interna es confiable, es decir, que de existir alguna amenaza, ésta vendrá desde Internet, no desde dentro. Tenemos que tener en cuenta que la IP proporcionada por ONO es dinámica, esto puede traernos pequeños quebraderos de cabeza en el caso que deseemos (como es nuestro caso, ya que corremos servicios en el cortafuegos) filtrar utilizando nuestra dirección IP pública. Hay diversas soluciones, pero dado que mi ISP ofrece cierta “estabilidad” en la duración17 de las IP (algunas me han durado meses). La solución más sencilla, es obtener la IP mediante la ejecución de:
EXT_IP=`ifconfig $EXT_IF | grep inet | cut -d : -f 2 | cut -d \ -f 1`

O de algún método equivalente como:

EXT_IP="`ifconfig $EXT_IF | grep 'inet addr' | awk '{print $2}' | sed -e 's/.*://'`"
Resuelto estros problemas, sólo queda estructurar el script en función de nuestras

necesidades.
Por ejemplo, para permitir accesos al servidor HTTP, FTP o SSH de canyizares desde cualquier lugar, bastaría con:

```
#Servicio FTP
iptables -t filter -A INPUT -p TCP -s 0/0 --dport 21 -j allowed
#Servicio SSH
iptables -t filter -A INPUT -p TCP -s 0/0 --dport 22 -j allowed
#Servicio HTTP
iptables -t filter -A INPUT -p TCP -s 0/0 --dport 80 -j allowed
```

Si el servidor HTTP se ejecutase en gustavo en lugar de en canyizares, deberíamos emplear DNAT.

```
#Servicio HTTP dentro de la LAN. Se exporta al exterior
iptables -t nat -A PREROUTING –i $EXT_IF -p tcp --dport 80 -j \
DNAT --to-destination 192.168.0.2
```

Tras ver la sintaxis de iptables y comprender como es su funcionamiento, el configurar un cortafuegos a medida es una cuestión relativamente sencilla. A media que nos volvemos “paranoicos” o necesitamos una mayor seguridad y a la vez el poder exportar más servicios, es cuando las cosas se van complicando.
El script de configuración de un cortafuegos para la red local de la figura anterior, se presenta como el primer anexo del documento. El cortafuegos presentado, es relativamente seguro y bastante sencillo y aun así son cerca de 300 líneas de código.
Si no se desea una seguridad o perfilado extremo, actualmente existen herramientas gráficas que permiten generar scripts generales de configuración, a pesar de que puede que no excesivamente eficientes, pueden servir de base para “retocarlo” después a mano y evitar teclear demasiado.
Configuracion:

Conclusiones

Los firewalls distribuidos ofrecen en muchos casos una alternativa eficiente y flexible a las soluciones tradicionales basadas en las limitaciones impuestas por la topologia de una red, pero también puden complementar y aumentar el nivel de seguridad logrado con un firewall de perímetro ya que pueden ser desplegados sobre una variedad de arquitecturas, incluso sobre una arquitectura de seguridad sin afectar su desempeño.

Adicionalmente, los firewalls distribuidos utilizan las tecnologías ya desarrolladas para los esquemas de seguridad ya existentes y han aparecido nuevas tecnologias que se adaptan mas fácilmente a este nuevo modelo de firewalls como al modelo anterior, ampliando el rango de posibilidades a considerar al momento de asegurar una red privada.

Existen en el mercado variadas herramientas desarrolladas bajo este nuevo enfoque que implementan en mayor o menor medida las características de los firewalls distribuidos. De todas formas, aún no existe una alta compatibilidad entre las diferentes herramientas para lograr una solución que aproveche la completa interacción de todos los aspectos de seguridad que una organización necesita para efectuar sus actividades sobre Internet; los firewalls son relativamente nuevos y llevará algún tiempo para que logren el nivel de evolución que hoy en día disfrutan los firewalls tradicionales.

Observamos que la IP Table es equivalente a un firewall por software y viene en los SO Linux en los Kernel desde la versión 2.4 en adelante.

Se recomienda configurarla cuando se realiza la actualización del Kernel de los SO Linux

[Bellovin] **Distributed Firewalls,**por Steven M. Bellovin, Noviembre 1999.

[NIST-Draft] **IMPLEMENTING INTERNET FIREWALL SECURITY POLICY**, por Barbara Guttman, Robert Bagwill, NIST Special Publication 800-XX (DRAFT), 1998

Direcciones de Internet con información de iptables:

Pagina Principal de Netfilter/Iptables.

http://netfilter.samba.org | http://netfilter.filewatcher.org.

Excelente página con gran cantidad de información sobre iptables y netfilter.

http://www.linuxguruz.org/iptables/.

Charla en el IRC de uninet sobre el netfilter.

http://6fevu.uninet.edu/text/laforge.html, http://6fevu.uninet.edu/text/laforge1.html

http://6fevu.uninet.edu/text/laforge2.html, http://6fevu.uninet.edu/text/laforge22.html

http://6fevu.uninet.edu/text/redes22.html.

Distintos tutoriales sobre iptables.

http://www.glug.es/sections.php?op=viewarticle&artid=1

http://www.linux-mag.com/cgi-bin/printer.pl?issue=2000-01&article=bestdefense

http://pinehead.com/articles.php?view=371.

Direcciones de Internet con información sobre Linux:

http://bulmalug.net/

http://lucas.hispalinux.es/

http://www.europe.redhat.com/documentation/

http://www.google.com/intl/es/