IPTABLES

Antonio Tapiador del Dujo

Noviembre, 2001

Índice General

1 Objetivos y descripción del servicio

IPTABLES es la herramienta que crea y configura las tablas de filtrado IP en los núcleos Linux 2.4. Además permite hacer traducción de direcciones de red origen (SNAT) y destino (DNAT).

Las opciones de filtrado permiten construir el típico cortafuegos. Un cortafuegos es un programa que examina la cabecera de los paquetes que llegan a una máquina, y decide si se dejan pasar o si se descartan de acuerdo con unas reglas que se le han introducido previamente. El cortafuegos protegerá del acceso externo a ciertos servicios privados de nuestra red o de la propia máquina conectada a una red pública.

La traducción de direcciones origen hace que todas las conexiones desde el interior de una red privada hacia otra red pública parezcan provenir de la máquina que encamina los paquetes entre las dos redes. Un ejemplo ampliamente usado de DNAT es el enmascaramiento de IP, el cual permite a varias máquinas hacer conexiones al exterior teniendo disponible una única dirección IP, y además oculta la red interna al exterior.

La traducción de direcciones destino permite modificar ciertas conexiones de forma transparente. Esto puede ser útil para realizar redireccionamiento de puertos o situar proxies transparentes, en los cuales las máquinas clientes creen realmente que están realizando una conexión con el exterior.

IPTABLES sucede a IPCHAINS, herramienta de los núcleos Linux 2.2.

2 Requisitos

Lo primero es disponer de las herramientas IPTABLES. Como siempre, se pueden descargar las fuentes de cualquiera de los sitios oficiales[1] y proceder a su compilación e instalación. O más sencillo y recomendable, hacer uso de la paquetería de la distribución.

En Debian se pueden obtener instalando el paquete *iptables*, mediante un simple:

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

Requisito indispensable para usar IPTABLES es tener un núcleo Linux con netfilter, esto es, posterior al 2.3.15. Este núcleo debe tener el soporte compilado. Si no, podemos configurarlo y compilarlo a partir de las fuentes. La configuración se explica más adelante.

Además, necesitamos tener instalada la librería libc6.

Para que el uso de las herramientas tenga sentido, la máquina donde se va a ejecutar IPTABLES deberá tener conexiones con otras redes...

3 Arquitectura del servicio

En el núcleo hay tres posibles tablas de reglas, cuya existencia dependerá de su configuración y de los módulos cargados en ese momento. Las tres posibles tablas son la de filtrado, la de traducción de direcciones y la de alteración de los paquetes. Cada tabla contiene unas cadenas construidas por defecto, las cuales hacen alusión a un momento en la gestión del paquete por parte del núcleo. Los paquetes según llegan, se generan o se encaminan tienen que pasar por las diferentes cadenas. Además de estas cadenas, se pueden definir otras nuevas en cualquiera de las tablas, y redireccionar ciertos paquetes hacia ellas.

En cada cadena se definen unas reglas. Los paquetes que entran en una cadena se van comparando con las reglas de dicha cadena, y si cumplen alguna, su destino estará determinado por lo que diga dicha regla. Si el paquete no cumple ninguna de las reglas, entonces seguirá la política por defecto de la cadena.

3.1 Tablas y cadenas

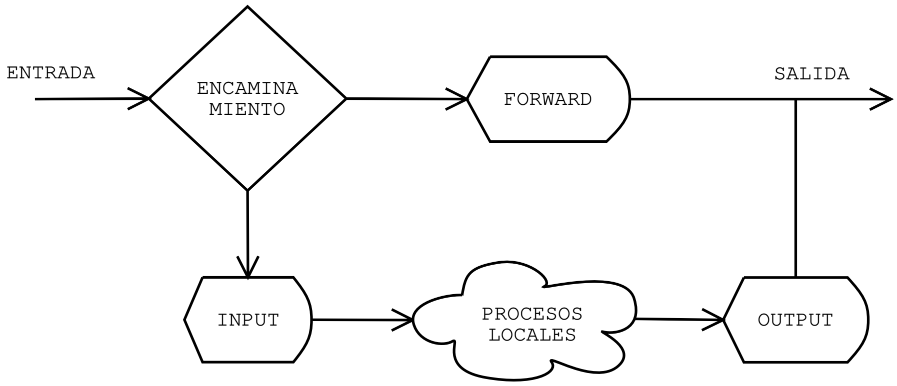
3.1.1 filter

Es la tabla de filtrado. Se utiliza para definir los paquetes que se van a dejar pasar y los que se van a tirar, según las reglas que definamos en la tabla. Contiene tres cadenas ya construidas:

- INPUT - para los paquetes que tienen como destino la máquina
- OUTPUT - para los paquetes que tienen como origen la máquina
- FORWARD - para los paquetes que encamina la máquina

El aspecto gráfico de dichas cadenas dentro del procesamiento de paquetes del núcleo es el siguiente:

[1] Esto es: http://netfilter.filewatcher.org, http://netfilter.samba.org y http://netfilter.gnumonks.org



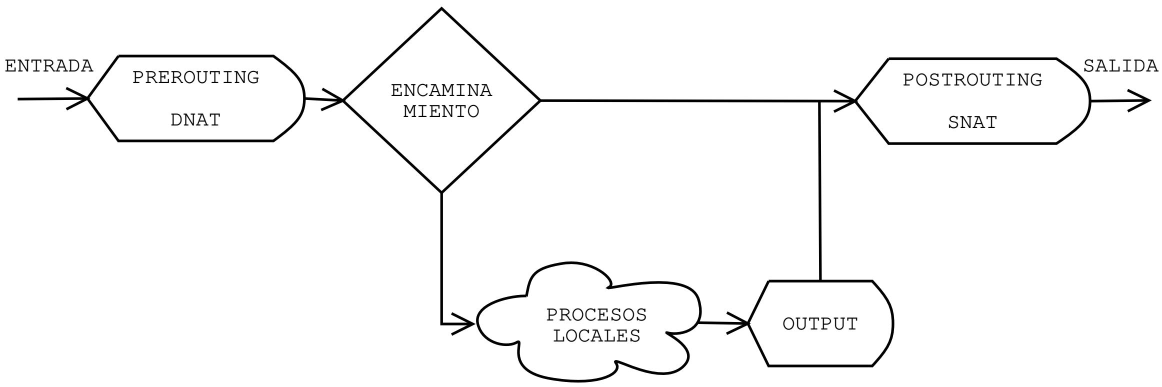
Según el destino que tenga el paquete que llega, se decidirá si tiene como objetivo la misma máquina u otra. Entonces se examinarán las reglas de la cadena correspondiente, INPUT o FORWARD.
Si por el contrario el paquete lo genera algún proceso de la máquina, la cadena a examinar será OUTPUT.

3.1.2 nat

Se consulta cuando se encuentra cualquier paquete que crea una conexión. Tiene tres cadenas por defecto:

- PREROUTING - altera los paquetes según llegan a la máquina
- OUTPUT- altera los paquetes que se han creado en la máquina
- POSTROUTING - altera los paquetes al salir de la máquina

El esquema de dichas cadenas es el siguiente:



Como se puede observar, los paquetes que sufren DNAT o SNAT se alteran según entran en la máquina, antes de la decisión de encaminamiento, o inmediatamente antes de salir de la máquina. Así tenemos la posibilidad, por ejemplo, de alterar paquetes que no tenían como destino la máquina, para que a la hora del encaminamiento sí que la tengan como destino.

3.1.3 mangle

Esta tabla se usa para alteraciones especiales de los paquetes. Tiene dos cadenas:

- PREROUTING - para manipular los paquetes que llegan antes de encaminarlos

- OUTPUT - para manipular los paquetes generados en la máquina

Su aspecto gráfico es similar al de la tabla anterior.

3.2 Las reglas

Las reglas tienen el siguiente aspecto:

```
# iptables -L
Chain INPUT (policy DROP)
target  prot  opt   source       destination
ACCEPT  all   --    anywhere     anywhere         state RELATED,ESTABLISHED
ACCEPT  all   --    localhost    localhost        state NEW

Chain FORWARD (policy DROP)
target  prot  opt   source       destination

Chain OUTPUT (policy ACCEPT)
target  prot  opt   source       destination
```

Observamos que la única cadena con reglas es INPUT.

Cada regla tiene un origen y destino de los paquetes a los cuales se refiere. Además puede hacer distinción entre protocolos y varias opciones más. Uno de los campos, target, define el destino de los paquetes que cumplan dicha regla. Cuando un paquete no cumple ninguna de las reglas de la cadena, entonces sigue la política por defecto de la cadena.

En este ejemplo, en la primera regla se dice que se dejen pasar (ACCEPT) los paquetes de origen y destino cualquiera, de cualquier protocolo, cuyo estado corresponda a una conexión establecida (ESTABLISHED), o relacionada con una ya activa (RELATED).

En la segunda, se permiten las conexiones nuevas sólo de la propia máquina consigo misma.

La política por defecto de la regla es tirar los paquetes: DROP. Por lo tanto, un paquete que iniciara una conexión desde fuera de la máquina, no cumpliría ninguna de las dos reglas, por lo tanto seguiría la política de la cadena: se descartaría como si el paquete no hubiera llegado nunca a la máquina.

4 Configuración

4.1 El núcleo Linux

Podemos hacer uso de un núcleo ya compilado con soporte para netfilter y los módulos que vayamos a usar. Sin embargo, es aconsejable hacernos un núcleo a medida. No entraré a describir cómo se compila e instala un núcleo, ya que es algo que un administrador sabrá hacer[2].

Las diferentes características se pueden compilar dentro del núcleo o como módulos cargables, según sean nuestras preferencias. Si elegimos módulos cargables y no activamos el autocargador (CONFIG_KMOD), deberemos de incluir los módulos a mano según los necesitemos.

Además de las opciones propias de netfilter, existen otras que deben de estar activadas (como soporte para red, para tarjetas de Ethernet y/u otros adaptadores...) que no entraré a comentar. Las opciones propias de netfilter son:

Sección Networking options:

- Network packet filtering (replaces ipchains): Es la opción imprescindible para poder usar IPTABLES.

  Dentro de IP: Netfilter Configuration:

[2]Información en el Kernel-HOWTO o, en Debian, la documentación del paquete *kernel-package*, para compilar e instalar un núcleo al estilo Debian.

- Connection tracking(required for masq/NAT): para llevar una cuenta de los paquetes de las distintas conexiones. Necesario para enmascaramiento y traducción de direcciones.
- FTP protocol support: para enmascarar conexiones FTP.
- Userspace queueing via NETLINK (EXPERIMENTAL[3]): para copiar paquetes en espacio de usuario y analizarlos más tarde.
- IP tables support (required for filtering/masq/NAT): autoexplicativo.
- limit match support: permite controlar la tasa de paquetes que verifican una regla. Útil para evitar ataques de denegación de servicio o llenar los ficheros de registro.
- MAC address match support: soporte para direcciones MAC de Ethernet.
- netfilter MARK match support: para manejar los paquetes según el valor nfmark
- Multiple port match support: permite crear reglas para rangos múltiples de puertos.
- TOS match support: permite seleccionar paquetes según el valor del campo "Tipo de Servicio" de la cabecera IP.
- tcpmss match support: para controlar los paquetes según la longitud máxima de los segmentos TCP.
- Connection state match support: permite definir reglas teniendo en cuenta el estado en que se encuentran las conexiones.
- Unclean match support (EXPERIMENTAL): para identificar paquetes extraños o inválidos mirando los campos de las cabeceras de los distintos protocolos.
- Owner match support (EXPERIMENTAL): permite separar paquetes según su UID, GID, PID o sesión.
- Packet filtering: permite filtrar paquetes. Define la tabla filter explicada más adelante.
- REJECT target support: permite devolver un paquete de error ICMP en vez de simplemente tirar el paquete.
- MIRROR target support (EXPERIMENTAL): permite devolver un paquete semejante al recibido. Con esta opción un posible atacante podría, por ejemplo, escanearse los puertos a sí mismo.
- Full NAT: soporte para realizar los distintos tipos de traducción de direcciones. Define la tabla nat que se explica más adelante.
- MASQUERADE target support: soporte para enmascaramiento.
- REDIRECT target support: para redireccionamiento de conexiones. Útil para proxies transparentes.
- Packet mangling: para realizar alteraciones en los paquetes. Define la tabla mangle explicada más adelante.
- TOS target support: permite manipular el campo Tipo de Servicio antes de encaminar el paquete.
- MARK target support: permite manipular el paquete basándose en el campo MARK.
- LOG target support: permite registrar paquetes mediante el syslog.
- TCPMSS target support: permite alterar el valor MSS en los paquetes SYN de TCP.
- ipchains (2.2-style) support: para poder usar la herramienta antigua ipchains y la mayoría de sus reglas.
- ipfwadm (2.0-style) support: para poder usar la herramienta más antigua todavía, ipfwadm.

[3]Las opciones marcadas como EXPERIMENTAL están en desarrollo, y se debe activar la opción "*Code maturity level options → Prompt for development and/or incomplete code/drivers*" para poder contar con ellas.

4.2 iptables

La introducción, modificación y eliminación de reglas y cadenas, listado de las tablas, etc.. se realiza mediante la orden `iptables`, según la "acción" que se le pase como argumento (`-Letra`).

- Para creación, borrado y renombrado de cadenas usaremos, respectivamente:

  ```
  # iptables -[NXE] cadena [nombre_nuevo]
  ```

  Son muchas las posibilidades de las que se pueden hacer uso mediante la creación de nuevas cadenas.

- Para cambiar la política por defecto de una cadena:

  ```
  # iptables -P cadena objetivo [opciones]
  ```

- Para sacar una lista de las cadenas de una tabla:

  ```
  # iptables -L [-t tabla]
  ```

- Se puede resetear el contador de paquetes y bytes de las cadenas mediante:

  ```
  # iptables -Z
  ```

- La sintaxis usada para manipular las reglas de una cadena es:

  ```
  # iptables -[ADRIF] cadena [parámetros y/o opciones] [-j TARGET]
  ```

  Las acciones descritas son varias: añadir, eliminar, reemplazar, insertar o eliminar todas las reglas de la cadena.

  Las opciones también son muy diversas. Con ellas vamos a poder ajustar bastante los paquetes a los que se refiere cada regla. Se pueden elegir protocolos, IPs, redes y subredes origen y destino, rangos de puertos, interfaces de entrada y salida, estados de la conexión, PID, UID, GID...

  Por último, la opción `-j` indica el destino de los paquetes que cumplen la regla. Dicho destino puede ser aceptar el paquete, tirarlo, devolverlo, enmascararlo, manipular sus direcciones origen o destino... Varias de estas opciones necesitan soporte en el núcleo según se ha comentado en el punto anterior.

Las opciones y posibilidades de IPTABLES son bastante numerosas como para comentarlas todas en este trabajo, ya que su extensión se alargaría considerablemente. Se pueden consultar cómodamente en la página de manual.

4.3 Reglas permanentes

Las reglas y cadenas nuevas que se introducen mediante el uso de IPTABLES, se guardan en el espacio del núcleo, por lo que una vez apagada la máquina estas se perderán. Hay varias opciones para hacer las reglas permanentes, o por lo menos que se carguen cuando las vayamos a necesitar.

- Una de las maneras es salvar la configuración de las tablas para cargarlas posteriormente. Para ello se pueden utilizar las órdenes `iptables-save` y `iptables-restore`, con las que utilizaremos un archivo para guardarlas y recuperarlas más tarde.

  Para guardar la configuración una vez tengamos las reglas adecuadas:

  ```
  # iptables-save > /etc/iptables.conf
  ```

  Para cargarlas sólo tendríamos que invocar:

```
# cat /etc/iptables.conf | iptables-restore
```

Esta orden se pondría en un script para que se ejecute automáticamente.

- La otra manera es sencillamente hacer un script con las reglas que queremos cargar. Por ejemplo:

```
#!/bin/sh
# Script para filtrado basico con conexion por modem

# Denegamos por defecto:
iptables -P INPUT DROP
iptables -P FORWARD DROP

# Aceptamos las conexiones ya establecidas
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A INPUT -m state --state NEW -i !ppp0 -j ACCEPT
```

A la hora de ejecutar el script correspondiente tenemos dos opciones:

- Cargar el script según se inicia el sistema. Para ello lo situaremos en el directorio adecuado, que varía según la distribución.

  En Debian, la manera correcta sería situar el script en `/etc/init.d/` y darle permisos de ejecución. Además, se debe hacer un enlace simbólico desde los directorios de los niveles de ejecución donde se desea que se ejecute (típicamente serían del 2 al 5), esto es, en `/etc/rcX.d/`, con una prioridad posterior al del inicio de la red (que suele ser 20). Para ello se puede utilizar la orden `update-rc.d` del siguiente modo (suponiendo que el script se llama `iptables`):

```
# update-rc.d iptables start 21 2 3 4 5 .
Adding system startup for /etc/init.d/iptables ...
/etc/rc2.d/S21iptables -> ../init.d/iptables
/etc/rc3.d/S21iptables -> ../init.d/iptables
/etc/rc4.d/S21iptables -> ../init.d/iptables
/etc/rc5.d/S21iptables -> ../init.d/iptables
```

- Cargar el script cada vez que nos conectemos. Para ello podemos incluir el script, en el caso de conexiones ppp, en el directorio `/etc/ppp/ip-up.d/` y darle permisos de ejecución.

  Si el script contiene las reglas, y tenemos una conexión que se corta cada dos por tres (bastante habitual por estas fechas), se debe tener cuidado ya que cada vez que se reconecta se incluirán unas nuevas reglas iguales. Para evitar esto se debe incluir otro script en `/etc/ppp/ip-down.d/` que nos deje las cosas como estaban cada vez que se cae la conexión.

5 Pruebas

Para las pruebas elegiremos un escenario cada vez más habitual. Tenemos una conexión ppp y una red local. La máquina que actúa como router entre la red local e internet filtrará los paquetes de internet para proteger servicios internos (como por ejemplo, servicios de impresión o montaje de ficheros. Aunque éstos deberían de estar correctamente configurados, pueden existir fallos en los programas que los hagan vulnerables hasta que se actualicen). Dejaremos un servidor de correo accesible. Además enmascarará las conexiones de la red interna.

En la máquina que actúa como router definimos las siguientes reglas:

```
#!/bin/sh
# Reglas para filtrado y enmascaramiento de red local
# Cargamos el modulo para FTP
/sbin/modprobe ip_nat_ftp

# Habilitamos el reenvio de paquetes
echo "1" > /proc/sys/net/ipv4/ip_forward

# Habilitamos medidas para direcciones dinamicas
echo "1" > /proc/sys/net/ipv4/ip_dynaddr

# Politica por defecto: se deniega todo lo que no este
# expresamente permitido
iptables -P INPUT DROP
iptables -P FORWARD DROP

# Habilitamos el enmascaramiento a conexiones por ppp0
iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE

# Reglas de entrada:
# Aceptamos conexiones ya abiertas o relacionadas
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# Aceptamos las conexiones nuevas propias y de la red local
iptables -A INPUT -m state --state NEW -i lo -s 127.0.0.1 -d 127.0.0.1 -j ACCEPT
iptables -A INPUT -m state --state NEW -i eth0 -s 192.168.0.0/24 -j ACCEPT

# Dejamos abierto el servidor de correo
iptables -A INPUT -p tcp --dport 25 -j ACCEPT

# El resto de los paquetes los monitorizamos para comprobar las pruebas
iptables -A INPUT -i ppp0 -m limit -j LOG --log-prefix "Bad packet from ppp0 in:"
iptables -A INPUT -i ! ppp0 -m limit -j LOG --log-prefix "Bad packet not from
ppp0 in:"

# Reglas de reenvio
# Aceptamos conexiones establecidas o relacionadas
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

# Aceptamos conexiones nuevas de la red local
iptables -A FORWARD -m state --state NEW -i eth0 -s 192.168.0.0/24 -j ACCEPT

# Monitorizamos el resto
iptables -A FORWARD -i ppp0 -m limit -j LOG --log-prefix "Bad packet from ppp0
fwd:"
iptables -A FORWARD -i ! ppp0 -m limit -j LOG --log-prefix "Bad packet not from
ppp0 fwd:"
```

Configuraremos las máquinas de la red local adecuadamente, indicándolas que la pasarela hacia internet es la ip interna del router e indicando los correspondientes DNS.

Ahora probaremos la configuración.

5.1 Prueba del cortafuegos

Haremos unas pruebas para comprobar que se filtran los paquetes correctamente. Para ello, nos conectamos a la máquina de alumnos de la Escuela, e intentamos acceder al puerto 631 (donde realmente está escuchando CUPS):

```
antoniot@enano:~$ telnet 83-madr-x12.libre.retevision.es ipp
Trying 62.83.1.83...
telnet: Unable to connect to remote host: Connection timed out
```

El intento de conexión aparece en la monitorización:

```
Nov 29 00:35:31 langri kernel: Bad packet from ppp0 in:IN=ppp0 OUT= MAC=
SRC=138.100.17.69 DST=62.83.1.83 LEN=60 TOS=0x10 PREC=0x00 TTL=47 ID=29711 DF
PROTO=TCP SPT=51991 DPT=631 WINDOW=5840 RES=0x00 SYN URGP=0
```

Sin embargo, haciendo una conexión al puerto del MTA:

```
antoniot@enano:~$ telnet 83-madr-x12.libre.retevision.es 25
Trying 62.83.1.83...
Connected to 83-MADR-X12.libre.retevision.es.
Escape character is '^]'.
220 langri ESMTP Exim 3.32 #1 Thu, 29 Nov 2001 00:41:58 +0100
quit
221 langri closing connection
Connection closed by foreign host.
```

Y no aparece nada en los registros del cortafuegos.

El cortafuegos ha filtrado la primera conexión y ha dejado pasar la segunda, tal y como cabría esperar.

5.2 Prueba del enmascaramiento

Desde una máquina interna hacemos un ping al exterior:

```
$ ping debian.org
PING debian.org (198.186.203.20): 56 data bytes
64 bytes from 198.186.203.20: icmp_seq=0 ttl=241 time=318.9 ms
...
```

El el router observamos los paquetes que llegan:

```
# tcpdump -i eth0
tcpdump: listening on eth0
19:08:09.161303 192.168.0.2 > klecker.debian.org: icmp: echo request (DF)
19:08:09.478689 klecker.debian.org > 192.168.0.2: icmp: echo reply
...
```

Y los paquetes que salen:

```
# tcpdump -i ppp0
tcpdump: listening on ppp0
19:08:59.041421 62.83.2.26 > 198.186.203.20: icmp: echo request (DF)
19:08:59.378648 198.186.203.20 > 192.168.0.2: icmp: echo reply
...
```

Como podemos comprobar, los paquetes que entran al router por el interfaz de Ethernet tienen la dirección origen de la máquina de la red interna. El router realiza el enmascaramiento de direcciones, y los paquetes que salen por el modem llevan la dirección IP origen externa del router. Los paquetes de vuelta sufren el proceso contrario.

6 Gestión diaria

Una vez montado el servicio, la gestión que hay que hacer es poca.

Si se piensa montar algún servicio nuevo, se deberá abrir el puerto correspondiente, lo cual no requerirá mucho trabajo.

En todo caso, se podrán revisar los ficheros de registro de vez en cuando, por si se encuentra alguna cosa extraña, o para vigilar su tamaño (el cual se puede gestionar automáticamente mediante la herramienta LOGROTATE).

Referencias

1. Páginas de manual de IPTABLES

2. *Linux 2.4 Packet Filtering HOWTO*, Rusty Russell
   `http://netfilter.samba.org/unreliable-guides/packet-filtering-HOWTO/packet-filtering-HOWTO.linuxdoc.html`

3. *Linux 2.4 NAT HOWTO*, Rusty Russell
   `http://netfilter.samba.org/unreliable-guides/NAT-HOWTO/NAT-HOWTO.linuxdoc.html`

4. *Linux IP Masquerade HOWTO*, David A. Ranch
   `http://www.linuxdoc.org/HOWTO/IP-Masquerade-HOWTO.html`