Iptables

Firewall de inspección de paquetes con estado

Juan Nieto - http://tecnoloxiaxa.blogspot.com

ÍNDICE

1.- INTRODUCCIÓN SPI:
¿QUÉ ES UN FILTRO DE PAQUETES?
INSTALACIÓN DE IPTABLES
CREACIÓN DE FIREWALLS DE IPTABLES
2.- EL FUNCIONAMIENTO DE IPTABLES I
2.1- COMANDOS IPTABLES
2.2 - REGLAS IPTABLES
2.3.- OPCIONES
3. PRIMEROS PASOS
4.IPTABLES AVANZADO
4.1- ESPECIFICACIONES DE FILTRADO
4.2- EXTENSIONES A IPTABLES
4.2.1- LAS EXTENSIONES DE COINCIDENCIAS
4.3 ESPECIFICACIÓN DE OBJETIVOS (TARGET)
4.3.1 CADENAS DEFINIDAS POR EL USUARIO
4.3.2 EXTENSIONES A IPTABLES: NUEVOS OBJETIVOS
4.3.3 OBJETIVOS ESPECIALES DE SERIE
5. OPERACIONES CON UNA CADENA
6. CONFIGURACIÓN DE RED DE HOST INDEPENDIENTES
7. ¿QUÉ ES NAT?
7.1 ENMASCARAR
7.2 QUÉ PASAR POR NAT
7.3 SELECCIONES USANDO IPTABLES
7.4 CÓMO MODIFICAR LOS PAQUETES
7.4.1 SOURCE NAT (CAMBIO DE ORIGEN)
7.4.2 ENMASCARAMIENTO
7.4.3 DESTINATION NAT (CAMBIO DE DESTINO)
7.4.3.1 REDIRECCIÓN
7.5 PROTOCOLOS ESPECIALES Y DEFECTOS DE NAT
8. FIREWALL NAT, CASOS PRÁCTICOS
8.1 UN CASO SENCILLO
8.2 FIREWALL CON REENVÍO DE PUERTOS
8.3 FIREWALL CON DMZ Y PROXY WEB TRANSPARENTE
8.4 VPN IPSEC A TRAVÉS DEL FIREWALL
9. IMPLEMENTACIÓN DE MARCAS DE TIPO DE SERVICIO (TOS)
10. BIBLIOGRAFÍA

1.- INTRODUCCIÓN SPI:

1. ¿QUÉ ES UN FILTRO DE PAQUETES?
2. INSTALACIÓN DE IPTABLES
3. CREACIÓN DE FIREWALLS DE IPTABLES

1.- Introducción SPI

- A partir del desarrollo del núcleo 2.3 (el precursor del núcleo 2.4), los programadores del núcleo de Linux comenzaron a trabajar en Iptables, la respuesta de Linux a los firewalls comerciales de inspección de paquetes de estado.
- Con Iptables, el firewall se puede programar para asociar todo el tráfico devuelto generado a partir de una regla input anterior
- Las ventajas de la tecnología de inspección de paquetes de estado (SPI, *Stateful Packet Inspection)* no se limitan a la eficacia de las reglas. Ipchains no contiene la inteligencia para diferenciar la «verdadera naturaleza» del tráfico de red.
- Los filtros de paquetes pasan simplemente el tráfico aceptado a través de huecos en el firewall y rechazan el tráfico no válido. Estos huecos del firewall siempre permanecen abiertos. Por ejemplo, un firewall Ipchains programado para permitir el tráfico Telnet de salida también tendrá una regla input asociada para permitir la devolución de paquetes. Si un atacante pudiera fabricar paquetes que simularan paquetes Telnet devueltos, Ipchains permitiría su entrada. Con SPI no existiría ninguna sesión para asociar estos paquetes falsificados, y, por tanto, el firewall los rechazaría.

¿Qué es un filtro de paquetes?

- Un filtro de paquetes es un software que **examina la *cabecera*** de los paquetes según van pasando, y decide la suerte del paquete completo. Podría decidir descartarlo (DROP) (esto es, como si nunca lo hubiera recibido), aceptarlo (ACCEPT) (dejar que pase), o cosas más complicadas.
- En Linux, el filtrado de paquetes está programado en el núcleo (como módulo o como componente estático), y hay algunas cosas curiosas que podemos hacer con los paquetes, pero *aún* sigue ahí el principio general de mirar las cabeceras para decidir la suerte del paquete
- Necesitamos un núcleo que contenga la infraestructura netfilter. Netfilter es un área de trabajo general dentro del núcleo, a la que pueden conectarse otras cosas (como el módulo (le iptables). http: //www.netfilter.org/
- La herramienta iptables se comunica con el núcleo y le dice qué paquetes filtrar

Instalación de Iptables

- El firewall de núcleo de Iptables está compuesto por dos elementos diferentes: Iptables y el núcleo de Linux. El primer componente, Iptables, es la interfaz de **la** línea de comandos para las funciones de firewall de núcleo. Se trata de una aplicación como cualquier otro software del sistema que necesita compilación. La mayoría de las versiones de Linux incluyen Iptables instalado de manera pre-determinada. Sin embargo, para los que desean realizar la instalación «por sí mismos», el código fuente y las instrucciones de instalación se encuentran en *http://netfilter.samba.org.*

- El segundo componente del sistema de firewall de núcleo Iptables es el núcleo de Linux. El firewall es, simplemente, una de las muchas capacidades indi-viduales inherentes a los núcleos 2.4. Para habilitar las funciones de firewall, debe definir ciertas opciones de configuración.

Creación de firewalls de Iptables

- Antes de establecer ninguna regla en Iptables, el núcleo de Linux debe configurarse para actuar como un enrutador. En la mayoría de las versiones actuales de Linux, la capacidad del núcleo para enrutar paquetes está deshabilitada. Se trata de una medida de seguridad para que los intrusos no puedan utilizar un host Linux como un enrutador para tener acceso a redes internas.
- El enrutamiento, al igual que muchas otras carac-terísticas de la pila TCP/IP, se define en el directorio /proc/sys/net/ipv4. /proc es un sistema de archivos virtual que permite al usuario definir dinámicamente valores de configuración de núcleo. En este caso, la variable ip_forwarding debe definirse como 1.
- El siguiente comando, colocado en una secuencia de comandos de inicio como / etc/rc.d/rc.local, activará el enrutamiento al indicar:
    - echo 1 > /proc/sys/net/ipv4/ip_forward
- La mayoría de los usuarios de Iptables escribirán una sencilla secuencia de comandos de shell que contiene todas las llamadas de la línea de comandos de Iptables. Esta secuencia de comandos se puede configurar para eje-cutarse al iniciar como una función rc.d. Es aconsejable ejecutar la secuencia de comandos inmediatamente después de inicializar las interfaces de red. Esto asegu-ra que los intrusos no abusen de las debilidades del sistema durante el inicio. En realidad, el núcleo puede cargar reglas de firewall incluso antes de mostrar las interfaces, siempre que las reglas utilicen direcciones IP en lugar de nombres de dominio completos. El uso estricto de direcciones IP en este caso es crítico, ya que el firewall

2.- EL FUNCIONAMIENTO DE IPTABLES

I

2.1- COMANDOS IPTABLES

2.2 - REGLAS IPTABLES

2.3.- OPCIONES

2.- El funcionamiento de Iptables I

- La herramienta iptables inserta y elimina reglas de la tabla de filtrado de paquetes del núcleo. Esto significa que cualquier cosa que establezca, se perderá cuando reinicie.
- Para hacer **permanentes las** reglas creadas hay que tener en cuenta:
  - La configuración que el cortafuegos tiene en este momento está almacenada en el núcleo, de manera que se perderá cuando reinicie.
  - Para evitarlo, escribiremos las órdenes necesarias para configurar las reglas en un guión (script) de inicio. Asegurándonos de que hace algo inteligente en el caso de que alguna de las reglas falle (normalmente «exec /shin/sulogin»)
  - Esta secuencia de comandos se puede configurar para ejecutarse al iniciar como una función rc.d. (es aconsejable hacerlo tras inicializar las interfaces de red)

2.- El funcionamiento de Iptables II

2.1- Comandos Iptables

- *-A cadena* Se anexa a la cadena especificada.
- *-I cadena* Inserta la regla especificada en la parte superior de la cadena.
- *-D cadena [númeroDeregla]* Elimina la regla especificada de la cadena o la regla en la posición específica.
- -*R cadena [númeroDeregla]* Reemplaza la regla especificada en la posición especificada.
- *-L cadena* Enumera las reglas en una cadena especi-ficada.
- *-F [cadena]* Vacía las reglas de la cadena especificada. Vacía todas las reglas si no se especifica ninguna cadena.
- *-Z [cadena]* Quita los contadores de bytes de la cadena es-pecificada. Quita todos los contadores de bytes si no se especifica ninguna cadena.
- -*N cadena* Crea una nueva cadena especificada por el usuario.
- *-X cadena* Elimina la cadena especificada definida por el usuario.
- -*P cadena* Define la directiva predeterminada para la cadena: DROP, REJECT, REDIR o ACCEPT.

2.2 - Reglas Iptables

- Las opciones para el campo de parámetros Especificación de reglas del comando *Iptables* se describen en la siguiente tabla:

- **Especificación de reglas Descripción**
- *-p protocolo* Especifica un protocolo: tcp, udp, icmp, all, o un número de protocolo.
- -s --sport Especifica el puerto y la dirección de origen.
- -d --dport Especifica el puerto y la dirección de destino.
- -i -o Especifica la interfaz de red, de entrada y salida.
- *-j cadena* Salta a la cadena especificada: ACCEPT, DROP, REJECT, REDIRECT, SNAT [- -to-source], DNAT [- -to-dest], MASQUERADE o LOG [- -log- prefix].

2.3.- Opciones

- Las opciones para el campo de parámetros Opción del comando *Iptables* se describen en la siguiente tabla:
- **Opción Descripción**
- -v Resultado detallado.
- -m -state Marcar el paquete con.
- -n Al enumerar las reglas, no realiza una búsqueda DNS en direcciones IP.
- -l Registra una coincidencia con esta regla.
- -y Hace coincidir paquetes con el conjunto de bits SYN.
- *-t máscaraDebit* Permite la manipulación del tipo de servicio (ToS) bit.

3. Primeros pasos (I)

- Para ver los filtros activos: iptables –L
- Un caso sencillo: filtrar el tráfico icmp de una dirección ip.
  - Para el ejemplo vamos a filtrar todo el tráfico icmp (ping) proveniente de la interfaz “loopback” (127.0.0.1), de tal manera que sea descartada
  - Primero comprobamos que funciona
    - **ping -c 1 127.0.0.1**
      - PING 127.0.0.1 (127.0.0.1): 56 data bytes 64 bytes from 127.0.0.1: icmp_seq=0 tt1=64 time=0.2 ms-- 127.0.0.1 ping statistics
      - 1 packets transmitted, 1 packets received, O% packet loss round-trip min/avg/max = 0.2/0.2/0.2 ms
  - Activamos el filtro y lo probamos:
    - **iptables -A INPUT -s 127.0.0.1 -p icmp -j DROP**
    - **ping -c 1 127.0.0.1**
      - PING 127.0.0.1 (127.0.0.1): 56 data bytes
      - -- 127.0.0.1 ping statistics 1 packets transmitted, 0 packets received, 100% packet loss
  - Para borrar la regla dos opciones:
    - # iptables -D INPUT -s 127.0.0.1 -p icmp -j DROP
    - # iptables -D INPUT 1

3.- Primeros Pasos (II)

- La mayoría (le la gente tiene sólo una conexión PPP *a* Internet, y no quiere que nadie entre *a* su red, o al cortafuegos:
- # Insertar los módulos de seguimiento de la conexión (no es necesario
- # si están en el núcleo).
- insmod ip_conntrack
- insmod ip_conntrack_ftp
- # Crear una cadena que bloquee las conexiones nuevas, excepto si
- # vienen de dentro.
- iptables-N block
- iptables**-A** block -m state --state ESTABLISHED,RELATED -j ACCEPT
- iptables**-A** block -m state --state NEW -i ! ppp0 -j ACCEPT
- iptables**-A** block -j DROP
- # Saltar a esa cadena desde las cadenas INPUT y FORWARD.
- iptables **-A** INPUT -j block
- iptables **-A** FORWARD -j block

4.IPTABLES AVANZADO

4.1- ESPECIFICACIONES DE FILTRADO

4.2- EXTENSIONES A IPTABLES

4.2.1- LAS EXTENSIONES DE COINCIDENCIAS

4.3 ESPECIFICACIÓN DE OBJETIVOS (TARGET)

4.3.1 CADENAS DEFINIDAS POR EL USUARIO

4.3.2 EXTENSIONES A IPTABLES:NUEVOS OBJETIVOS (I)

4.3.3 OBJETIVOS ESPECIALES DE SERIE

4.1- Especificaciones de filtrado (I)

- Especificar las direcciones IP de Origen y Destino:
  - Las direcciones IP de origen («-s». «–source», O «–src») y destino («-d», «–destination», o «–dst») pueden especificarse de varias maneras:
    - IP: 10.14.13.2
    - Nombre: www.linuxiso.org, localhost,…
    - IP/máscara: 192.168.1.2/0
    - IP, máscara por defecto: 0/0 (significaría cualquier ip)
  - Ejemplo:
    - NOTA: «-s 0/0» es redundante en este caso.
    - iptables -A INPUT -s 0/0 -j DROP
- Especificar una inversión
  - Hay muchos indicadores, como «-s» (o «–source») y «-d» («–destination») cuyos respectivos argumentos pueden ir precedidos por «!» (se pronuncia «not» o «no»), para que coincidan con direcciones que NO sean iguales a las proporcionadas.
  - Por ejemplo, «-s ! localhost» coincide con cualquier paquete que no venga de localhost.

4.1- Especificaciones de filtrado (II)

- **Especificar el protocolo**
  - Se puede especificar el protocolo con el indicador «-p» (o «–protocol»).
  - El protocolo puede ser un número (si sabe los valores numéricos) o un nombre en el caso especial de «TCP», «UDP» o «ICMP
  - Puede ir precedido de "!"p.e: –p !TCP especifica paquetes que no sean TCP
- **Especificar la Interfaz**
  - Las opciones «-i» (o «–in-interface») y «-o» (o «–out-interface») especifican el nombre de una interfaz con la que coincidir. Una interfaz es el dispositivo físico por el que entra («-i») O sale («-O») un paquete.
  - Los paquetes que pasan por la regla **INPUT** no tienen un interfaz de salida, con lo que nunca se activará una regla de esta cadena que use «-O». De forma similar, los paquetes que atraviesan **OUTPUT** no tienen interfaz de salida. de manera que ninguna regla que use «-i» en esta cadena funcionará.
  - Sólo los paquetes que pasan por la cadena **FORWARD** tienen a la vez interfaz de entrada, y de salida.
  - Es perfectamente correcto especificar una interfaz que no existe en este momento; la regla no será activada por nada hasta que la interfaz empiece a funcionar. Esto es extremadamente útil con enlaces PPP (le llamada (normalmente la interfaz ppp0) y similares.
  - Como caso especial, un nombre (le interfaz que acabe en **« + »** coincidirá con todas las interfaces (que existan en ese momento o no) cuyo nombre empiece de esa manera. Por ejemplo. para especificar una regla que funcione para todas las interfaces PPP, se podría usar la opción -i ppp+.
  - El nombre de la interfaz puede ir precedido por «!» para coincidir con un paquete que **no** vaya por la(s) interfaz/ces especificada(s).
- **Especificar fragmentos**
  - Cuando un paquete es demasiado grande se fragmenta, el problema es que sólo el primero contiene toda la información de los protocolos, el resto no.
  - Si queremos que una regla se aplique también a los fragmentos debemos especificar "–f", en cualquier caso, en muchos casos se considera una "buena", práctica aplicarlo sólo al primero, ya que el destinatario no será capaz de reensamblarlo.
  - Ejemplo: descartar cualquier fragmento dirigido a 192.168.1.1:
    - iptables -A OUTPUT -f -d 192.168.1.1 -j DROP

4.2- Extensiones a Iptables

- **Extensiones** *a* **iptables**
  - Las extensiones son de dos tipos: nuevos objetivos (targets) y nuevas coincidencias (matches).
  - Algunos protocolos ofrecen de manera automática nuevos tipos de comprobaciones: en la actualidad son TCP, UDP e ICMP
  - Para obtener ayuda sobre una extensión, use la opción que se usa para cargarla («-p». «-j» o «-m») seguida por «-h» o «—help», por ejemplo:
  - **iptables -p tcp --help**

4.2.1- Las extensiones de coincidencias (I)

- **Extensiones TCP**
  - Se cargan de forma automática si se especifica «-p tcp». (ninguna con fragmentos)
  - **--tcp-flags**
    - La primera cadena es la máscara: una lista de los indicadores que desea examinar. La segunda cadena indica cuales deben estar activos.
    - iptables -A INPUT -protocol tcp -tcp-flags ALL SYN,ACK -j DENY
    - Esto indica que deben ser examinados todos los indicadores («ALL» es sinónimo de «SYN,ACK,FIN, RST,URG,PSH»), pero sólo deben estar activos SYN y ACK. Hay otro argumento llamado «NONE», que significa «ningún indicador
  - **syn**
    - equivalente a «--tcp-flags SYN,RST,ACK SYN».
  - **source-port o --sport**
    - después un puerto o rango de puertos TCP,representados por su nombre, tal como viene en /etc/services, o por su número
  - **destination-port ó -- dport**
    - Igual al anterior pero para el puerto de destino
- **Extensiones UDP**
  - Las mismas y de igual funcionamiento que las TCP
- **Extensiones ICMP**
  - icmp-type : seguido de un nombre o un número indicando el código

4.2.1- Extensiones de coincidencias: match (II)

- mac
  - de forma explícita con «-m mac» o «--match *mac».* Se usa para coincidencias en las direcciones Ethernet (MAC) de los paquetes entrantes. Sólo tiene una opción
    - --mac-source: por ejemplo mac-source 00:60:08:91:CC:B7.
    - iptables –A INPUT –m mac --mac-source 00:00:00:00:00:00 –j DROP
- limit
  - Para restringir el número de coincidencias, sobre todo para los mensajes de registro.
  - También puede usar este módulo para evitar varios ataques por denegación de servicio (DoS) con una tasa más rápida para incrementar la respuesta.
    - Protección contra Syn-flood (inundación mediante Syn):
      - iptables **-A** FORWARD -p tcp --syn -m limit --limit 1/s -j ACCEPT
    - Furtivo buscando puertos (port scanner):
      - iptables **-A** FORWARD -p tcp --tcp-flags SYN,ACK,FIN,RST RST -m limit --limit 1/s -j ACCEPT
    - Ping de la muerte:
      - iptables **-A** FORWARD -p icmp --icmp-type echo-request -m limit --limit 1/s -j ACCEPT

4.2.1- Extensiones de coincidencias: match (III)

- State
  - El criterio de coincidencia más útil vienen proporcionado por la extensión «state», que interpreta el análisis de seguimiento de conexión del módulo «ip_conntrack»
  - permite una opción «–state» adicional, que es una lista separada de estados a buscar :
    - **NEW :**Paquete que crea una nueva conexión.
    - **ESTABLISHED :**Paquete que pertenece a una conexión existente (esto es, que tuvo paquetes de respuesta).
    - **RELATED** Paquete que está relacionado a una conexión existente, pero que no es parte de ella. como un error ICMP o (con el módulo de FTP insertado), un paquete que establece una conexión de datos ftp.
    - **INVALID** Paquete que no pudo ser identificado por alguna razón: incluye quedarse sin memoria y errores ICMP que no corresponden a ninguna conexión conocida. Normalmente estos paquetes deberían ser descarta-dos.

4.3 Especificación de objetivos (target)

- Se emplean para decidir qué hacer con los paquetes que se ajustan a nuestras pruebas. A esto se le denomina **objetivo** (target) de una regla.
- Hay dos objetivos implementados en el sistema bastante sencillos: **DROP y ACCEPT**. Si una regla coincide con un paquete, y su objetivo es alguno de estos dos, no se consultarán más reglas: la suerte del paquete ha sido decidida.
- Hay dos tipos de objetivos aparte de los que ya vienen: extensiones y cadenas definidas por el usuario.

4.3.1 Cadenas definidas por el usuario

- El usuario puede crear nuevas cadenas, aparte de las tres que ya vienen implementadas (INPUT. FORWARD y OUTPUT).
- Por convención, las cadenas definidas por el usuario van en minúsculas para distinguirlas.
- Cuando un paquete coincide con una regla cuyo objetivo es una cadena definida por el usuario, el paquete empieza a atravesar esa otra cadena. Si en ella no se decide la suerte del paquete, una vez llegado el final de la cadena, se vuelve al mismo punto desde el que se saltó, a la siguiente regla de la antigua cadena.
- Para crear una nueva cadena usamos «-N» o «—new-chain»:
    - iptables -N prueba

- Consideremos un paquete TCP que venga de 192.168.1.1, y vaya a 1.2.3.4. Entra en la cadena **INPUT,** y pasa se compara con Regla1 (no coincide). Coincide con Regla2, y su objetivo es prueba, de manera que la siguiente regla que se examina es el comienzo (le prueba. La Reglal de prueba coincide, pero no especifica mi objetivo, de manera que se examina la siguiente regla. Regla2. No coincide, de manera que hemos llegado al final de la cadena. Volvemos a INPUT, donde acabábamos
- `INPUT' `prueba'
- Regla1: -p ICMP -j DROP Regla1: -s 192.168.1.1
- Regla2: -p TCP -j prueba 1 Regla2: -d 192.168.1.1
- Regla3: -p UDP -j DROP

4.3.2 Extensiones a iptables: Nuevos objetivos (I)

- **LOG**
  - Este módulo proporciona un registro de paquetes coincidentes mediante el núcleo. Dispone de estas opciones adicionales:
  - **log-level**
    - Seguido de un número (o nombre) de nivel. Los nombres válidos son (distingue mayúsculas/minúsculas) «debug», «info». «notice», «warning», «err», «crit», «alert». «emerg», que corresponden a los números 7 a O. Vea la página de manual de syslog.conf si necesita una explicación de estos niveles.
  - **log-prefix**
    - Seguido de una cadena de hasta 30 caracteres, que corresponden a un texto que se escribe al comienzo de cada «log» (registro), para permitir que sean identificados unívocamente.
    - Este módulo es más útil junto a un objetivo «limit», de manera que no se inunden los archivos de registro.

4.3.2 Extensiones a iptables: Nuevos objetivos (II)

- **REJECT**
  - Este módulo tiene el mismo efecto (que, «DROP», excepto que al remitente se le envía un mensaje de error ICMP «port unreachable». Fíjese que el mensaje de error ICMP no se envía si (vea el RFC 1122):
    - El paquete a filtrar ya era un mensaje de error ICMP o algún tipo desconocido de ICMP.
    - El paquete a filtrar no era un fragmento con la cabecera del paquete original.
    - Hemos enviado demasiados mensajes de error ICMP a ese destino recientemente.
  - REJECT también admite un argumento opcional «–reject-with» que altera el paquete re respuesta a usar

4.3.3 Objetivos especiales de serie

RETURN

- tiene el mismo efecto que si hubiésemos llegado al final de la cadena: si la regla estaba en una cadena de las que hay por defecto, entonces se ejecuta la política de la cadena. Para una regla que esté en una cadena definida por el usuario, se sale de ella. y se continúa atravesando la cadena anterior al salto, justo después de la regla con la que se saltó.

QUEUE

- es un objetivo especial, que pasa al paquete en una cola destinada al procesamiento en espacio de usuario. Para que esto sea útil. hacen falta otros dos componentes:
  - un «controlador de cola» *(queue handler'),* que lleva la mecánica real de paso de paquetes entre el núcleo y el espacio de usuario; y
  - un proceso en espacio de usuario que recibe, posiblemente manipula, y dicta veredicto sobre los paquetes.

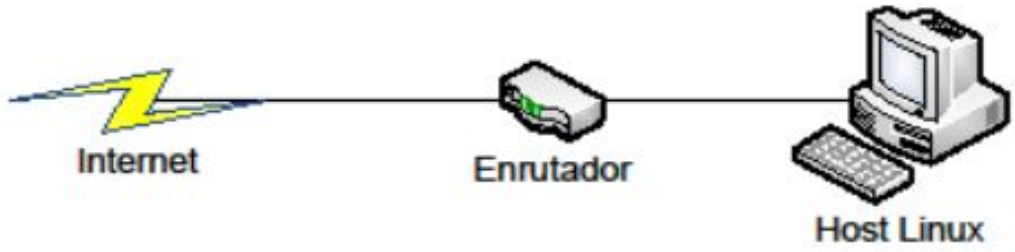
5. Operaciones con una cadena (I)

- **Crear una nueva cadena**
  - Usaremos las opciones «-N» o «—new-chain»:
  - iptables -N prueba
- **Borrar una cadena**
  - Se utilizan las opciones «-X» o «-delete-chain».
  - iptables **-X** prueba
- **Vaciar una cadena**
  - Eliminar todas las reglas de una cadena, usando la orden «-F» (o «—flush»).
  - iptables -F forward
  - Si no especifica una cadena, *todas* serán vaciadas

5. Operaciones con una cadena (II)

- **Listar el contenido de una cadena**
  - Puede listar todas las reglas de una cadena usando la orden «-L» (o «—list»).
  - El «refcnt» mostrado para cada cadena definida por el usuario es el número de reglas que tienen esa cadena como objetivo. Deber ser cero (y la cadena estar vacía) antes de que la cadena pueda ser borrada.
  - Si se omite el nombre de la cadena, se muestran todas, incluso las vacías.
  - Hay tres opciones que pueden acompañar a -L.
    - La opción **«-n»** (numérico) es útil ya que evita que iptables intente averiguar el nombre de las direcciones IP, lo que (si está usando DNS como la mayoría de la gente) puede causar grandes retrasos si el DNS no está configurado adecuadamente, o está filtrando las consultas DNS. También hace que los puertos TCP y UDP aparezcan listados por número, no por nombre.
    - **"-v"** muestra todos los detalles de las reglas: contadores de paquetes y bytes, comparaciones TOS, e interfaces.
    - Los contadores de paquetes y bytes se muestran usando los sufijos k, M, o G, indicando 1000, 1.000.000 y 1.000.000.000. Podemos usar el indicador **"x"** (expandir números) para mostrar los números completos.
- **Restablecer (poner a cero) los contadores**
  - Opcion "-Z" o "-zero"
  - Si queremos saber el valor del contador inmediatamente antes de ser restablecidos podemos usar –L y –Z juntas
- **Establecer la política**
  - Sólo las cadenas del sistema (INPUT, OUTPUT Y FORWARD) pueden tenerla
  - Indican qué hacer con un paquete que llega al final de la cadena definida por el usuario
  - Pueden ser: ACCESP o DROP

6. Configuración de red de host independientes



 # directiva de firewall predeterminada con DROP para todas las cadenas
Iptables -P INPUT DROP
Iptables -P OUTPUT DROP
Iptables -P FORWARD DROP
#Permitir todo el tráfico originado en LinuxFirewall a cualquier lugar de Internet
Iptables -A OUTPUT -i eth0 -s LinuxFirewall -d 0/0 - j ACCEPT
Permitir el tráfico relacionado con el anterior, llamamos a la SPI para ello
Iptables -A INPUT -m state -state ESTABLISHED,RELATED -j ACCEPT
#Permitir acceso SSH al host (eth0 redundante pues sólo hay una interfaz)
Iptables -A INPUT –i eth0 -p tcp -tcp-flags SYN -s 0/0 -d LinuxFirewall –dport ssh –j ACCEPT
Iptables -A OUTPUT –m state -state ESTABLISHED,RELATED
#Permitir acceso HTTP, SMTP y POP3 al host
Iptables -A INPUT -i eth0 -p tcp -tcp-flags SYN -s 0/0 -d LinuxFirewall -dport http -j ACCEPT
Iptables -A INPUT -i eth0 -p tcp -tcp-flags SYN -s 0/0 -d LinuxFirewall -dport smtp -j ACCEPT
iptables -A INPUT -i eth0 -p tcp -tcp-flags SYN -s 0/0 -d LinuxFirewall -dport pop3 -j ACCEPT

6. El fichero del ejemplo anterior

```
#!/bin/bash
##################
#Standalone Host#
##################
#Flush all chains
iptables -F
iptables -t nat –F
# directiva de firewall predeterminada con DROP para todas las cadenas
Iptables -P INPUT DROP
Iptables -P OUTPUT DROP
Iptables -P FORWARD DROP
#Permitir todo el tráfico originado en LinuxFirewall a cualquier lugar de Internet
Iptables -A OUTPUT -i eth0 -s LinuxFirewall -d 0/0 - j ACCEPT
# Permitir el tráfico relacionado con el anterior, llamamos a la SPI para ello
Iptables -A INPUT -m state -state ESTABLISHED,RELATED -j ACCEPT
#Permitir acceso SSH al host (eth0 redundante pues sólo hay una interfaz)
Iptables -A INPUT –i eth0 -p tcp -tcp-flags SYN -s 0/0 -d LinuxFirewall –dport ssh –j ACCEPT
Iptables -A OUTPUT –m state -state ESTABLISHED,RELATED
#Permitir acceso HTTP, SMTP y POP3 al host
Iptables -A INPUT -i eth0 -p tcp -tcp-flags SYN -s 0/0 -d LinuxFirewall -dport http -j ACCEPT
Iptables -A INPUT -i eth0 -p tcp -tcp-flags SYN -s 0/0 -d LinuxFirewall -dport smtp -j ACCEPT
iptables -A INPUT -i eth0 -p tcp -tcp-flags SYN -s 0/0 -d LinuxFirewall -dport pop3 -j ACCEPT
```

7. ¿QUÉ ES NAT?

7.1 ENMASCARAR

7.2 QUÉ PASAR POR NAT

7.3 SELECCIONES USANDO IPTABLES

7.4 CÓMO MODIFICAR LOS PAQUETES

7.4.1 SOURCE NAT (CAMBIO DE ORIGEN)

7.4.2 ENMASCARAMIENTO

7.4.3 DESTINATION NAT (CAMBIO DE DESTINO)

7.4.3.1 REDIRECCIÓN

7.5 PROTOCOLOS ESPECIALES Y DEFECTOS DE NAT

7. ¿Qué es NAT?

- Normalmente, los paquetes viajan en una red desde su origen (por ejemplo su ordenador) a su destino (como por ejemplo www.gnumonks.org) a través de varios enlaces diferentes. Ninguno de estos enlaces altera realmente el paquete: simplemente lo envían un paso adelante.
- Si uno de estos enlaces hiciera NAT, podría alterar el origen o destino del paquete según pasa a través suyo. Como puede imaginar, ésta no es la función para la que se diseñó el sistema, y por tanto NAT es siempre un tanto enrevesado.
- Normalmente, el enlace que esté haciendo NAT recordará cómo modificó el paquete, para hacer la acción inversa con el paquete de respuesta, de manera que todo funciona como se esperaba.
- Razones para usar NAT:
  - Conexiones con módem a Internet
  - Proxy transparente
  - Varios servidores
- Tipos de NAT
  - *Source NAT* es cuando alteramos el origen del primer paquete: esto es, estamos cambiando el lugar de donde viene la conexión. *Source NAT* siempre se hace después del encaminamiento, justo antes de que el paquete salga por el cable. El enmascaramiento es una forma especializada de SNAT.
  - *Destination NAT* es cuando alteramos la dirección de destino del primer paquete: esto es, cambiamos la dirección a donde se dirige la conexión. DNAT siempre se hace antes del encaminamiento, cuando el paquete entra por el cable. El port forwarding (renvío de puerto), el balanceo de carga y el proxy transparente son formas de DNAT.

7.1 Enmascarar

- Si tengo una conexión PPP con IP dinámica (si no sabe lo que es, entonces tiene una), simplemente querré decirle a mi máquina que todos los paquetes que vengan de la red interna deberían aparentar venir de la máquina que tiene el enlace PPP.
- #Cargue el módulo NAT (esto carga también los otros).

modprobe iptable_nat

#Agrega (-A) una regla a la tabla NAT (-t nat), después del encaminamiento (POSTROUTING) para todos los paquetes que salgan por ppp0 (-o ppp0) enmascarando la conexión (-j **MASQUERADE).**

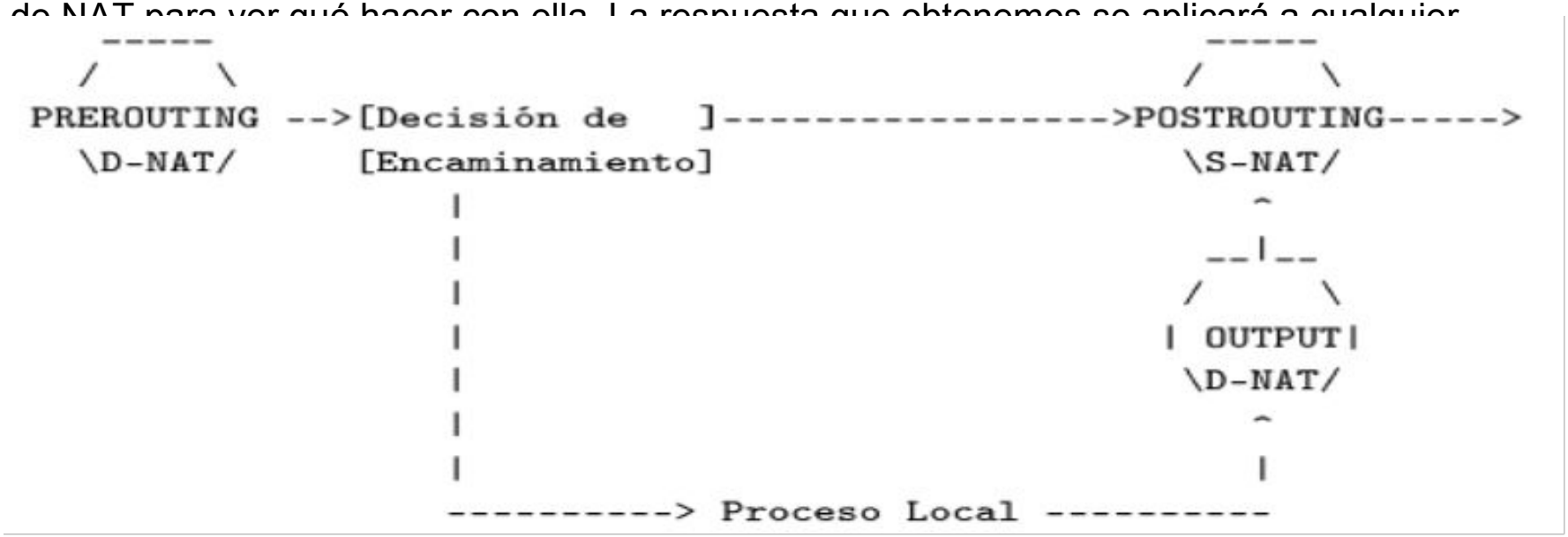
iptables -t nat **-A** POSTROUTING -o ppp0 -j **MASQUERADE**

#Ponga en marcha el reenvío de IP (IP forwarding)

echo 1 > /proc/sys/net/ipv4/ip_forward

7.2 Qué pasar por NAT

- Necesita crear reglas NAT que le digan al núcleo qué conexiones cambiar, y cómo hacerlo. Para ello, usaremos la muy versátil herramienta iptables, y le diremos que altere la tabla de NAT usando la opción «-t nat».
- La tabla de reglas NAT contiene tres listas llamadas «cadenas»: cada regla se examina por orden hasta que una coincide. Las tres cadenas se llaman:
  - PREROUTING (para *Destination NAT,* según los paquetes entran)
  - POSTROUTING (para SOURCE NAT, según los paquetes salen)
  - OUTPUT (para *Destination NAT* con los paquetes generados en la propia máquina).
- En cada uno de los puntos anteriores, cuando un paquete pasa miramos la conexión a la que está asociado. Si es una conexión nueva, comprobamos la cadena correspondiente en la tabla de NAT para ver qué hacer con ella. La respuesta que obtenemos se aplicará a cualquier



7.3 Selecciones usando Iptables

- La opción más importante aquí es la opción de selección de tabla, «-t». Para todas las operaciones de NAT, querrá usar «-t nat» para la tabla NAT. La segunda más importante es «-A» para añadir una nueva regla al final de una cadena («-A POSTROUTING»), o «-I» para insertarla al principio («-I PREROUTING»).
- Puede especificar el origen («-s» o «—source») y el destino («-d» o «—destination») de los paquetes sobre los que quiere hacer NAT. Estas opciones pueden ir seguidas por una IP sencilla (192.168.1.1), un nombre (www.gnumonks.org), o una dirección de red (192.168.1.0/24 o 192.168.1.0/255.255.255.0).
- Puede especificar qué interfaz de entrada («-i» o «—in-interface») o de salida («-o» o «—out-interface») mirar, pero lo que puede especificar depende de en qué cadena esté poniendo la regla:
  - en PREROUTING sólo puede elegir la interfaz de entrada
  - en POSTROUTING (y OUTPUT) sólo la de salida. Si usa la equivocada, iptables le avisará con un mensaje de error.
- Exactamente igual que antes podemos refinar los filtros seleccionando: protocolos, puertos de origen/destino ,….
- Todas las opciones con doble guión pueden ser abreviadas, siempre que iptables pueda distinguirlas de otras opciones posibles

7.4.1 Cómo modificar los paquetes:

Source NAT (Cambio de Origen)

- Cambiar la dirección de origen de las conexiones a algo diferente.
- Esto se hace en la cadena **POSTROUTING**, justo antes de que sea enviado. Este es un detalle importante, ya que significa que cualquier otro servicio de la máquina Linux (encaminamiento, filtrado de paquetes) verá el paquete sin cambiar. También significa que se puede utilizar la opción «-o» (interfaz de salida).
- **El *Source NAT* se especifica** indicando «-j SNAT», y la opción «--to-source» especifica una dirección IP, un rango de direcciones IP, y un puerto o rango de puertos opcionales (sólo con los protocolos UDP y TCP).
- **Por ejemplo**:

```
# Cambiar la dirección de origen por 1.2.3.4
iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 1.2.3.4
# Cambiar la dirección de origen a 1.2.3.4, 1.2.3.5 o 1.2.3.6
iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 1.2.3.4-1.2.3.6
# Cambiar la dirección de origen por 1.2.3.4, puertos 1-1023
iptables -t nat -A POSTROUTING -p tcp -o eth0 -j SNAT --to 1.2.3.4:1-1023
```

7.4.2 Cómo modificar los paquetes:

Enmascaramiento

- Hay un caso especializado de *Source NAT* denominado «enmascaramiento» (masquerading).
- Sólo debería ser usado para direcciones IP asignadas de forma dinámica, tales como las de conexiones por llamada estándar (para direcciones IP estáticas, utilize el SNAT descrito anteriormente).
- No es necesario escribir la dirección de origen de forma explícita con el enmascaramiento: utilizará la dirección de origen de la interfaz por la que el paquete está saliendo. Pero más importante aún, si el enlace cae, las conexiones (que se iban a perder de todas maneras) se olvidan, lo que significa que habrá menos follón cuando la conexión vuelva a la normalidad con una IP diferente.
- Por ejemplo:

```
# Enmascarar todo lo que salga por ppp0.
iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
```

7.4.3 Cómo modificar los paquetes:

Destination NAT (Cambio de destino)

- Esto se hace en la cadena **PREROUTING**, según entra el paquete; esto significa que cualquier otro servicio de la máquina con Linux (encaminamiento, filtrado de paquetes) verá el paquete yendo a su destino «real» (el definitivo).
- Esto significa que se puede utilizar la opción «-i» (interfaz de entrada).
- Para alterar el destino de un paquete generado de forma local (en la máquina que hace el NAT), se debe usar la cadena OUTPUT, pero esto es más inusual.
- ***Destination NAT* se especifica** utilizando «-j DNAT», y la opción «—to-destination» especifica una dirección IP, un rango de direcciones IP, y un puerto o rango de puertos opcionales (sólo para los protocolos UDP y TCP).
- **Ejemplo**
  - # Cambia la dirección de destino por 5.6.7.8
    iptables -t nat **-A** PREROUTING -i ethl -j DNAT --to 5.6.7.8
  - # Cambia la dirección de destino por 5.6.7.8, 5.6.7.9 o 5.6.7.10.
    iptables -t nat **-A** PREROUTING -i ethl -j DNAT --to 5.6.7.8-5.6.7.10
  - # Cambia la dirección de destino del tráfico web por 5.6.7.8, puerto 8080.
    iptables -t nat **-A** PREROUTING -p tcp -dport 80 -i eth1-j DNAT --to 5.6.7.8:8080
  - # Redirige los paquetes locales que van a 1.2.3.4 hacia el dispositivo loopback.
    iptables -t nat **-A** OUTPUT -d 1.2.3.4 -j DNAT --to 127.0.0.1

7.4.3.1 Cómo modificar los paquetes: Redirección

- Hay un caso especializado de *Destination NAT* llamado redirección: es una simple conveniencia que es exactamente lo mismo que hacer DNAT, pero con la dirección de la interfaz de entrada.
- # Envía el tráfico que entra dirigido al puerto 80 (web) a nuestro ## proxy squid (transparente)
- # iptables -t nat **-A** PREROUTING -i eth1 -p tcp --dport 80 -j REDIRECT --to-port 3128

7.5 Protocolos especiales y defectos de NAT

- Protocolos especiales
  - A algunos protocolos no les gusta pasar por NAT. Se deben escribir dos extensiones para cada uno de ellos; uno para el seguimiento de las conexiones de ese protocolo, y otro para el propio NAT.
  - Junto con la distribución de netfilter, hay módulos para el ftp: ip_conntrack_ftp.o e ip_nat_ftp.o. Si inserta estos módulos en el núcleo (o los compila de forma permanente), entonces podrá hacer NAT sobre conexiones FTP. Si no lo hace, entonces sólo podrá utilizar FTP pasivo, e incluso eso no será del todo fiable si hace algo más que un sencillo *Source Nat.*
- Defectos del NAT
  - Si hace NAT sobre una conexión, todos los paquetes que pasen en **ambas** direcciones (dentro y fuera de la red) deberán hacerlo a través de la máquina que hace NAT, ya que si no, no funcionará de forma fiable. En particular, el código de seguimiento de conexiones ensambla los fragmentos, lo que significa que no sólo la conexión no será fiable, sino que los paquetes pueden terminar por no llegar, ya que los fragmentos quedarán retenidos.

8. FIREWALL NAT, CASOS PRÁCTICOS

8.1 UN CASO SENCILLO

8.2 FIREWALL CON REENVÍO DE PUERTOS

8.3 FIREWALL CON DMZ Y PROXY WEB TRANSPARENTE

8.4 VPN IPSEC A TRAVÉS DEL FIREWALL

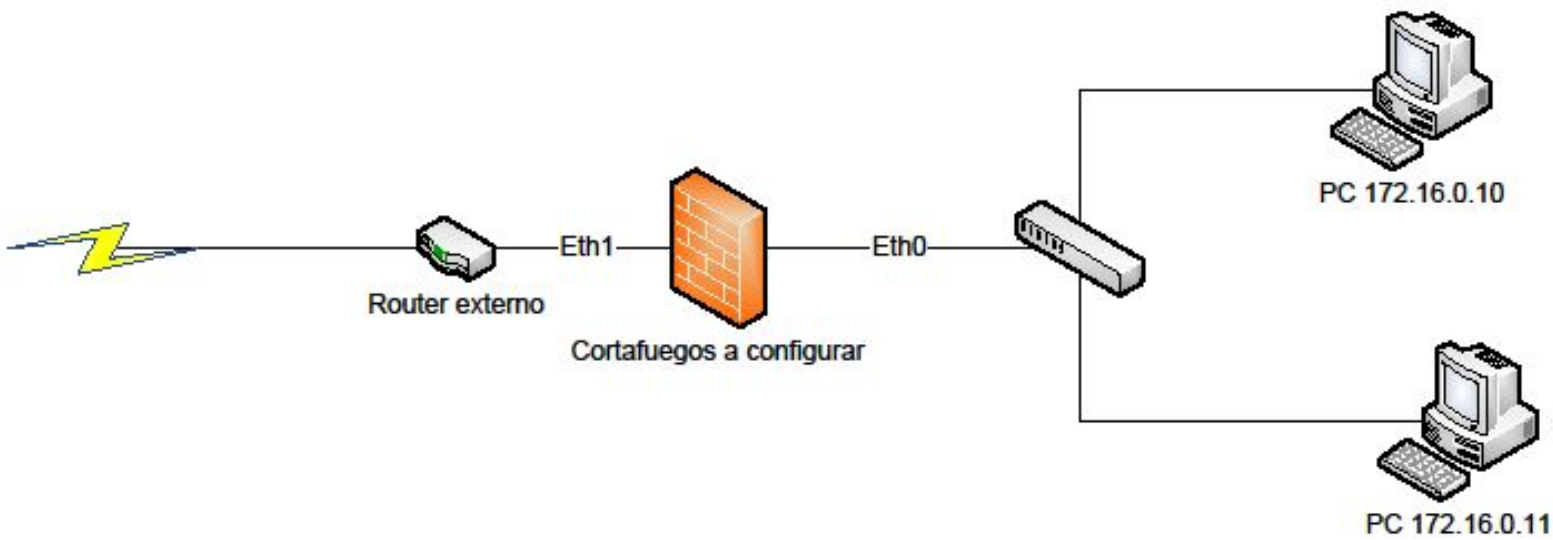
8. Firewall NAT, Casos prácticos

- Iptables puede utilizar tres tipos de NAT:

NAT de origen (SNAT, *Source NAT)*	El firewall destroza paquetes especificados de forma que se modifique su dirección de origen (SOURCE).
NAT de destino (DNAT, *Destination NAT)*	El firewall destroza paquetes especificados de forma que se modifique su dirección de destino (DESTINATION). Esto se utiliza mucho para la redirección de tráfico y el equilibrio de carga.
Enmascaramiento	Este NAT es parecido a SNAT, pero se ha diseñado para utilizarlo con conexiones de acceso telefónico, donde NAT IP es dinámico

- La traducción de direcciones de red se implementa en las cadenas PREROUTING y POSTROUTING.
- Para DNAT, el paquete se modifica tan pronto como sale del cable, por lo que sus reglas se controlan en la cadena PREROUTING.
- Para SNAT, el paquete se modifica inmediatamente antes de llegar al cable, por lo que sus reglas se controlan en la cadena POSTROUTING.
- Estas cadenas residen en una tabla diferente a las cadenas estándar INPUT, OUTPUT y FORWARD, por lo que se requieren opciones adicionales de la línea de comandos para hacer refe-rencia a las cadenas.
- Las cadenas INPUT y OUTPUT han de descartar todos los paquetes ya que los

8.1 Un caso sencillo



#Políticas predeterminadas

Iptables -P INPUT DROP

Iptables -P OUTPUT DROP

Iptables -P FORWARD DROP

8.1 Un caso sencillo (II)

- Esta regla debe hacer referencia explícitamente a la tabla NAT, ya que las cadenas POSTROUTING y PREROUTING no están en la tabla predeterminada.

Iptables -t nat -A POSTROUTING -o eth0 -s 172.16.0.0/24 -d 0/0 -j SNAT -to-source 10.1.1.10 -j ACCEPT

- Esto especifica que todos los paquetes que se originan en la red interna deberán cambiar sus direcciones SOURCE a la dirección IP de la interfaz de red externa del firewall (10.1.1.10) antes de salir del firewall.
- Las siguientes reglas se aplican a los paquetes antes de llegar a POSTROUTING para permitir los paquetes en FORWARD y controlar SPI para los paquetes devueltos:

Iptables -A FORWARD –i eth1 -o etho –s 172.16.0.0/24 -d 0/0 - j ACCEPT

Iptables -A FORWARD –m state –state ESTABLISHED, RELATED – j ACCEPT

8.1 Fichero del ejemplo anterior

```
!/bin/bash
#####################
#Simple NAT Firewall#
#####################
#Vaciar todas las cadenas
iptables -F
iptables -t nat -F
#Establecer políticas por defecto
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
#Hacer Nat de todo el tráfico saliente
Iptables -t nat -A POSTROUTING -o eth0 -s 172.16.0.0/24 -d 0/0 -j SNAT -to-source 10.1.1.10
Iptables -A FORWARD-i ethl -o eth0 -s 172.16.0.0/24 -d 0/0 -j ACCEPT
Iptables -A FORWARD -m state -state ESTABLISHED, RELATED-j ACCEPT
```

8.2 Firewall con reenvío de puertos

PC 172.16.0.11

- Muchas organizaciones carecen de los recursos para crear zonas desmilitarizadas (DMZ, *Demilitarized Zones)* o para mantener utilidades combinadas fuera del sitio; sin embargo, necesitan proporcionar servicios de Internet públicamente disponibles.
- El siguiente ejemplo incluye un segmento LAN interno con un servidor que ejecuta servicios Web y SMTP.
- En este caso, el firewall utiliza DNAT para redirigir el tráfico destinado a puertos Web y SMTP del firewall al servidor interno.
- El diseño agrega más seguridad al ejemplo anterior porque el host del firewall no está ejecutando esos servicios. En este caso, la consolidación de la plataforma es extremadamente importante, ya que cualquier utilización del servicio Web o SMTP abrirá el acceso a la red interna.
- La siguiente tabla describe el tráfico que se permitirá en el firewall con reenvío de puertos:
- **Servicio Puerto abierto**
- Acceso de salida a Internet De SNAT a 10.1.1.10
- Web 80 (de DNAT a 172.16.0.10)
- SMTP 25 (de DNAT a 172.16.0.10)

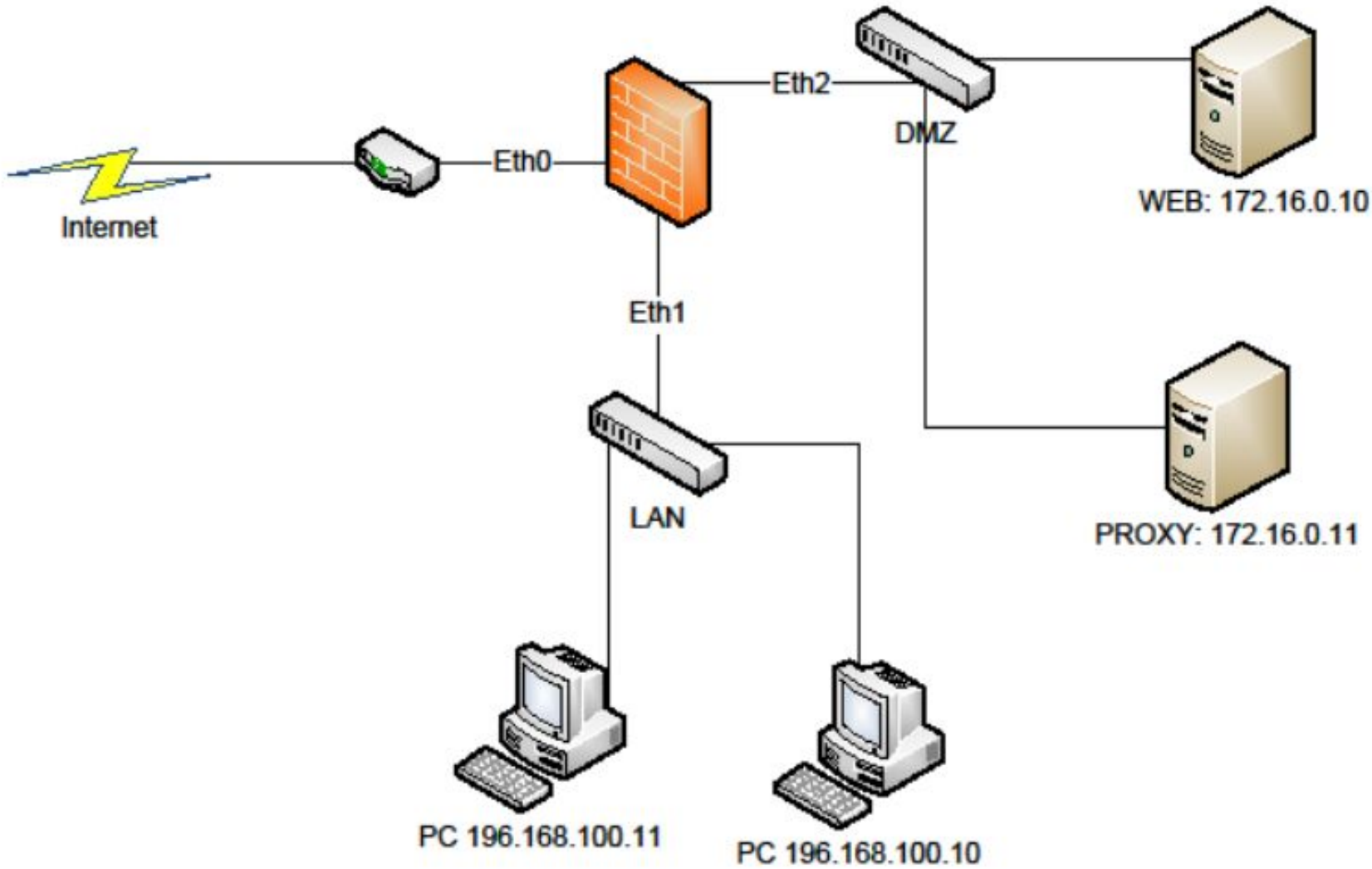
8.2 Firewall con reenvío de puertos (II)

- La siguiente regla indica al núcleo que tome los paquetes con un destino de LinuxFirewall y un puerto de destino de HTTP (80) y, a continuación, que los reenvíe al servidor Web en 172.16.0.10:
- Iptables –t nat -A PREROUTING -p tcp -s 0/0 –d 10.1.1.10 -dport http –j DNAT -to-dest 172.16.10.2 -jACCEPT
- En este caso, la cadena PREROUTING de la tabla NAT salta a la cadena DNAT para traducir los paquetes con una dirección de destino de LinuxFirewall a la dirección del servidor Web (172.16.0.10). En esta configuración, LinuxFirewall sólo expone el servidor Web a Internet en el puerto 80 mientras protege el resto del sistema.
- La siguiente regla realiza el mismo procedimiento para SMTP:
- Iptables -t nat -A PREROUTING -p tcp -s 0/0 -d 10.1.1.10 -dport smtp -j DNAT -to-dest 172.16.10.2 -j ACCEPT
- Cuando los paquetes hayan entrado en la cadena FORWARD, PREROUTING o POSTROUTING los habrán modificado. Si es necesario reenviar los paquetes, tendrán que ser controlados. por su nueva dirección de origen o destino. Las reglas de estado se utilizan para con-trolar dinámicamente los paquetes devueltos, a los que se hace referencia me-diante su dirección «original» (172.16.0.10) y no mediante la dirección NAT (10.1.1.10).

8.2 Fichero del ejemplo anterior (III)

```
#!/bin/bash
##########################################
#Simple NAT Firewall with Port Forwarding#
##########################################
#Flush all chains
iptables -F
iptables -t nat -F
#Set the default policies
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
*NAT all outgoing Internet access
Iptables -t nat -A POSTROUTING -o eth0 -s 172.16.0.0/24 d 0/0 -j SNAT -to-source 10.1.1.10 -j
ACCEPT
Iptables -A FORWARD-i eth1 -o eth0 -s 172.16.0.0/24 -d 0/0 -j ACCEPT
#SPI Rule
Iptables -A FORWARD -m state -state ESTABLISHED, RELATED-j ACCEPT
#Forward SMTP to the internal host
Iptables -t nat -A PREROUTING -p tcp -s 0/0 -d 10.1.1.10 -dport smtp -j DNAT -to-dest 172.16.10.2
#Forward HTTP to the internal host
Iptables -t nat -A PREROUTING -p tcp -s 0/0 -d 10.1.1.10 -dport http -j DNAT -to-dest 172.16.10.2
iptables -A FORWARD -i eth0 -o eth1 -p tcp --dport http -d 172.16.0.10 -j ACCEPT
iptables -A FORWARD -i eth0 -o eth1 -p tcp --dport smpt -d 172.16.0.10 -j ACCEPT
```

8.3 Firewall con DMZ y proxy Web transparente



8.3 Firewall con DMZ y proxy Web transparente (II)

- El modelo de seguridad más habitual para crear infraestructuras de firewall es DMZ.
- Lo más conveniente es que los servidores que proporcionan servicios a redes me-nos seguras, como Internet, se ubiquen en segmentos de red independientes que pasen por un firewall de la red interna. Este diseño mantiene el alto nivel de seguridad de la red interna y permite tráfico menos seguro en los servidores.
- En este ejemplo, LinuxFirewall utilizará tres tarjetas de interfaz de red:
  - Interna (ethl)
  - Externa (eth0)
  - DMZ (eth2). DMZ alojará un servidor Web y la caché de un servidor proxy que controla de forma transparente mediante el proxy todo el tráfico de salida de Internet.
- Tabla resumen del tráfico permitido para el firewall con un proxy Web transparente y DMZ

Servicio Puerto abierto

Web 80 (de DNAT a DMZ 172.16.0.10)

Acceso de salida a Internet (no web) De la red interna SNAT a 10.1.1.10

Acceso de salida a Internet (web) De DNAT 80 a la caché proxy en 3128

- La caché de un proxy como Squid *(http://www.squid-cache .org/),* que se ejecuta en un servidor dedicado en DMZ, se utilizará para contro-lar el acceso Web y optimizar la conexión a Internet. Squid escucha las conexiones entrantes en el puerto TCP 3128. Puesto que las sesiones Web se redirigen de forma transparente, Squid debe estar correctamente configurado para controlar este tráfico.

8.3.1 Configuración (I)

- ***Definir directivas predeterminadas***

  Iptables -P INPUT DROP
  Iptables -P OUTPUT DROP
  Iptables -P FORWARD DROP

- ***Configurar la función de inspección de paquetes de estado***

  Iptables -A INPUT -m state -state ESTABLISHED,RELATED -j ACCEPT
  Iptables -A OUTPUT -m state -state ESTABLISHED,RELATED -j ACCEPT
  Iptables -A FORWARD -m state -state ESTABLISHED,RELATED -j ACCEPT
  Iptables -t nat -A PREROUTING -m state -state ESTABLISHED, RELATED -j ACCEPT
  Iptables -t nat -A POSTROUTING -m state -state ESTABLISHED, RELATED -j ACCEPT

- ***Configuración del tráfico a través de Squid***
  - El siguiente código usa DNAT, para de modo transparente redirigir (forward) el tráfico al proxy Squid:
    Iptables -t nat -A PREROUTING -p tcp -s 192.168.100 0/24 -d 0/0 -dport http -j DNAT -to-dest 172.16.0.11:3128
    Iptables -A FORWARD -p tcp -i ethl -o eth2 -s 192.168.100.0/24 -d 172.16.0.11 –dport 3128 -j ACCEPT
  - La primera regla toma tráfico web del segmento de red interno y lo redirige al servidor Squid en el puerto 3128. La segunda regla asegura que el firewall permitirá el reenvío de los paquetes.
  - Estas dos reglas aseguran que el proxy Squid se puede comunicar a través del DMZ
    Iptables -t nat -A POSTROUTING -p tcp -i eth2 -o eth0-s 172 16.0.11-d 0/0 -j SNAT -to-source 10.1.1.10
    Iptables -A FORWARD -p tcp -i eth2 -o eth0 -s 172.16.0.11-d 0/0 -j ACCEPT

- ***Permitir el acceso de salida de Internet no web***

  Iptables -t nat -A POSTROUTING -s 192.168.100.0/24 -d 0/0 -SNAT -to-source 10.1.1.10
  Iptables -A FORWARD -i ethl -o eth0 -s 192.168.100.0/24 -d 0/0 -j ACCEPT

8.3.1 Configuración (II)

- Aunque la primera y la segunda regla son parecidas a las demás de este ejemplo, se deben tener en cuenta la tercera y cuarta regla. Puesto que muchos servidores Web requieren resolución DNS para sus registros y otras funciones del sitio Web, la tercera y cuarta regla permiten solicitudes DNS UDP y TCP de salida. En el resto de los ejemplos, todo el tráfico de salida se ha ejecutado mediante SNAT, lo que incluye DNS. En DMZ, el tráfico se restringe cuando es posible, por lo que la regla SNAT sólo permite el tráfico DNS que se origina en el servidor Web.

Iptables -t nat -A PREROUTING -p tcp -i eth0 -o eth2 -s 0/0 -d 10.1.1.10 -dport http -j DNAT -to-source 172.16.0.10

Iptables -A FORWARD -p tcp -i eth0 -o eth2 -s 0/0 -d 172.16.0.10 -dport http -j ACCEPT

Iptables -t nat -A POSTROUTING -p udp -s 172.16.0.10 -d DNSserver -dport 53 -j SNAT 10.1.1.10

Iptables -t nat -A POSTROUTING -p tcp -s 172.16.0.10 -d DNSserver -dport 53 -j SNAT 10.1.1.10

Iptables -A FORWARD -p udp -s 172.16.0.10 -d DNSserver -dport 53 -j ACCEPT

Iptables -A FORWARD -p tcp -s 172.16.0.10 -d DNSserver -dport 53 -j ACCEPT

8.3.2 Fichero del ejemplo

```
#!/bin/bash
##########################################*############
# *Simple NAT Firewall with DMZ and Transparent Web Proxy#
###################################*######################
#Flush all chains
iptables -F
iptables -t nat -F
#Set the default policies
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
#Set up Stateful Inspection Rules
Iptables -A INPUT -m state -state ESTABLISHED,RELATED -j ACCEPT
Iptables -A OUTPUT -m state -state ESTABLISHED,RELATED -j ACCEPT
Iptables -A FORWARD -m state -state ESTABLISHED,RELATED -j ACCEPT
Iptables -t nat -A PREROUTING -m state -state ESTABLISHED,RELATED -j ACCEPT
Iptables -t nat -A POSTROUTING -m state -state ESTABLISHED,RELATED j ACCEPT
#NAT all NON-WWW outgoing traffic
Iptables -t nat -A POSTROUTING -s 192.168.100.0/24 -d 0/0 -SNAT -to-source 10.1.1.10
Iptables -A FORWARD -i ethl -o eth0 -s 192.168.100.0/24 -d 0/0 -j ACCEPT
#Transparently redirect outgoing WWW traffic to the Squid proxy
Iptables -t nat -A PREROUTING -p tcp -s 192.168.100.0/24 -d 0/0 - dport http -j DNAT -to-dest 172.16.0.11:3128
Iptables -A FORWARD -p tcp -i ethl -o eth2 -s 192.168.100.0/24 -d 172.16.0.11 -dport 3128 -j ACCEPT
#Allow incoming web requests to the internal web server.
Iptables -t nat -A PREROUTING -p tcp -i eth0 -o eth2 -s 0/0 -d 10.1.1.10 -dport http -j DNAT -to-source 172.16.0.10
Iptables -A FORWARD -p tcp -i eth0 -o eth2 -s 0/0 -d 172.16.0.10 -dport http -j ACCEPT
```

8.4 VPN IPSec a través del firewall

Puerta de enlace VPN IPSec	Firewall	Firewall	Puerta de enlace VPN IPSec
216.100.80.20	216.100.200.250	126.210.106.55	216.210.106.120

- IPSec se ha convertido en un estándar actual para implementar redes privadas virtuales. Los conjuntos de reglas de firewall deben tener en cuenta los tres componentes de tráfico que aparecen en la Tabla para interoperar correctamente con IPSec. La Figura muestra el diseño de red de este ejemplo.
- En este ejemplo, los dos firewalls y sus puertas de enlace VPN son públicamente direccionables. Sólo se requieren cuatro reglas relacionadas con IPSec en cada lado para completar la conexión.
- La primera regla configura SPI, las dos siguientes reglas controlan AH y ESP, y la última regla asegura el paso del tráfico UDP ISAKMP.
- Protocolo Componente de tráfico IPSec

ESP Protocolo 50 RFC IANA

AH Protocolo 51 RFC IANA

ISAKMP Puerto UDP 500

8.4 Reglas VPN

- ***Lado A***
- Se necesitan las siguientes reglas para permitir el tráfico IPSec hacia el lado B:
- Iptables -A FORWARD -m state -state **ESTABLISHED,RELATED** -j ACCEPT
- Iptables -A FORWARD -p 50 -s 126.210.106.120 -d 216.100.80.20 -j ACCEPT
- Iptables -A FORWARD -p 51 -s 126.210.106.120 -d 216.100.80.20 -j ACCEPT
- Iptables -A FORWARD -p udp -s 126.210.106.120 -sport 500 -d 216.100.80.20 -dport 500 -j ACCEPT
- ***Lado B***
- Se requieren las siguientes reglas para permitir el tráfico IPSec hacia el A:
- Iptables -A FORWARD -m state -state **ESTABLISHED,RELATED** -j ACCEPT
- Iptables -A FORWARD -p 50 -s 216.100.80.20 -d 126.210.106.120 -j ACCEPT
- Iptables -A FORWARD -p 51 -a 216.100.80.20 -d 126.210.106.120 -j ACCEPT
- Iptables -A FORWARD -p udp -s 216.100.80.20 -sport 500 -d 126.210.106.120 -dport 500 -j ACCEPT
- Esta sección no dirigirá el paso de IPSec a través de NAT por distintas razones. Puesto que AH y ESP validan la autenticidad de los paquetes por sus direcciones de origen, no funciona debido a la eliminación de paquetes NAT. Cuando se escribieron estas líneas, se estaban revisando dos borradores IETF para solucionar este problema. Las dos soluciones intentan encapsular el tráfico IPSec en un paquete UDP para separar el encabezado IP del encabezado IPSec, permitiendo que NAT se aplique al primero, pero no al segundo

9. Implementación de marcas de tipo de servicio (TOS)

- El encabezado IP contiene un campo diseñado para ayudar a que los enrutadores compatibles tomen decisiones de enrutamiento más inteligentes. Los indicadores TOS son utilizados por los enrutadores para decidir dónde enviar los paquetes; los indica-dores, por sí mismos, no afectan al rendimiento. No todos los enrutadores pueden utilizar indicadores TOS, y los que pueden hacerlo, pueden estar programados para omitirlos.
- **Tabla indicadores TOS Descripción**

Minimize-Delay 16 (0 x 10) Se utiliza cuando es importante limitar la latencia.

Maximize-Throughput 8 (0 x 08) Se utiliza cuando el alto ancho de banda es más importante que la baja latencia.

Maximize-Reliability 4 (0 x 04) Se utiliza cuando es necesario enrutar los paquetes a través de vínculos más fiables.

Minimize-Cost 2 (0 x 02) Se utiliza cuando es necesario enrutar los paquetes a través de vínculos que sean relativamente más baratos.

- Las reglas utilizadas para definir indicadores TOS se insertan en la tabla de «eli-minación». Una regla TOS se inserta en la parte superior de la cadena OUTPUT o PREROUTING. Cuando un paquete coincide con la regla, se define su indicador y, a continuación, se pasa. Las reglas colocadas en la cadena OUTPUT marcan los pa-quetes que procesarán los enrutadores de dirección ascendente. Normalmente, las reglas colocadas en la cadena PREROUTING son procesadas por código de enruta-miento avanzado en el firewall. Estos son dos ejemplos probables de reglas TOS:

Iptables-t mangle-A-ptcp-dport20 -jTOS -set-tos 8

Iptables-t mangle-A-ptcp-dportssh -j TOS -set-tos 16

Iptables-t mangle-A-ptcp-dportsmtp –j TOS -set-tos 2

- La primera regla marca los paquetes de datos FTP para conexiones de ancho de banda superior, posiblemente con latencia más baja. La segunda regla intenta limi-tar la latencia en conexiones SSH. La tercera regla limita el costo de envío y recep-ción de tráfico de correo electrónico.

10. Bibliografía

- Casi toda la documentación ha sido recogida de los manuales de la página: www.netfilter.org (manuales cuya consulta recomiendo)
- Otra buena parte ha sido extraída del libro “firewalls- Manual de Referencia” de la editorial Mc Graw-Hill