

Routers y Firewalls

Tabla de Contenidos

4. Routers y Firewalls.....	2
4.1 Routers	2
Pila de protocolos TCP/IP.....	2
Paquetes de datos.....	3
Comunicación entre ordenadores en una red	4
Cómo funciona un router.....	6
Componentes básicos de un router.....	7
Tipos de routers.....	8
Protocolos de enrutamiento.....	9
4.2 Firewalls.....	10
La red de microsoft windows.....	11
Activando el cortafuegos de Windows 2000	13
Configurando los servicios de red.....	14
Utilizando el Registro de Seguridad.....	17
Habilitando los mensajes ICMP.....	18
Creando un firewall con iptables.....	21
Qué es iptables.....	21
Proteger la propia máquina.....	22
Firewall de una LAN con salida a internet.....	25
Firewall de una LAN con salida a internet con DMZ.....	32
Firewall de una LAN con salida a internet y VPNS.....	38
Firewall puro y duro entre redes.....	41
Firewall con política por defecto DROP.....	44
4.3 Introducción a NAT.....	49
Cómo Funciona NAT.....	50
Activación de NAT.....	51



4. Routers y Firewalls.

4.1 Routers

Un ordenador solitario, sin conexión con ningún otro, es una isla de información y de recursos que no resulta rentable, especialmente cuando para el trabajo diario se precisa recurrir a diferentes fuentes de datos.

De esto se dieron cuenta muy pronto las empresas, que solicitaron a las compañías de desarrollo de hardware y software un medio compartido de trabajo, en el que diferentes estaciones de trabajo, servidores e impresoras pudieran comunicarse entre ellos y compartir recursos. De este modo surgieron las primeras redes de ordenadores.

Una red está formada por una serie de estaciones de trabajo unidas entre sí por medios de transmisión físicos (cables) o basados en ondas (redes inalámbricas), coordinados por unas máquinas especiales, denominadas servidores, y por un conjunto variable de dispositivos de trabajo, como impresoras, escaners, etc. Además, existen diferentes dispositivos que añaden funcionalidades a las redes, como los routers, switches y hubs.

Pila de protocolos TCP/IP

Las diferentes máquinas que forman una red se comunican entre sí usando un medio compartido, pudiendo tener además cada una de ellas características propias, como componentes de hardware, sistemas operativos y aplicaciones de usuario.

Por solventar estas diferencias se hizo necesaria la introducción de una serie de reglas que controlaran el acceso al medio compartido y la forma correcta en que las máquinas se debían comunicar y transmitir los datos, surgiendo con ello diferentes protocolos de comunicación y control.

En un principio, cada empresa desarrolladora implementó un sistema propio de comunicación de red, con una arquitectura y unos protocolos diferentes, por lo que no fue posible, cuando se necesitó, unir redes de diferentes fabricantes. Simplemente, no se entendían entre ellas, pues hablaban "idiomas" diferentes.

Intentando solucionar este problema, la ISO (Organización Internacional de Estándares) creó un modelo de comunicación para redes dividido en una serie de niveles de trabajo, denominados capas, cada uno de los cuales se encargaría de uno o más aspectos concretos de la comunicación mediante una serie de protocolos específicos.

Este modelo se llamó **OSI** (Intercomunicación de Sistemas Abiertos) y, lamentablemente, no llegó a utilizarse en

la práctica, debido a que cuando se publicó ya se habían desarrollado otras arquitecturas de comunicación en redes que funcionaban más o menos bien, y que fueron las que al final se usaron y extendieron.

De ellas, la más conocida y usada en la actualidad es la arquitectura TCP/IP, formada por un extenso conjunto de protocolos, cada uno de los cuales se encarga de un aspecto concreto de la comunicación entre máquinas en red. TCP/IP se basa en un modelo de capas, al igual que OSI, pero más reducido, actuando cada protocolo en una de las capas del mismo.

El número de capas en que se divide TCP/IP y el nombre de las mismas varía según el autor (recordemos que no es un estándar, si no una implementación "de facto"), pero podemos considerar la siguiente división:

- Capa de Aplicación: encargada de dar soporte de red a las aplicaciones de usuario, convirtiendo los datos de estas a un formato estándar apropiado para su transmisión por red. En ella actúan protocolos como HTTP (web), FTP (transferencia de ficheros) y SMTP (correo electrónico).
- Capa de Transporte: encargada de dividir los datos en unidades de información de tamaño apropiado y de controlar la correcta transmisión lógica de las mismas. Sus principales protocolos son TCP y UDP.
- Capa de Internet: su misión principal es enrutar o dirigir los datos de una máquina a otra, usando para ello el protocolo IP, siendo el responsable principal del tráfico de datos entre diferentes redes interconectadas.
- Capa de Enlace de datos: se ocupa de identificar los datos transmitidos entre máquinas de una misma red y de controlar la validez de los mismos tras su emisión y recepción a través del medio físico.
- Capa Física: responsable de la conversión de los datos a transmitir en impulsos eléctricos o en ondas y de su transmisión física.

Cada capa trabaja independientemente de las otras, comunicándose entre ellas por medio de interfaces apropiadas.

Paquetes de datos

Cuando un host desea enviar una serie de datos a otro, estos son convertidos a un formato de red apropiado (capa de Aplicación) y divididos en una serie de unidades, denominadas segmentos (capa de Transporte), que son numerados para su correcto reensamble en la máquina destino.

Posteriormente, son pasados a la capa de Internet, que les coloca las direcciones IP de la máquina origen y de la máquina destino. Las unidades así obtenidas se conocen con el nombre de paquetes. Entonces son pasados a la capa de Enlace de Datos, que les añade las direcciones MAC de ambas máquinas y un número calculado para la verificación posterior de errores en el envío, pasando entonces a denominarse tramas.

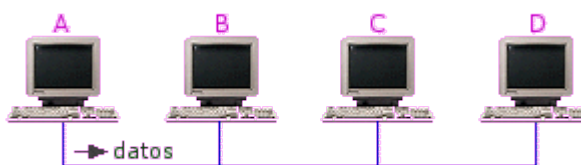
Por último, las tramas son pasadas a la capa Física que las une en trenes de bits apropiados para su transformación en impulsos eléctricos o en ondas, que posteriormente son enviados al medio.

Cuando los impulsos llegan a la máquina destino el proceso se invierte, obteniendo la aplicación receptora los datos en su formato original.

A pesar de que lo que se transmite por el medio físico son impulsos eléctricos, se suele hablar de paquetes transmitidos, ya que son las unidades de información con entidad propia.

Comunicación entre ordenadores en una red

Imaginemos una red formada por varios host, como la representada en la siguiente imagen:



Si el host A (IP=210.23.5.14) se desea comunicar con el host C (IP=210.23.5.27), construye sus paquetes de datos y la capa de Internet les coloca su dirección IP (emisor) y la de C (destinatario), pasándolos a la capa de Enlace de Datos, que no sabe la dirección MAC de C. Para averiguarla, envía un mensaje a todos las máquinas de la red, conocido como petición ARP, preguntando cuál es la dirección MAC correspondiente a la IP 210.23.5.27. Las peticiones ARP son de tipo broadcast, es decir, peticiones que son enviadas a todos y cada uno de los equipos en la red.

La pregunta llega a todas las máquinas, pero sólo C contesta, enviando una respuesta con su dirección MAC. Entonces, A añade ambas direcciones MAC a los paquetes y los pasa a la capa Física, que lo transmite al medio.

Comunicación entre ordenadores en dos redes. Routers.

Imaginemos ahora que el host C (IP=190.200.23.5) no se encuentra en la misma red que A (IP=210.23.5.14). Cuando éste envíe el broadcast preguntando la dirección MAC de C nadie le responderá, por lo que, si no se hace nada al respecto, la comunicación entre ambas máquinas resultará imposible.

Los encargados de solucionar este problema son unos dispositivos de red especiales, llamados routers, que conectan dos o más redes, sirviendo de enlace entre ellas. Los routers trabajan en la capa de Internet, encargándose de encaminar o enrutar paquetes de datos entre máquinas de redes diferentes.

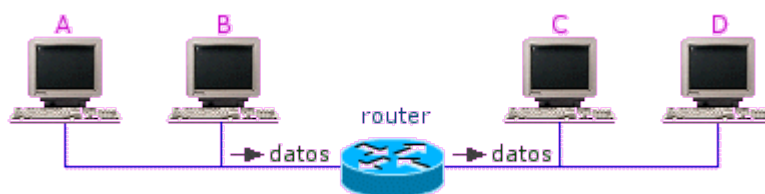




Router Cisco 1601-R

Para poder funcionar de esta forma deben pertenecer a cada una de las redes que conectan, como si fueran un host más de las mismas. De esta forma, un router que conecte dos redes debe tener una tarjeta de red diferente para cada una de las redes y, consecuentemente, dos direcciones MAC diferentes. También debe tener asignada una dirección IP en cada una de las dos redes, ya que si no sería imposible la comunicación con las máquinas de las mismas.

El esquema de dos redes conectadas por un router podría ser el representado en la siguiente imagen:



Ahora, cuando un host envía una petición ARP para averiguar la dirección MAC correspondiente a una IP dada y no es respondido por ningún equipo de su red, envía los paquetes correspondientes a un router que tiene configurado para este tipo de envíos, denominado gateway por defecto.

Una vez que el router recibe los paquetes de datos utiliza un parámetro especial, denominado máscara de red, que sumado lógicamente a la dirección IP destino le da la red a la que pertenece el host buscado. Pasa entonces los paquetes a la red a la que pertenece C, haciendo una nueva petición de broadcast preguntando la MAC de C. Este le responde, y entonces el router le envía los paquetes directamente. Si C desea responder a A, el proceso se invierte.

Este proceso es necesario realizarlo sólo una vez, ya que en esta tanto los host A y C como el router anotan las parejas de direcciones MAC-IP en unas tablas especiales, denominadas tablas de enrutamiento, que usarán en envíos de datos posteriores para enrutar los paquetes directamente.

Resumiendo, los routers son los principales responsables de la correcta comunicación entre máquinas de diferentes redes, encargándose en este proceso de enrutar correctamente los paquetes de datos.

Cómo funciona un router

Los routers son dispositivos de red que raramente se encuentran aislados entre sí. Al contrario, suelen estar interconectados, formando una especie de "telaraña" que hace posible el tráfico de datos entre redes separadas físicamente.

Tomando como ejemplo la Red de redes, Internet, cuando un ordenador envía una serie de paquetes de datos a otro situado en otra ciudad o país, estos son encaminados de router a router a lo largo del camino entre ambas máquinas. Cada paso de un paquete de un router a otro se denomina "salto", y el principal objetivo de todos y cada uno de los routers que intervienen en la transferencia del paquete es que éste llegue a su destino en el menor número posible de saltos, por la mejor ruta posible.

Para poder realizar esta tarea, los routers se comunican constantemente entre sí, informándose de las rutas bloqueadas, de las máquinas intermedias que se encuentran caídas o saturadas de tráfico, aprendiendo con ello cuál es el router idóneo para enviarle los paquetes recibidos.

Si consideramos ahora el caso de un router segmentando una red local (LAN), aunque ahora no debe enviar los paquetes a otro router, sí que tiene que saber por qué puerto debe enviar los datos para que lleguen a la máquina local destino.

Esta habilidad de "saber" a dónde tienen que enviar los paquetes de datos que reciben la consiguen almacenando en su interior una tabla especial, conocida como tabla de ruteo, en la que van anotando las direcciones IP de las máquinas que se comunican con él y el puerto por el que está accesible esa máquina.

Así, cuando a un router llega un paquete, mira en su tabla de ruteo. Si está en ella referenciada la dirección IP de la máquina destino, también lo estará el puerto por el que ésta es accesible, con lo que envía por él el paquete. En caso de no estar la IP en la tabla, manda una petición de respuesta por todos los puertos, preguntando en cuál de ellos se encuentra la máquina destino, y una vez obtenido el puerto de acceso, ingresa la nueva pareja IP/PUERTO en su tabla de ruteo, con lo que los próximos paquetes para esa máquina los enviará directamente.

Podemos buscar una analogía del funcionamiento de los routers con el de las oficinas de correos. Cuando enviamos una carta desde Mérida (Badajoz) a Linares (Jaén), ésta llega en primer lugar a la oficina local de Mérida, que la reenvía a la de Badajoz, que a su vez la manda a la de Jaén, que la remite a la oficina de Linares, que hace la entrega. Si la oficina de correos de Jaén está cerrada por obras, la de Badajoz la enviará a la de Madrid, que la remitirá a la de Andujar, que a su vez se encargará de mandarla a la de Linares, haciendo ésta de nuevo la entrega. Cierre la oficina que cierre, siempre se encontrará un camino para entregar la carta.

Para evitar mantener en su tabla direcciones IP que hayan quedado obsoletas, cada cierto tiempo borra aquellas que no tienen actividad y las que, tras enviarles paquetes, no han respondido. Esto lo consiguen manteniendo "conversaciones" entre ellos, en unos lenguajes especiales denominados "protocolos de enrutamiento".

Destino	Métrica	Interface
210.100.12.0	0	eth0
210.100.12.5	0	eth0
192.80.26.13	8	s0
135.0.25.124	5	s1

Tabla de enrutamiento simple

Componentes básicos de un router

Básicamente, podemos considerar un router como un ordenador especial que funciona solo en las tres primeras capas de la arquitectura TCP/IP, al que se le han eliminado una serie de componentes físicos y funcionalidades lógicas que no necesita para su trabajo, mientras que se le han añadido otros componentes de hardware y de software que le ayudan en su trabajo de enrutamiento.

Como todo ordenador, un router necesita un sistema de arranque (bootstrap), encargado de realizar un chequeo del resto de los componentes antes de pasar el control a un sistema operativo (Cisco IOS, en el caso de los routers Cisco).

El sistema de arranque se almacena en una memoria ROM (Read Only Memory=Memoria de Solo Lectura), junto con una parte básica del sistema operativo, la que toma el control inicialmente, mientras que el cuerpo principal de éste se almacena en una memoria especial, de tipo FLASH, que se puede borrar y reprogramar, permitiendo con ello las actualizaciones necesarias. El contenido de la memoria Flash se conserva en caso de cortes de energía o durante los reinicios del router.

Por otra parte, las funcionalidades operativas de los routers son configurables mediante una serie de instrucciones escritas en un fichero de texto, denominado archivo de configuración, que se almacena en un módulo de memoria de tipo NVRAM (No Volatil RAM), cuyo contenido se conserva durante un corte de energía o si se reinicia el equipo.

Una vez inicializado un router, el fichero de configuración es cargado en una memoria RAM (Random Access Memory=Memoria de Acceso Aleatorio), desde la que se va ejecutando el conjunto de órdenes en él contenido. También se almacenan en esta memoria las tablas de enrutamiento, encargadas de almacenar los puertos del router por los que son accesibles las diferentes máquinas.

Por último, el router posee una serie de puertos o interfaces físicas, puntos de conexión del mismo con las diferentes redes a las que está unido, y a través de los cuales se produce la entrada y salida de datos al equipo. El número de interfaces depende del tipo y funcionalidades del router (y de su precio, claro).



Tipos de routers

Los tipos de router a usar en una red varían dependiendo del tipo de ésta, del número de usuarios y de la función o funciones que deba desempeñar, pudiendo variar mucho la complejidad y el precio de ellos en función del tipo elegido.

Si queremos segmentar nuestra red en diferentes subredes, nos hará falta un router de segmentación, con tantos puertos Ethernet como subredes queremos crear (más los de enlace con otros routers), siendo siempre conveniente que nos sobren puertos, con vista a futuras ampliaciones en la red. Cada subred utilizará luego un hub concentrador o un switch para dar acceso a sus clientes individuales.

Podemos desear un ancho de banda dedicado para un número elevado de equipos individuales, prescindiendo así de los hubs. Necesitaremos entonces un routers de concentración, que precisa aún más puertos Ethernet, aunque no suele ser necesario que sean de alta velocidad de transmisión.

Para conectar una red corporativa a Internet necesitaremos un router de frontera, que actuará como gateway de la red interna, recogiendo todos aquellos paquetes de datos destinados a máquinas externas.

En caso de tener que conectar dos redes WAN o dos segmentos de red en sucursales o campus diferentes, necesitaremos un routers de backbone, que proporciona transporte óptimo entre nodos de la red, con interfaces de alta velocidad que proporcionan un elevado ancho de banda. Generalmente estarán basados en tecnología de fibra óptica.

Por último, también es posible al acceso a redes inalámbrico a redes mediante routers con tecnología wireless, un medio práctico de liberar los equipos de las limitaciones de los cables físicos.





Protocolos de enrutamiento

Hemos visto antes que los routers mantienen unas tablas de enrutamiento, en las que van anotando las direcciones IP de las máquinas destino y los puertos adecuados para darles salida de forma óptima.

Los routers suelen encontrarse interconectados entre ellos, pasándose los paquetes de datos de uno a otro, hasta llegar a la máquina destino. Como cada router tan solo es responsable de las máquinas directamente conectadas a él (incluyendo los routers vecinos), se hace necesario un mecanismo que permita a los routers comunicarse entre sí, para evitar que cada uno tenga en sus tablas registros inválidos.

Esto se consigue por medio de una serie de protocolos de enrutamiento, responsables de que los diferentes routers mantengan sus tablas de enrutamiento acordes, obteniéndose una red convergente. Con ello se consigue, por ejemplo, que si un ordenador o un servidor se apaga en una red, los routers sepan que ya no está accesible, evitando el envío de datos que no llegarán a su destino, y disminuyendo con ello el tráfico de red.

Para mantener las tablas de enrutamiento actualizadas, un router pueden mandar a los routers vecinos una copia de su tabla cada determinado periodo de tiempo (enrutamientos por vector de distancia) y también cuando alguna máquina en su red sufre algún cambio (enrutamiento por estado de enlace). Depende del protocolo de enrutamiento con que funcione.

Existen diferentes protocolos de comunicación entre routers, cada uno de los cuales utiliza mecanismos propios para conseguir la convergencia en la red y para determinar el mejor camino que puede seguir un paquete de datos en su viaje hasta la máquina destino, y cada uno utiliza un sistema de determinación de mejor ruta (métrica) diferente.

Según su misión en una red podemos diferenciar dos tipos principales de protocolos de enrutamiento: los protocolos de gateway interior (IGP), encargados de la comunicación entre routers de una misma red, entre los que destacan RIP e IGRP, y los protocolos de gateway exterior (EGP) o de frontera, encargados de la comunicación entre routers de redes diferentes.

Entre los más importantes protocolos de enrutamiento podemos destacar los siguientes:

- **RIP** (Protocolo de Información de Enrutamiento), es un protocolo de enrutamiento por vector de distancia que calcula las distancias hacia la máquina destino en función de cuántos routers debe atravesar un paquete para llegar a su destino (saltos), enviando cada paquete de datos por el camino que en cada momento muestre una menor distancia. RIP actualiza las tablas de enrutamiento a intervalos programables, generalmente cada 30 segundos. Es un buen protocolo de enrutamiento, pero necesita que constantemente se conecten los routers vecinos, generándose con ello una gran cantidad de tráfico de red.
- **IGRP** (Protocolo de Enrutamiento de Gateway Interior), desarrollado por Cisco System, es un protocolo de enrutamiento por vector de distancia que usa una métrica compuesta basada en diferentes variables de red, como ancho de banda, unidades máximas de transmisión (MTU), confiabilidad, etc. Envía actualizaciones de las tablas de enrutamiento cada 90 segundos.
- **EIGRP** (Protocolo de Enrutamiento de Gateway Interior Mejorado), protocolo mixto basado en IGRP, basado en una métrica de vector distancia, pero que manda actualizaciones de las entradas de las tablas que han cambiado por haber sido alterado el estado de alguna máquina de su red.
- **OSPF**, protocolo puro de estado de enlace, que calcula las rutas más cortas y accesibles mediante la construcción de un mapa de la red y el mantenimiento unas bases de datos con información sobre su sistema local y sobre los vecinos. Cuando una máquina de su sistema cambia, se envía esa entrada de la tabla a los routers vecinos.

El protocolo de enrutamiento a elegir en cada caso depende del tipo de red (LAN, WAN, etc.), de su topología y del uso de la misma, siendo posible en la mayoría de los casos configurar varios protocolos en un mismo router.

4.2 Firewalls

Un sistema básico de seguridad, que debemos utilizar para nuestra conexión a Internet, es la instalación de un Firewall o cortafuegos. Un firewall es un sistema de defensa que se basa en la instalación de una "barrera" entre tu PC y la Red, por la que circulan todos los datos. Este tráfico entre la Red y tu PC es autorizado o denegado por el firewall (la "barrera"), siguiendo las instrucciones que le hayamos configurado.

El funcionamiento de éste tipo de programas se basa en el "filtrado de paquetes". Todo dato o información que circule entre nuestro PC y la Red es analizado por el programa (firewall) con la misión de permitir o denegar su paso en ambas direcciones (Internet-->PC ó PC--->Internet).

El comprender esto último es muy importante, ya que si autorizamos un determinado servicio o programa, el firewall no va a decirnos que es correcto o incorrecto, o incluso, que siendo correcto los paquetes que están entrando o saliendo, éstos contienen datos perniciosos para nuestro sistema o la Red, por lo que hay que tener buen cuidado en

las autorizaciones que otorguemos.

Como ejemplo de esto último podemos poner el Correo Electrónico. Si autorizamos en nuestro firewall a que determinado programa de correo acceda a Internet, y al recibir nuestro correo, en un mensaje recibido viene un adjunto con un virus, por ejemplo tipo gusano, el firewall no nos va a defender de ello, ya que le hemos autorizado a que ese programa acceda a la Red. Lo que si va a hacer es que si al ejecutar el adjunto, el gusano intenta acceder a la Red por algún puerto que no esté previamente aceptado por nosotros, no lo va a dejar propagarse. Ahora bien, si hace uso por ejemplo del mismo cliente de correo, si va a propagarse. La misión del firewall es la de aceptar o denegar el trafico, pero no el contenido del mismo. En éste caso, la misión de protegernos es (además del sentido común de no ejecutar sin más un adjunto) de un programa Antivirus.

La red de microsoft windows

Para que haya comunicación por la red es necesario que en ambos ordenadores haya un programa capaz de entenderse con el del otro lado. Pues bien, el propio windows es ese programa, y tiene la capacidad que puede ser más perjudicial para nosotros: la de compartir ficheros.

La mayoría de vosotros habreis visto alguna vez como se comporta una red de windows 95 o 98: marcamos la carpeta que queremos compartir (aparece la mano debajo de ella) y automáticamente los ficheros de esa carpeta pueden ser leídos o incluso modificados o eliminados desde cualquier otro ordenador de la red, y si estamos conectados a internet, eso equivale a cualquier ordenador del mundo. Siguiendo el símil de las casas y los vecindarios, esto equivale a dejar todas las puertas abiertas de par en par, lo que hacen los ladrones es buscar directamente las empresas con el objetivo de obtener algún beneficio.

En el caso de Windows NT, 2000 o XP, seguimos teniendo un sistema potencialmente peligroso ya que la capacidad para compartir ficheros en red continua siendo inherente al sistema operativo, pero al contrario que en 95-98, ahora cualquier recurso de la red se puede proteger con contraseña, con lo que los ladrones al menos se encuentran con las puertas cerradas. Eso sí, los usuarios poco habituados tienden a dejar las contraseñas en blanco. Esto es especialmente peligroso en el caso del usuario "Administrador", ya que si un pirata se encuentra al entrar en nuestro sistema con que el administrador no tiene contraseña, podrá hacer y deshacer a sus anchas sin que podamos remediarlo.... será como tener una puerta cerrada y entregarle la llave en mano.

Como hemos dicho antes, el sistema de contraseñas es solo una primera barrera, pero no puede considerarse un seguro, ya que de todos son conocidos los múltiples agujeros que en los sistemas operativos de Microsoft son descubiertos a diario, por lo que debemos buscar otras soluciones.

Un firewall funciona, en principio, DENEGANDO cualquier tráfico que se produzca cerrando todos los puertos de nuestro PC. En el momento que un determinado servicio o programa intente acceder a Internet o a nuestro PC nos lo hará saber. Podremos en ese momento aceptar o denegar dicho tráfico, pudiendo asimismo hacer (para no tener que repetir la operación cada vez) "permanente" la respuesta hasta que no cambiemos nuestra política de aceptación.

Una buena política debería ser, ante la duda, no aceptar nunca cualquier acceso hasta comprobar que es necesario para un correcto funcionamiento del servicio que pretendamos usar y no es potencialmente peligroso para el sistema. Si denegamos el acceso y nuestro sistema sigue funcionando bien, no nos es necesario por lo que lo debemos denegar.

Con la instalación de un firewall conseguiremos hacer nuestro sistema mucho menos vulnerable a intrusiones, así que, manos a la obra y vamos a ver los pasos, de forma práctica, para instalarlo. Vamos a mostrar un firewall para Windows y otro para Unix.

Existen 2 tipos de firewalls o cortafuegos, los de hardware (aparatos) y los de software (programas).

Ambos funcionan de la misma manera. En esta guía os enseñaremos como configurarlos ya que cada cortafuegos tiene su propia manera de funcionar y de configurarse, os ayudarán a moveros por ellos y entender como actúan.

Un firewall es como un muro que ponemos delante de la puerta de nuestra casa, de manera que, aunque tengamos la puerta abierta de par en par nadie pueda entrar en ella.

Lo que hace un firewall “filtrar” (no permitir) el paso de información a través de una determinada dirección ip o un determinado puerto. De esta manera, aunque en nuestro ordenador haya un troyano instalado escuchando, si el firewall no deja pasar la información a través de ese puerto, no tenemos de que preocuparnos.

Pueden restringir la comunicación entrante (ordenadores de fuera que intentan acceder al nuestro) o saliente (nuestro ordenador intentando acceder a otros externos).

Básicamente el filtrado se realiza de 2 maneras: por direcciones o por puertos. En ambos casos, cuando se configura un firewall se suelen establecer reglas (rules) .

Una regla dice por ejemplo:

- no dejes salir ninguna información hacia la dirección 217.23.53.23 .
- no dejes entrar ninguna información desde la dirección 215.23.63.23
- no dejes salir ninguna información a través del puerto 80.

o incluso:

- no dejes entrar ninguna información de ninguna dirección por ningún puerto.

pero también reglas positivas:

- sí deja salir la información a través del puerto 21.



Las reglas se ordenan según su importancia, y el firewall las va aplicando por orden, ya que las últimas pueden hacer cosas contrarias a las primeras. Por ejemplo, podemos poner estas dos reglas en nuestro firewall, y él primero aplicará la primera y luego la segunda:

- no dejes salir ninguna información hacia ninguna dirección.
- pero sí deja salir la información hacia la dirección 215.23.63.23

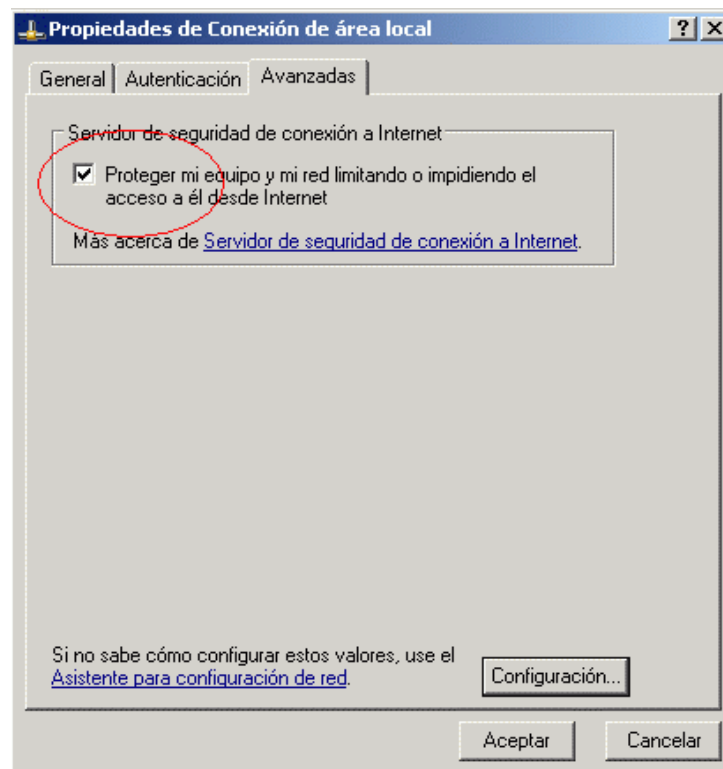
Teniendo claros estos conceptos podremos configurar la mayoría de los firewalls.

En Windows 2000 encontramos en las propiedades del protocolo TCP/IP las opciones “Seguridad IP” y “Filtrado IP”, donde podremos definir nuestras reglas a modo de firewall. En Windows 2000, esta opción se ha transformado en “Servidor de seguridad para conexión a internet” que funciona a la inversa, debemos marcar de una lista las aplicaciones que deseamos que puedan transmitir información hacia en exterior y “se supone” que Windows cerrará el resto de puertos existentes.

Activando el cortafuegos de Windows 2000

Para activar el "Servidor de Seguridad de Conexión a Internet" bastará con ir a las propiedades de la conexión de red que queramos proteger (que puede ser de red local, xDSL, Cable, acceso telefónico, firewire, etc) y en la pestaña "Avanzadas" activar "Proteger mi equipo y mi red limitando o impidiendo el acceso a él desde Internet". A partir de entonces el cortafuegos protegerá nuestro equipo. Sin embargo, es posible que la configuración por defecto interfiera con algunos programas que actúen de servidores. Si servimos un sitio web desde nuestro ordenador, o creamos en un juego una partida a la que esperamos que se unan jugadores, podemos encontrarnos en esta situación.



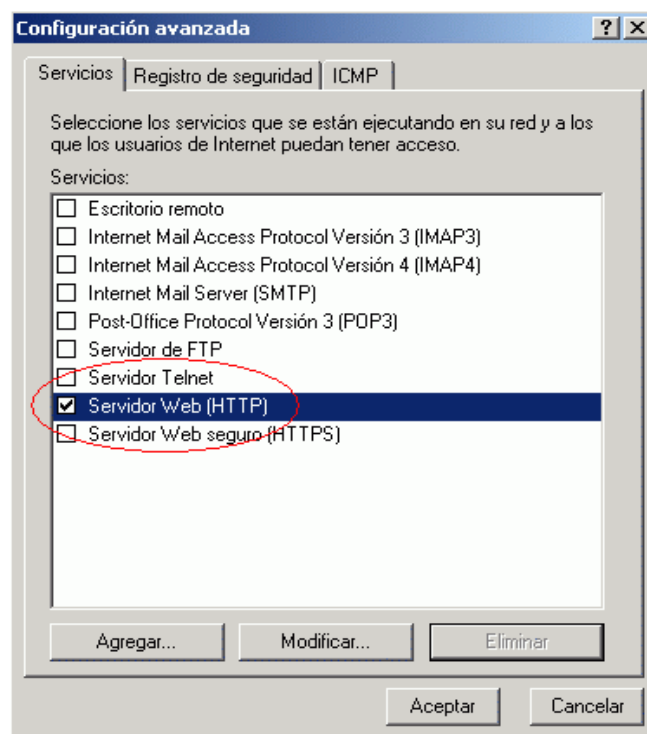


Para que el firewall permita acceder a esos servicios deberemos configurarlo. Para ello, en la misma ventana donde hemos activado el cortafuegos tenemos un botón de configuración, que una vez pulsado nos abrirá una ventana con tres pestañas: Servicios, Registro de Seguridad y ICMP.

Configurando los servicios de red

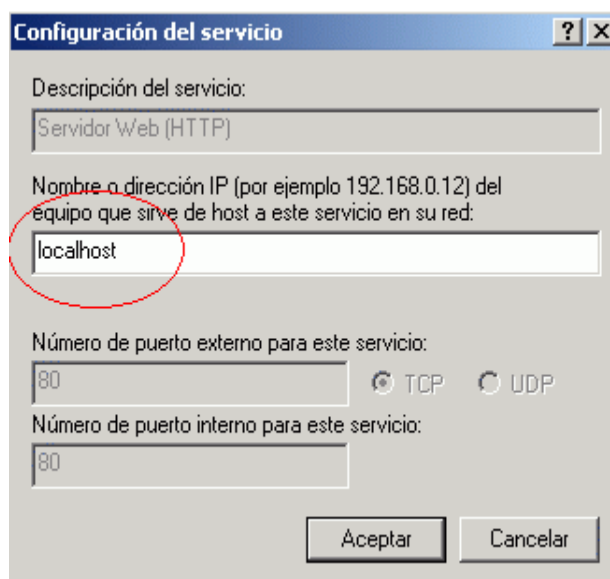
Deberemos añadir a la tabla del cortafuegos aquellos servicios que se ejecuten en nuestra red. Supongamos que tenemos un servidor web con nuestra página personal en nuestro ordenador, que trabaja en el puerto 80. Para permitir que se pueda acceder a la web desde el exterior, deberemos marcar la opción "Servidor Web (HTTP)" en la pestaña Avanzadas:





En la ventana que aparece sólo se nos da opción a introducir el nombre del ordenador que está ejecutando el servidor. Esto se debe a que el servicio no tiene por qué ejecutarse en nuestra propia máquina, sino que puede estar en cualquier otro ordenador de una red local que acceda a través del nuestro a Internet. Si somos nosotros mismos escribiríamos nuestro nombre, o "localhost" (sin las comillas). Si tuviéramos una red con varios ordenadores y se ejecutara en uno que se llame "servidor_web" pondríamos ese nombre en la casilla. De esta forma, indicamos a donde tiene que llevar las peticiones que se hagan al servidor web desde Internet en caso de que no se ejecute en la máquina que corre el cortafuegos.





Configuración del servicio

Descripción del servicio:
Servidor Web (HTTP)

Nombre o dirección IP (por ejemplo 192.168.0.12) del equipo que sirve de host a este servicio en su red:
localhost

Número de puerto externo para este servicio:
80

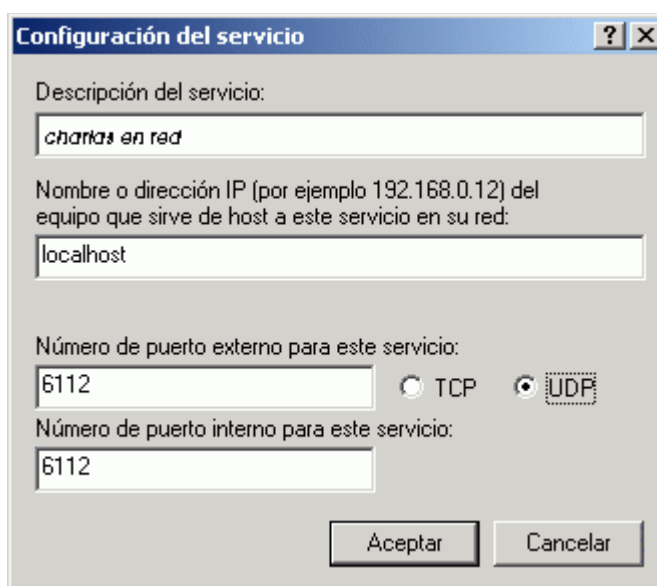
Número de puerto interno para este servicio:
80

☒ TCP ☐ UDP

Aceptar Cancelar

Supongamos además que habitualmente utilizamos un software de mensajería instantánea, y que transmitimos por Internet. Si creamos conversaciones compartidas a las que se debe unir gente, el software de mensajería actúa como un servidor, por lo que también deberemos añadirlo a la tabla del cortafuegos. Para ello deberemos saber en que puerto trabaja. Supongamos que hemos mirado en la ayuda y ésta nos indica que utiliza el puerto 6112 UDP. Pulsando en el botón Agregar nos aparecerá una ventana como la anterior, pero con todas las opciones por rellenar. Lo primero sería poner la descripción del servicio (por ejemplo: charlas en red) y en el siguiente campo el nombre del ordenador que ejecuta el software (si somos nosotros mismos escribiremos de nuevo "localhost" sin las comillas). A continuación pondremos el puerto externo del servicio, es decir, el puerto desde el que será visible la partida desde Internet; y el puerto interno, que debe ser el que esté escuchando el servidor. En nuestro caso, tanto el externo como el interno debe ser el 6112. El primero porque es el que utilizan los clientes para conectarse desde Internet, y el segundo porque es el que escucha el servidor de la conversación compartida. Por último, tendremos que indicar el protocolo utilizado, que en este caso será UDP. Una vez aceptados los parámetros, el juego funcionará correctamente.





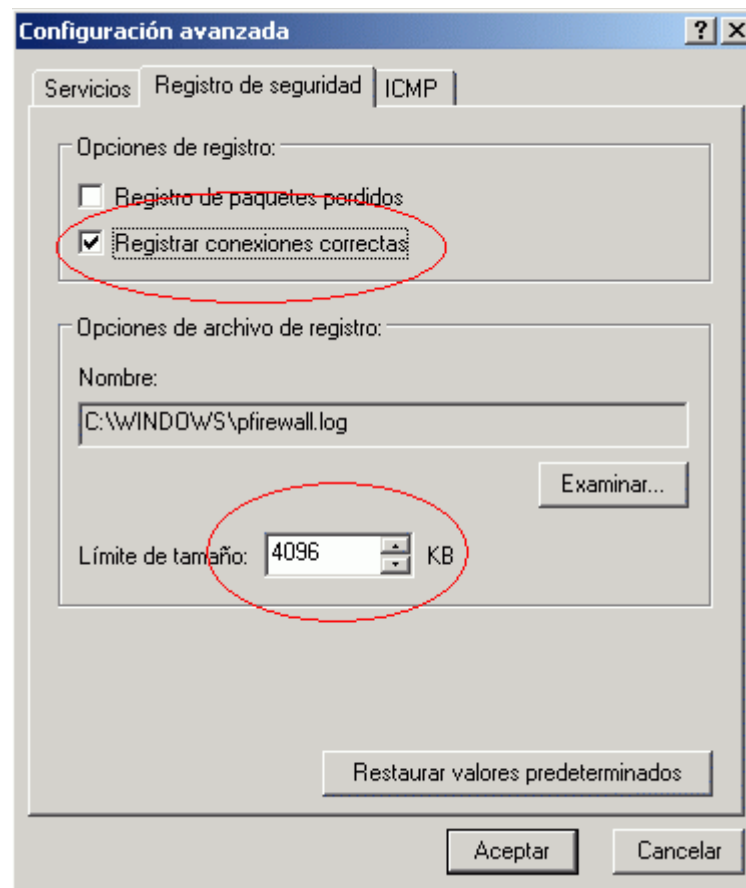
Utilizando el Registro de Seguridad

El cortafuegos de Windows 2000 no muestra mensajes de aviso cuando se produce una incidencia. Teniendo en cuenta la cantidad de alertas que podemos recibir cuando estamos conectados a Internet, sería demasiado molesto. Hay que tener en cuenta que un simple scan de puertos con el fin de comprobar cuales tenemos abiertos podría provocar decenas de incidencias de seguridad en el cortafuegos.

Para evitar los molestos mensajes de aviso se lleva un control de sucesos sobre archivo, que por defecto está situado en "c:\windows\pfirewall.log". Podemos cambiar la ruta y el nombre de archivo desde la pestaña "Registro de Seguridad" y limitar el tamaño máximo del mismo.

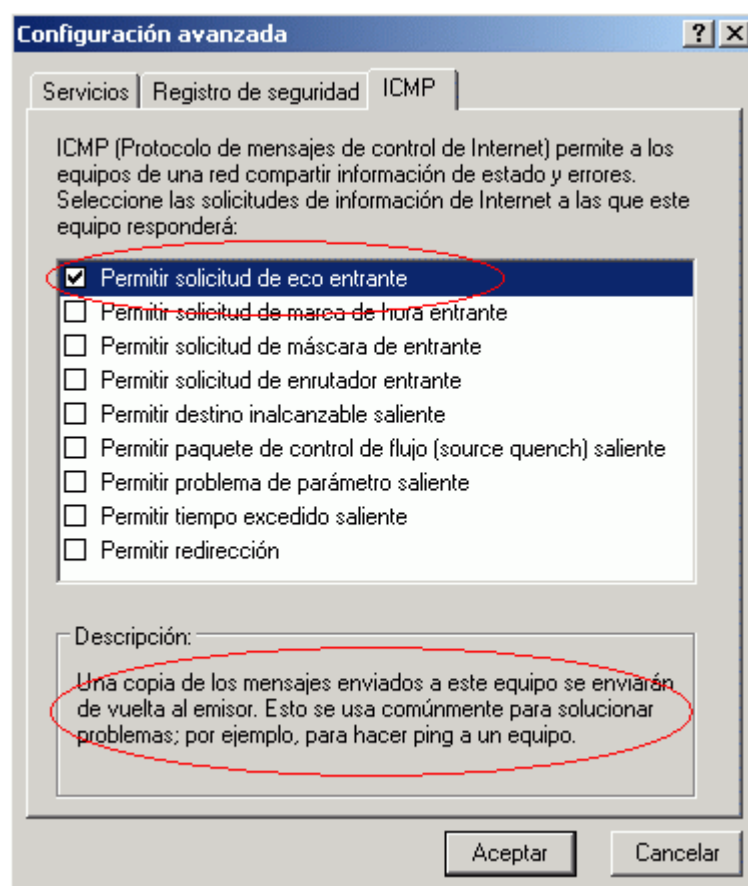
Además podemos registrar también las conexiones correctas con el fin de descubrir posibles puntos de entrada que se nos hayan olvidado proteger debidamente. Bastará con un breve vistazo de forma periódica para identificar accesos no deseados por troyanos y programas similares.





Habilitando los mensajes ICMP

El protocolo ICMP (definido en el documento RFC 792 de especificaciones de red) es utilizado para notificar errores en la conexión, como por ejemplo que parte de la información enviada a una máquina se ha perdido por el camino, o que no podemos llegar a la máquina con la que queremos establecer la conexión. Sin embargo, uno de los mensajes más utilizados de este protocolo es el eco entrante, utilizado para hacer ping y saber si una máquina está conectada.



Por defecto el firewall evita el envío de estos mensajes a la red. De esta forma permaneceremos ocultos ante cualquier usuario que haga ping a nuestra máquina. Sin embargo, si nos interesa que responda a esta petición deberemos marcar la opción "permitir solicitud de eco entrante" en la pestaña ICMP. Esto puede ser útil para comprobar desde otro lugar que el ordenador donde tenemos varios servicios ejecutando sigue respondiendo correctamente. De la misma forma, activaremos el resto de mensajes ICMP que nos puedan interesar. Seleccionando cada uno de ellos aparecerá una pequeña descripción en la parte inferior de la ventana que ayudará a entender el propósito de cada uno.

Configurando Un Firewall Para Unix (iptables)

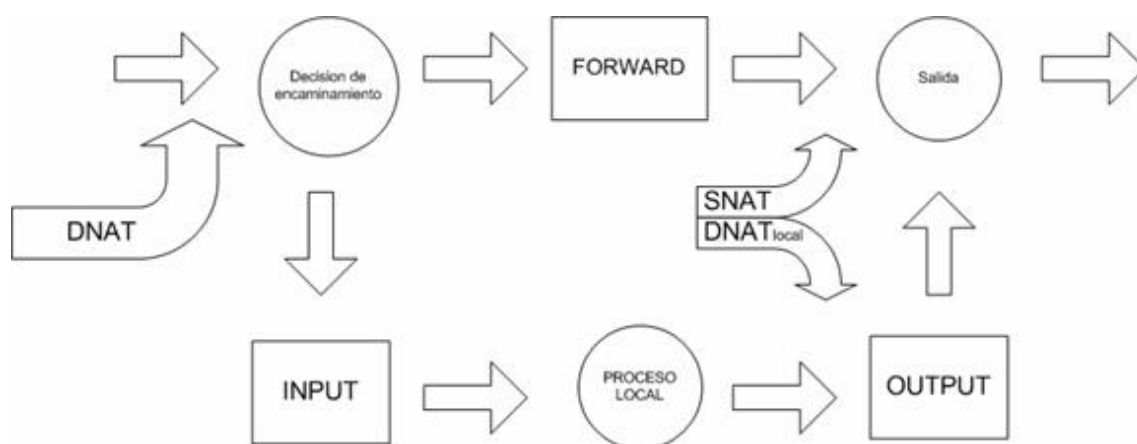
Iptables (también conocido como netfilter) nos permite configurar un Firewall de forma que tengamos controlado quien entra, sale y/o enruta a través de nuestra máquina Unix.

Precisamente estas son las tres reglas básicas de todo Firewall: INPUT, OUTPUT y FORWARD. Aunque con iptables podremos hacer muchas más cosas como: NAT (Network Address Translation), Masquering, Control de ancho de banda, Control según la MAC, evitar ataques DoS, Control Estado (esta es una de las principales novedades respecto a ipchain), etc...

Iptables es un sistema de firewall vinculado al kernel de Linux que se ha extendido enormemente a partir del kernel 2.4 de este sistema operativo. Al igual que el anterior sistema ipchains, un firewall de iptables no es como un servidor que lo iniciamos o detenemos o que se pueda caer por un error de programación (esto es una pequeña mentira, ha tenido alguna vulnerabilidad que permite DoS, pero nunca tendrá tanto peligro como las aplicaciones que escuchan en determinado puerto TCP): iptables está integrado con el kernel, es parte del sistema operativo. ¿Cómo se pone en marcha? Realmente lo que se hace es aplicar reglas. Para ellos se ejecuta el comando iptables, con el que añadimos, borramos, o creamos reglas. Por ello un firewall de iptables no es sino un simple script de shell en el que se van ejecutando las reglas de firewall.

Notas: bueno, para los más geeks y tocapelotas. Vale, se puede implementar un script de inicio en /etc/rc.d/INIT.d (o /etc/INIT.d) con el que hagamos que iptables se "inicie o pare" como un servidor más. Lo podemos hacer nosotros o es probable que venga en la distribución (como en Redhat por ejemplo). También se pueden salvar las reglas aplicadas con el comando iptables-save en un fichero y gestionar ese fichero con una aplicación o front-end desde la X o desde webmin.

Vale, tenemos una máquina Linux con soporte para iptables, tiene reglas aplicadas y empiezan a llegar/salir/pasar paquetes. No nos liemos: olvidemos cuántas tarjetas de red hay, que direcciones IP tiene la máquina y olvidemos si el paquete entra o sale. Las reglas de firewall están a nivel de kernel, y al kernel lo que le llega es un paquete (digamos, un marrón ;)) y tiene que decidir qué hacer con él. El kernel lo que hace es, dependiendo si el paquete es para la propia máquina o para otra máquina, consultar las reglas de firewall y decidir qué hacer con el paquete según mande el firewall. Este es el camino que seguiría un paquete en el kernel:



Cuando un paquete u otra comunicación llega al kernel con iptables se sigue este camino.

Como se ve en el gráfico, básicamente se mira si el paquete esta destinado a la propia maquina o si va a otra. Para los paquetes (o datagramas, según el protocolo) que van a la propia maquina se aplican las reglas INPUT y OUTPUT, y para filtrar paquetes que van a otras redes o maquinas se aplican simplemente reglas FORWARD. INPUT, OUTPUT y FORWARD son los tres tipos de reglas de filtrado. Pero antes de aplicar esas reglas es posible aplicar reglas de NAT: estas se usan para hacer redirecciones de puertos o cambios en las IPs de origen y destino. Veremos ejemplos.

E incluso antes de las reglas de NAT se pueden meter reglas de tipo MANGLE, destinadas a modificar los paquetes; son reglas poco conocidas y es probable que no las usen.

Por tanto tenemos tres tipos de reglas en iptables:

- MANGLE
- NAT: reglas PREROUTING, POSTROUTING
- FILTER: reglas INPUT, OUTPUT, FORWARD.

Creando un firewall con iptables

Vamos al grano y empezamos a ver configuraciones de firewall con iptables, empezando desde la más básica a las más complejas, en las que se establece la denegación como política por defecto.

Nota: se recomienda encarecidamente ir practicando estas reglas en alguna maquina linux disponible, y especialmente hacer uso de la herramienta iptraf para depurar y comprobar el funcionamiento de iptables. Con iptraf podemos comprobar si las conexiones TCP/IP se llegan a establecer o no. Una conexión tcp/ip empieza con el three-way-handshake:

- La maquina que desea conectarse a otra envia un paquete con fln SYN
- Si la otra maquina acepta, envia un SYN/ACK
- Entonces la máquina establece la conexión.

Si el firewall esta denegando la conexión, con iptraf veremos que la maquina origen solo manda paquetes con el fln S (de SYN), y que del otro lado no sale nada. Saber usar iptraf nos ayudará mucho.

Qué es iptables

IPtables es un sistema de firewall vinculado al kernel de linux que se ha extendido enormemente a partir del kernel 2.4 de este sistema operativo. Al igual que el anterior sistema ipchains, un firewall de iptables no es como un servidor que lo iniciamos o detenemos o que se pueda caer por un error de programación(esto es una pequeña

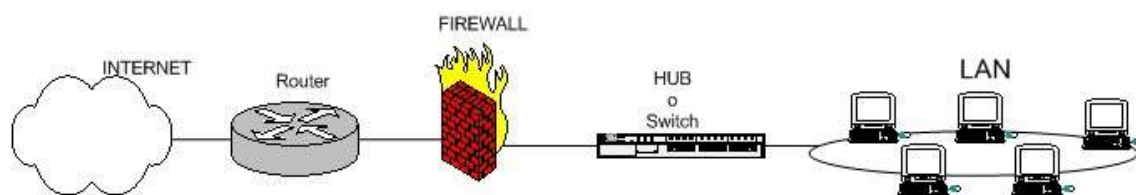
mentira, ha tenido alguna vulnerabilidad que permite DoS, pero nunca tendrá tanto peligro como las aplicaciones que escuchan en determinado puerto TCP): iptables está integrado con el kernel, es parte del sistema operativo. ¿Cómo se pone en marcha? Realmente lo que se hace es aplicar reglas. Para ellos se ejecuta el comando iptables, con el que añadimos, borramos, o creamos reglas. Por ello un firewall de iptables no es sino un simple script de shell en el que se van ejecutando las reglas de firewall.

Notas: bueno, para los más geeks y tocapelotas. Vale, se puede implementar un script de inicio en `/etc/rc.d/INIT.d` (o `/etc/INIT.d`) con el que hagamos que iptables se "inicie o pare" como un servidor más. Lo podemos hacer nosotros o es probable que venga en la distribución (como en redhat por ejemplo). También se pueden salvar las reglas aplicadas con el comando `iptables-save` en un fichero y gestionar ese fichero con una aplicación o front-end desde la X o desde webmin.

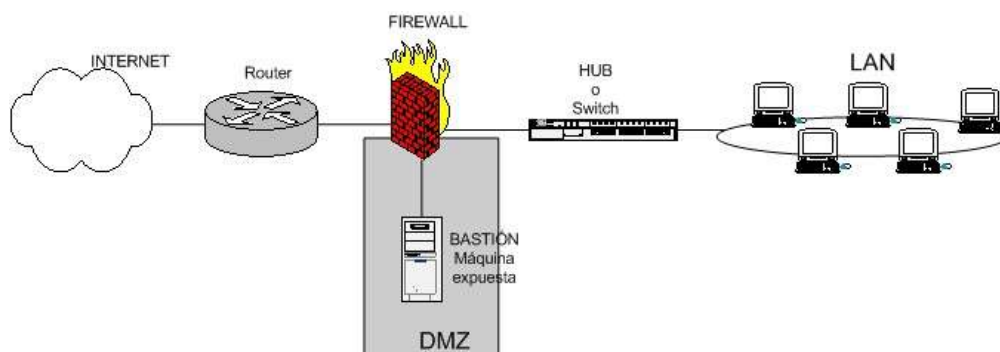
Vale, tenemos una máquina linux con soporte para iptables, tiene reglas aplicadas y empiezan a llegar/salir/pasar paquetes. No nos liemos: olvidemos cuantas tarjetas de red hay, que direcciones ip tiene la máquina y olvidemos si el paquete entra o sale. Las reglas de firewall están a nivel de kernel, y al kernel lo que le llega es un paquete (digamos, un marrón ;)) y tiene que decidir que hacer con él. El kernel lo que hace es, dependiendo si el paquete es para la propia máquina o para otra máquina, consultar las reglas de firewall y decidir que hacer con el paquete según mande el firewall.

Proteger la propia máquina

Muy bien, tenemos una máquina Unix pinchada en internet y queremos protegerla con su propio firewall. Lo único que tenemos que hacer es crear un script de shell en el que se van aplicando las reglas.



esquema de firewall típico entre red local e internet.



esquema de firewall entre red local e internet con zona DMZ para servidores expuestos.

los scripts de iptables pueden tener este aspecto:

- Saludo a la afición (echo)
- Borrado de las reglas aplicadas actualmente (flush)
- Aplicación de políticas por defecto para INPUT, OUPUT, FORWARD
- Listado de reglas iptables.

Ojo con el orden de las reglas!:

```
#!/bin/sh
## SCRIPT de IPTABLES - ejemplo del manual de iptables
## Ejemplo de script para proteger la propia máquina
## indetec Jose maría Molina
## www.indetec.tk - jmmolinafuentes@wanadoo.es
echo -n Aplicando Reglas de Firewall...
## FLUSH de reglas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F
## Establecemos politica por defecto
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT
## Empezamos a filtrar
# El localhost se deja (por ejemplo conexiones locales a mysql)
```

```
/sbin/iptables -A INPUT -i lo -j ACCEPT
# A nuestra IP le dejamos todo
iptables -A INPUT -s 195.65.34.234 -j ACCEPT
# A un conocido le dejamos entrar al mysql para que mantenga la BBDD
iptables -A INPUT -s 231.45.134.23 -p tcp --dport 3306 -j ACCEPT
# A un diseñador le dejamos usar el FTP
iptables -A INPUT -s 80.37.45.194 -p tcp -dport 20:21 -j ACCEPT
# El puerto 80 de www debe estar abierto, es un servidor web.
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
# Y el resto, lo cerramos
iptables -A INPUT -p tcp --dport 20:21 -j DROP
iptables -A INPUT -p tcp --dport 3306 -j DROP
iptables -A INPUT -p tcp --dport 22 -j DROP
iptables -A INPUT -p tcp --dport 10000 -j DROP
echo " OK . Verifique que lo que se aplica con: iptables -L -n"
# Fin del script
```

En fin, ya se ve, un script de los más simple, con unas pocas reglas con las que cerramos puertos al público a los que no tienen porque tener acceso, salvo el 80. Pero cualquiera con algo de ojo se habrá dado cuenta de que ni se filtra el UDP ni el ICMP. Apostaría cualquier cosa a que el sistema tiene algún puerto udp abierto, y además peligroso como el SNMP. Como he dicho anteriormente, en este tipo de firewall es recordable hacer un netstat para ver que puertos están en estado de escucha (abiertos), y salve que un rootkit nos haya modificado los binarios, netstat nos dará la información precisa que necesitamos. Hay gente que se decanta por hacerse un nmap así mismos. Cuidado: dependiendo de cómo lo ejecutemos quizá no nos muestre todos los puertos, ya que suele mirar los bien conocidos.

Imaginemos que hemos dado un repaso a nuestro sistema, y ahora si que tenemos mejor identificados los puertos tcp y udp abiertos. Pero por si acaso nos curamos en salud y al final del script cerraremos el rango de puertos del 1 al 1024, los reservados tanto para tcp como udp.

```
#!/bin/sh
## SCRIPT de IPTABLES - ejemplo del manual de iptables
## Ejemplo de script para proteger la propia máquina
## indetec Jose Maria Molina
## www.indetec.tk - jmmolinafuentes@wanadoo.es
echo -n Aplicando Reglas de Firewall...
## FLUSH de reglas
iptables -F
iptables -X
iptables -Z
```




```
iptables -t nat -F
## Establecemos política por defecto
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT
## Empezamos a filtrar
# El localhost se deja (por ejemplo conexiones locales a mysql)
/sbin/iptables -A INPUT -i lo -j ACCEPT
# A nuestra IP le dejamos todo
iptables -A INPUT -s 195.65.34.234 -j ACCEPT
# A un conocido le dejamos entrar al mysql para que mantenga la BBDD
iptables -A INPUT -s 231.45.134.23 -p tcp --dport 3306 -j ACCEPT
# A un diseñador le dejamos usar el FTP
iptables -A INPUT -s 80.37.45.194 -p tcp -dport 20:21 -j ACCEPT
# El puerto 80 de www debe estar abierto, es un servidor web.
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
# Cerramos rango de los puertos privilegiados. Cuidado con este tipo de
# barreras, antes hay que abrir a los que si tienen acceso.
iptables -A INPUT -p tcp --dport 1:1024
iptables -A INPUT -p udp --dport 1:1024
# Cerramos otros puertos que estan abiertos
iptables -A INPUT -p tcp --dport 3306 -j DROP
iptables -A INPUT -p tcp --dport 10000 -j DROP
iptables -A INPUT -p udp --dport 10000 -j DROP
echo " OK . Verifique que lo que se aplica con: iptables -L -n"
# Fin del script
```

Ahora basta con hacer copy-paste de estas reglas y aplicarlas y ajustarlas en su sistema (quizás uses PostgreSQL). Si tiene miedo de perder el control de una máquina remota, pruebe el script en una máquina local y asegúrese de que aplica lo que usted quiere.

Todo esto y mucho mas, lo podemos consultar en la abundante ayuda y documentacion relativa al iptables en Internet.

Firewall de una LAN con salida a internet

¿Qué es lo que hace falta? Obviamente, una regla que haga NAT hacia fuera (enmascaramiento en iptables), con

lo que se haría dos veces NAT en el firewall y en el router. Entre el router y el firewall lo normal es que haya una red privada (192.168.1.1 y 192.168.1.2 por ejemplo), aunque dependiendo de las necesidades puede que los dos tengan IP pública. El router se supone que hace un NAT completo hacia dentro (quizá salvo puerto 23), o sea que desde el exterior no se llega al router si no que de forma transparente se "choca" contra el firewall. Lo normal en este tipo de firewalls es poner la política por defecto de FORWARD en denegar (DROP), pero eso lo vemos más adelante.

Veamos como sería este firewall-gateway:

```
#!/bin/sh
## SCRIPT de IPTABLES - ejemplo del manual de iptables
## Ejemplo de script para firewall entre red-local e internet
##
## indetec Jose maria Molina
## www.indetec.tk - jmmolinafuentes@wanadoo.es
echo -n Aplicando Reglas de Firewall...
## FLUSH de reglas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F
## Establecemos politica por defecto
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT
## Empezamos a filtrar
## Nota: eth0 es el interfaz conectado al router y eth1 a la LAN
# El localhost se deja (por ejemplo conexiones locales a mysql)
/sbin/iptables -A INPUT -i lo -j ACCEPT
# Al firewall tenemos acceso desde la red local
iptables -A INPUT -s 192.168.10.0/24 -i eth1 -j ACCEPT
# Ahora hacemos enmascaramiento de la red local
# y activamos el BIT DE FORWARDING (imprescindible!!!!)
iptables -t nat -A POSTROUTING -s 192.168.10.0/24 -o eth0 -j MASQUERADE
# Con esto permitimos hacer forward de paquetes en el firewall, o sea
# que otras máquinas puedan salir a través del firewall.
echo 1 > /proc/sys/net/ipv4/ip_forward
## Y ahora cerramos los accesos indeseados del exterior:
```



```
# Nota: 0.0.0.0/0 significa: cualquier red
# Cerramos el rango de puerto bien conocido
iptables -A INPUT -s 0.0.0.0/0 -p tcp -dport 1:1024 -j DROP
iptables -A INPUT -s 0.0.0.0/0 -p udp -dport 1:1024 -j DROP
# Cerramos un puerto de gestión: webmin
iptables -A INPUT -s 0.0.0.0/0 -p tcp -dport 10000 -j DROP
echo " OK . Verifique que lo que se aplica con: iptables -L -n"
# Fin del script
```

Pero tenemos que conseguir que los empleados solamente puedan navegar por internet, denegando el acceso a un software de P2P para descargar archivos. Esta sería una configuración simple pero efectiva.

```
#!/bin/sh
## SCRIPT de IPTABLES - ejemplo del manual de iptables
## Ejemplo de script para firewall entre red-local e internet
## con filtro para que solo se pueda navegar.
## indetec Jose Maria Molina
## www.indetec.tk - jmmolinafuentes@wanadoo.es
echo -n Aplicando Reglas de Firewall...
## FLUSH de reglas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F
## Establecemos politica por defecto
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT
## Empezamos a filtrar
## Nota: eth0 es el interfaz conectado al router y eth1 a la LAN
# El localhost se deja (por ejemplo conexiones locales a mysql)
/sbin/iptables -A INPUT -i lo -j ACCEPT
# Al firewall tenemos acceso desde la red local
iptables -A INPUT -s 192.168.10.0/24 -i eth1 -j ACCEPT
## Ahora con regla FORWARD filtramos el acceso de la red local
## al exterior. Como se explica antes, a los paquetes que no van dirigidos al
## propio firewall se les aplican reglas de FORWARD
```



```
# Aceptamos que vayan a puertos 80
iptables -A FORWARD -s 192.168.10.0/24 -i eth1 -p tcp --dport 80 -j ACCEPT
# Aceptamos que vayan a puertos https
iptables -A FORWARD -s 192.168.10.0/24 -i eth1 -p tcp --dport 443 -j ACCEPT
# Aceptamos que consulten los DNS
iptables -A FORWARD -s 192.168.10.0/24 -i eth1 -p tcp --dport 53 -j ACCEPT
iptables -A FORWARD -s 192.168.10.0/24 -i eth1 -p udp --dport 53 -j ACCEPT
# Y denegamos el resto. Si se necesita alguno, ya avisaran
iptables -A FORWARD -s 192.168.10.0/24 -i eth1 -j DROP
# Ahora hacemos enmascaramiento de la red local
# y activamos el BIT DE FORWARDING (imprescindible!!!!)
iptables -t nat -A POSTROUTING -s 192.168.10.0/24 -o eth0 -j MASQUERADE
# Con esto permitimos hacer forward de paquetes en el firewall, o sea
# que otras máquinas puedan salir a través del firewall.
echo 1 > /proc/sys/net/ipv4/ip_forward
## Y ahora cerramos los accesos indeseados del exterior:
# Nota: 0.0.0.0/0 significa: cualquier red
# Cerramos el rango de puerto bien conocido
iptables -A INPUT -s 0.0.0.0/0 -p tcp -dport 1:1024 -j DROP
iptables -A INPUT -s 0.0.0.0/0 -p udp -dport 1:1024 -j DROP
# Cerramos un puerto de gestión: webmin
iptables -A INPUT -s 0.0.0.0/0 -p tcp -dport 10000 -j DROP
echo " OK . Verifique que lo que se aplica con: iptables -L -n"
# Fin del script
```

Supongamos que este firewall tiene alguna función adicional: es un servidor proxy y además es un servidor de correo. Darle funcionalidades de este tipo a un firewall no es recomendable, porque si no se protegen bien esos puertos o si no está actualizado el software pueden entrar en el firewall a base de exploits comprometiendo TODA la red local. De todas formas muchas empresas no se pueden permitir o no quieren tener una máquina para cada cosa, bastante les cuesta a muchas poner un firewall. Por tanto: si se añaden servicios que deben estar abiertos al público en el propio firewall, nos la estamos jugando, y se recomienda pasar el servicio a otra máquina y ponerla en la DMZ.

Supongamos también que la empresa tiene comerciales en ruta y que se conectan a internet desde su portátil y con una ip dinámica. Supongamos también que el jefe de la empresa quiere acceder a la red local desde casa con una conexión ADSL. Ahora en el firewall debieramos tener instalado un servidor SMTP, pop3, y un PPTPD.

```
#!/bin/sh
## SCRIPT de IPTABLES - ejemplo del manual de iptables
## Ejemplo de script para firewall entre red-local e internet
```



```
## con servicios abiertos de puerto 25, 110, y 1723
## indetec Jose Maria Molina
## www.indetec.tk- jmmolinafuentes@wanadoo.es
echo -n Aplicando Reglas de Firewall...
## FLUSH de reglas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F
## Establecemos politica por defecto
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT
## Empezamos a filtrar
## Nota: eth0 es el interfaz conectado al router y eth1 a la LAN
# El localhost se deja (por ejemplo conexiones locales a mysql)
iptables -A INPUT -i lo -j ACCEPT
# Al firewall tenemos acceso desde la red local
iptables -A INPUT -s 192.168.10.0/24 -i eth1 -j ACCEPT
## Abrimos el acceso a puertos de correo
# Abrimos el puerto 25, hay que configurar bien el relay del servidor SMTP
iptables -A INPUT -s 0.0.0.0/0 -p tcp --dport 25 -j ACCEPT
# Abrimos el pop3
iptables -A INPUT -s 0.0.0.0/0 -p tcp --dport 110 -j ACCEPT
# Y abrimos el puerto pptpd para la ip del adsl de casa del jefe
iptables -A INPUT -s 211.45.176.24 -p tcp --dport 1723 -j ACCEPT
## Ahora con regla FORWARD filtramos el acceso de la red local
## al exterior. Como se explica antes, a los paquetes que no van dirigidos al
## propio firewall se les aplican reglas de FORWARD
# Aceptamos que vayan a puertos 80
iptables -A FORWARD -s 192.168.10.0/24 -i eth1 -p tcp --dport 80 -j ACCEPT
# Aceptamos que vayan a puertos https
iptables -A FORWARD -s 192.168.10.0/24 -i eth1 -p tcp --dport 443 -j ACCEPT
# Aceptamos que consulten los DNS
iptables -A FORWARD -s 192.168.10.0/24 -i eth1 -p tcp --dport 53 -j ACCEPT
iptables -A FORWARD -s 192.168.10.0/24 -i eth1 -p udp --dport 53 -j ACCEPT
```



```
# Y denegamos el resto. Si se necesita alguno, ya avisaran
iptables -A FORWARD -s 192.168.10.0/24 -i eth1 -j DROP
# Ahora hacemos enmascaramiento de la red local
# y activamos el BIT DE FORWARDING (imprescindible!!!!)
iptables -t nat -A POSTROUTING -s 192.168.10.0/24 -o eth0 -j MASQUERADE
# Con esto permitimos hacer forward de paquetes en el firewall, o sea
# que otras máquinas puedan salir a través del firewall.
echo 1 > /proc/sys/net/ipv4/ip_forward
## Y ahora cerramos los accesos indeseados del exterior:
# Nota: 0.0.0.0/0 significa: cualquier red
# Cerramos el rango de puerto bien conocido
iptables -A INPUT -s 0.0.0.0/0 -i eth0 -p tcp -dport 1:1024 -j DROP
iptables -A INPUT -s 0.0.0.0/0 -i eth0 -p udp -dport 1:1024 -j DROP
# Cerramos un puerto de gestión: webmin
iptables -A INPUT -s 0.0.0.0/0 -i eth0 -p tcp --dport 10000 -j DROP
# Y cerramos el puerto del servicio PPTPD, solo abierto para el jefe.
iptables -A INPUT -s 0.0.0.0/0 -i eth0 -p tcp --dport 1723 -j DROP
echo " OK . Verifique que lo que se aplica con: iptables -L -n"
# Fin del script
```

Ahora queremos compartir algún servicio pero de un servidor que tenemos dentro de la red local, por ejemplo el IIS de un servidor windows2000, y además permitir la gestión remota por terminal server para esta máquina para una empresa externa. En este caso lo que hay que hacer es un redirección de puerto. Antes de iptables esto se podía hacer fácilmente con un servidor como rinet. Rinet lo que hace es simplemente abrir un puerto en el firewall y al conectarse a él te lleva hasta el puerto de otra máquina, como una tubería. Con iptables podemos hacer redirecciones con una ventaja: no perdemos la información de IP origen, cosa que con rinet sí ocurría. En fin, veamos la configuración, con las nuevas reglas de DNAT:

```
#!/bin/sh
## SCRIPT de IPTABLES - ejemplo del manual de iptables
## Ejemplo de script para firewall entre red-local e internet
## con servicios abiertos de puerto 25, 110, y 1723
## indetec Jose Maria Molina
## www.indetec.tk - jmmolinafuentes@wanadoo.es
echo -n Aplicando Reglas de Firewall...
## FLUSH de reglas
iptables -F
iptables -X
iptables -Z
```



```
iptables -t nat -F
## Establecemos politica por defecto
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT
## Empezamos a filtrar
## REDIRECCIONES
# Todo lo que venga por el exterior y vaya al puerto 80 lo redirigimos
# a una maquina interna
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j DNAT --to 192.168.10.12:80
# Los accesos de un ip determinada a Terminal server se redirigen e esa
# maquina
iptables -t nat -A PREROUTING -s 221.23.124.181 -i eth0 -p tcp --dport 3389 -j DNAT --to 192.168.10.12:3389
## Nota: eth0 es el interfaz conectado al router y eth1 a la LAN
# El localhost se deja (por ejemplo conexiones locales a mysql)
iptables -A INPUT -i lo -j ACCEPT
# Al firewall tenemos acceso desde la red local
iptables -A INPUT -s 192.168.10.0/24 -i eth1 -j ACCEPT
## Abrimos el acceso a puertos de correo
# Abrimos el puerto 25, hay que configurar bien el relay del servidor SMTP
iptables -A INPUT -s 0.0.0.0/0 -p tcp --dport 25 -j ACCEPT
# Abrimos el pop3
iptables -A INPUT -s 0.0.0.0/0 -p tcp --dport 110 -j ACCEPT
# Y abrimos el puerto pptpd para la ip del adsl de casa del jefe
iptables -A INPUT -s 211.45.176.24 -p tcp --dport 1723 -j ACCEPT
## Ahora con regla FORWARD filtramos el acceso de la red local
## al exterior. Como se explica antes, a los paquetes que no van dirigidos al
## propio firewall se les aplican reglas de FORWARD
# Aceptamos que vayan a puertos 80
iptables -A FORWARD -s 192.168.10.0/24 -i eth1 -p tcp --dport 80 -j ACCEPT
# Aceptamos que vayan a puertos https
iptables -A FORWARD -s 192.168.10.0/24 -i eth1 -p tcp --dport 443 -j ACCEPT
# Aceptamos que consulten los DNS
iptables -A FORWARD -s 192.168.10.0/24 -i eth1 -p tcp --dport 53 -j ACCEPT
iptables -A FORWARD -s 192.168.10.0/24 -i eth1 -p udp --dport 53 -j ACCEPT
# Y denegamos el resto. Si se necesita alguno, ya avisaran
```



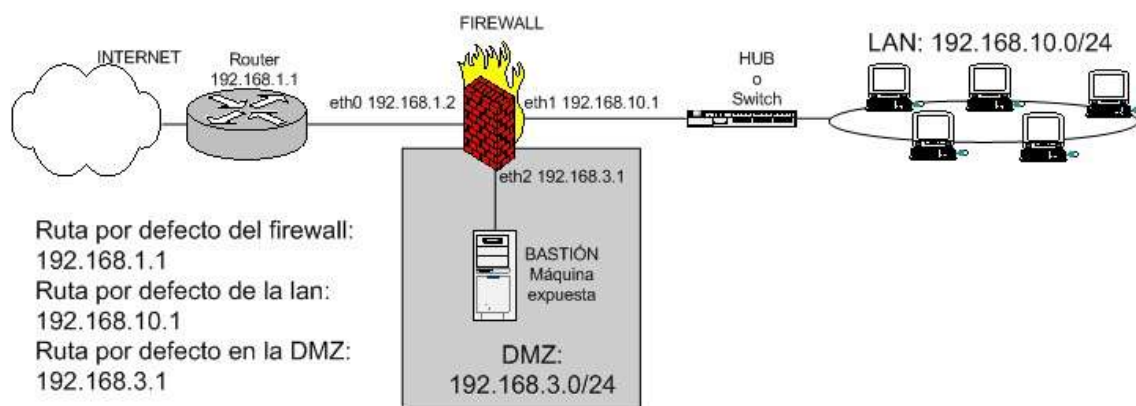
```
iptables -A FORWARD -s 192.168.10.0/24 -i eth1 -j DROP
# Ahora hacemos enmascaramiento de la red local
# y activamos el BIT DE FORWARDING (imprescindible!!!!)
iptables -t nat -A POSTROUTING -s 192.168.10.0/24 -o eth0 -j MASQUERADE
# Con esto permitimos hacer forward de paquetes en el firewall, o sea
# que otras máquinas puedan salir a traves del firewall.
echo 1 > /proc/sys/net/ipv4/ip_forward
## Y ahora cerramos los accesos indeseados del exterior:
# Nota: 0.0.0.0/0 significa: cualquier red
# Cerramos el rango de puerto bien conocido
iptables -A INPUT -s 0.0.0.0/0 -i eth0 -p tcp -dport 1:1024 -j DROP
iptables -A INPUT -s 0.0.0.0/0 -i eth0 -p udp -dport 1:1024 -j DROP
# Cerramos un puerto de gestión: webmin
iptables -A INPUT -s 0.0.0.0/0 -i eth0 -p tcp --dport 10000 -j DROP
# Y cerramos el puerto del servicio PPTPD, solo abierto para el jefe.
iptables -A INPUT -s 0.0.0.0/0 -i eth0 -p tcp --dport 1723 -j DROP
echo " OK . Verifique que lo que se aplica con: iptables -L -n"
# Fin del script
```

Bueno ya tenemos montada la red, pero conviene insistir en que esta última configuración, con las redirecciones y los servicios de correo funcionando en el firewall es bastante insegura. ¿Qué ocurre si hackean el servidor IIS de la red local? Pues que el firewall no sirve de gran cosa, lo poco que podría hacer una vez se ha entrado en la red local es evitar escaneos hacia el exterior desde la máquina atacada, aunque para ello el firewall debiera tener una buena configuración con denegación por defecto. Si necesitamos ese servidor IIS, basta con comprar una tarjeta de red por 6€ o dolares y crear una DMZ.

Firewall de una LAN con salida a internet con DMZ

Bueno, esto se va complicando. Imaginemos que tenemos una red parecida a la anterior pero ahora hacemos las cosas bien y colocamos ese servidor IIS en una DMZ





esquema de firewall entre red local e internet con zona DMZ para servidores expuestos.

En este tipo de firewall hay que permitir:

- Acceso de la red local a internet.
- Acceso público al puerto tcp/80 y tcp/443 del servidor de la DMZ
- Acceso del servidor de la DMZ a una BBDD de la LAN
- Obviamente bloquear el resto de acceso de la DMZ hacia la LAN.

¿Qué tipo de reglas son las que hay que usar para filtrar el tráfico entre la DMZ y la LAN? Solo pueden ser las FORWARD, ya que estamos filtrando entre distintas redes, no son paquetes destinados al propio firewall.

```
#!/bin/sh
## SCRIPT de IPTABLES - ejemplo del manual de iptables
## Ejemplo de script para firewall entre red-local e internet con DMZ
##
## indetec Jose Maria Molina
## www.indetec.tk - jmmolinafuentes@wanadoo.es
echo -n Aplicando Reglas de Firewall...
## FLUSH de reglas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F
## Establecemos politica por defecto
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
```

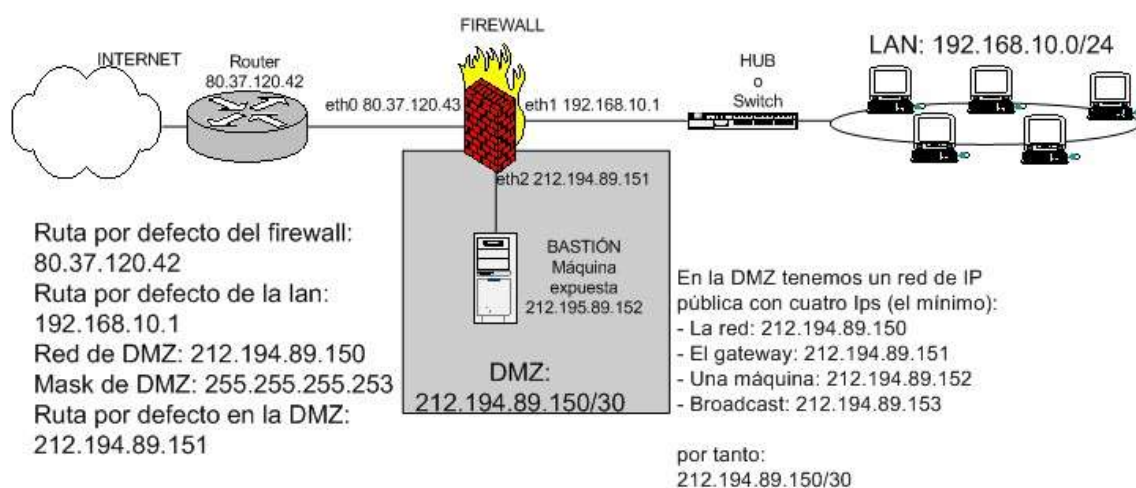
```
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT
## Empezamos a filtrar
## Nota: eth0 es el interfaz conectado al router y eth1 a la LAN
# Todo lo que venga por el exterior y vaya al puerto 80 lo redirigimos
# a una maquina interna
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j DNAT --to 192.168.3.2:80
# Los accesos de un ip determinada HTTPS se redirigen e esa
# maquina
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 443 -j DNAT --to 192.168.3.2:443
# El localhost se deja (por ejemplo conexiones locales a mysql)
/sbin/iptables -A INPUT -i lo -j ACCEPT
# Al firewall tenemos acceso desde la red local
iptables -A INPUT -s 192.168.10.0/24 -i eth1 -j ACCEPT
# Ahora hacemos enmascaramiento de la red local y de la DMZ
# para que puedan salir hacia fuera
# y activamos el BIT DE FORWARDING (imprescindible!!!!)
iptables -t nat -A POSTROUTING -s 192.168.10.0/24 -o eth0 -j MASQUERADE
iptables -t nat -A POSTROUTING -s 192.168.3.0/24 -o eth0 -j MASQUERADE
# Con esto permitimos hacer forward de paquetes en el firewall, o sea
# que otras máquinas puedan salir a través del firewall.
echo 1 > /proc/sys/net/ipv4/ip_forward
## Permitimos el paso de la DMZ a una BBDD de la LAN:
iptables -A FORWARD -s 192.168.3.2 -d 192.168.10.5 -p tcp --dport 5432 -j ACCEPT
iptables -A FORWARD -s 192.168.10.5 -d 192.168.3.2 -p tcp --sport 5432 -j ACCEPT
## permitimos abrir el Terminal server de la DMZ desde la LAN
iptables -A FORWARD -s 192.168.10.0/24 -d 192.168.3.2 -p tcp --sport 1024:65535 --dport 3389 -j ACCEPT
# ? hay que hacerlo en uno y otro sentido ?
iptables -A FORWARD -s 192.168.3.2 -d 192.168.10.0/24 -p tcp --sport 3389 --dport 1024:65535 -j ACCEPT
# ? por que luego:
# Cerramos el acceso de la DMZ a la LAN
iptables -A FORWARD -s 192.168.3.0/24 -d 192.168.10.0/24 -j DROP
## Cerramos el acceso de la DMZ al propio firewall
iptables -A INPUT -s 192.168.3.0/24 -i eth2 -j DROP
## Y ahora cerramos los accesos indeseados del exterior:
# Nota: 0.0.0.0/0 significa: cualquier red
# Cerramos el rango de puerto bien conocido
iptables -A INPUT -s 0.0.0.0/0 -p tcp -dport 1:1024 -j DROP
```



```
iptables -A INPUT -s 0.0.0.0/0 -p udp -dport 1:1024 -j DROP
# Cerramos un puerto de gestión: webmin
iptables -A INPUT -s 0.0.0.0/0 -p tcp -dport 10000 -j DROP
echo " OK . Verifique que lo que se aplica con: iptables -L -n"
# Fin del script
```

Vamos a ver: si las máquinas de la DMZ tienen una ip pública hay que tener muchísimo cuidado de no permitir el FORWARD por defecto. Si en la DMZ hay ip pública NO ES NECESARIO HACER REDIRECCIONES de puerto, sino que basta con rutar los paquetes para llegar hasta la DMZ. Este tipo de necesidades surgen cuando por ejemplo tenemos dos máquinas con servidor web (un apache y un IIS); ¿A cuál de las dos le redirigimos el puerto 80? No hay manera de saberlo (No, con servidores virtuales tampoco, piénsalo), por eso se deben asignar IPs públicas o en su defecto usar puertos distintos.

Por tanto hay que proteger convenientemente toda la DMZ. Tampoco haría falta enmascarar la salida hacia el exterior de la DMZ, si tiene una ip pública ya tiene una pata puesta en internet; obviamente hay que decirle al router como llegar hasta esa ip pública. Así podría ser esta red:



esquema de firewall entre red local e internet con zona DMZ para servidores expuestos usando IPs públicas.

Y este podría ser un firewall adecuado:

```
#!/bin/sh
## SCRIPT de IPTABLES - ejemplo del manual de iptables
## Ejemplo de script para firewall entre red-local e internet con DMZ
## pero con IPs públicas.
## indetec Jose María Molina
```

```
## www.indetec.tk - jmmolinafuentes@wanadoo.es
echo -n Aplicando Reglas de Firewall...
## FLUSH de reglas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F
## Establecemos politica por defecto
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT
## Empezamos a filtrar
## Nota: eth0 es el interfaz conectado al router y eth1 a la LAN
# El localhost se deja (por ejemplo conexiones locales a mysql)
/sbin/iptables -A INPUT -i lo -j ACCEPT
# Al firewall tenemos acceso desde la red local
iptables -A INPUT -s 192.168.10.0/24 -i eth1 -j ACCEPT
# Ahora hacemos enmascaramiento de la red local y de la DMZ
# para que puedan salir hacia fuera
# y activamos el BIT DE FORWARDING (imprescindible!!!!)
iptables -t nat -A POSTROUTING -s 192.168.10.0/24 -o eth0 -j MASQUERADE
# Con esto permitimos hacer forward de paquetes en el firewall, o sea
# que otras máquinas puedan salir a traves del firewall.
echo 1 > /proc/sys/net/ipv4/ip_forward
## Permitimos el acceso desde el exterior a los puertos 80 y 443 de DMZ
iptables -A FORWARD -d 212.194.89.152 -p tcp -dport 80 -j ACCEPT
iptables -A FORWARD -d 212.194.89.152 -p tcp -dport 443 -j ACCEPT
iptables -A FORWARD -d 212.194.89.150/30 -j DROP
## Permitimos el paso de la DMZ a una BBDD de la LAN:
iptables -A FORWARD -s 212.194.89.152 -d 192.168.10.5 -p tcp --dport 5432 -j ACCEPT
# en el otro sentido lo mismo
iptables -A FORWARD -s 192.168.10.5 -d 212.194.89.152 -p tcp --sport 5432 -j ACCEPT
## permitimos abrir el Terminal server de la DMZ desde la LAN
iptables -A FORWARD -s 192.168.10.0/24 -d 212.194.89.152 -p tcp --sport 1024:65535 --dport 3389 -j ACCEPT
# ? hay que hacerlo en uno y otro sentido ?
iptables -A FORWARD -s 212.194.89.152 -d 192.168.10.0/24 -p tcp --sport 3389 --dport 1024:65535 -j ACCEPT
```

```
# ? por que luego:
# Cerramos el acceso de la DMZ a la LAN
iptables -A FORWARD -s 212.194.89.152 -d 192.168.10.0/24 -j DROP
## Cerramos el acceso de la DMZ al propio firewall
iptables -A INPUT -s 212.194.89.152 -i eth2 -j DROP
## Y ahora cerramos los accesos indeseados del exterior:
# Nota: 0.0.0.0/0 significa: cualquier red
# Cerramos el rango de puerto bien conocido
iptables -A INPUT -s 0.0.0.0/0 -p tcp -dport 1:1024 -j DROP
iptables -A INPUT -s 0.0.0.0/0 -p udp -dport 1:1024 -j DROP
# Cerramos un puerto de gestión: webmin
iptables -A INPUT -s 0.0.0.0/0 -p tcp -dport 10000 -j DROP
echo " OK . Verifique que lo que se aplica con: iptables -L -n"
# Fin del script
```

ATENCIÓN

Merece la pena pararse a explicar esta parte del firewall:

```
## permitimos abrir el Terminal server de la DMZ desde la LAN
iptables -A FORWARD -s 192.168.10.0/24 -d 212.194.89.152 -p tcp -sport 1024:65535 --dport 3389 -j ACCEPT
# ? hay que hacerlo en uno y otro sentido ?
iptables -A FORWARD -s 212.194.89.152 -d 192.168.10.0/24 -p tcp --sport 3389 --dport 1024:65535 -j ACCEPT
# ? por que luego:
# Cerramos el acceso de la DMZ a la LAN
iptables -A FORWARD -s 212.194.89.152 -d 192.168.10.0/24 -j DROP
```

Lo que nos lleva a dos cuestiones:

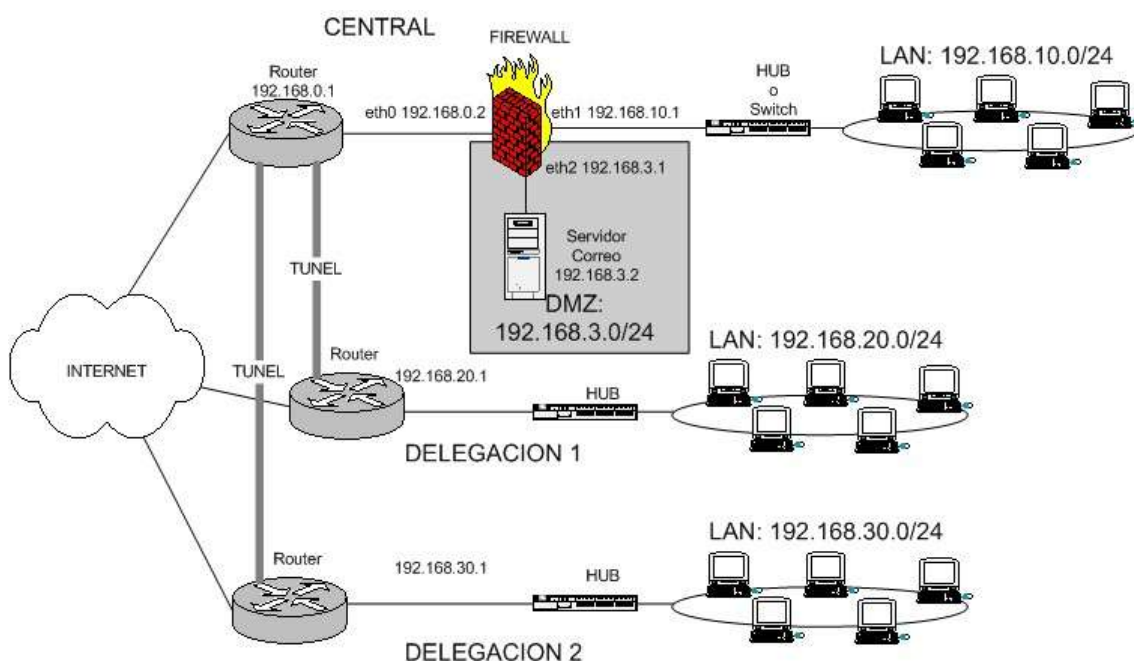
1. ¿Por qué hay que explicitar la abertura en uno y otro sentido? Porque la tercera regla cierra todo lo que va de la DMZ a la red local. Para abrir el puerto 3389 de tcp es imprescindible que un paquete de ida sea capaz de llegar hasta la DMZ y que a su vez pueda volver a la LAN. Esto de tener que especificar la abertura en uno y otro sentido será el pan de cada día en un iptables con política DROP por defecto: mejor protección pero más trabajo.
2. ¿Por qué se explicita el puerto de origen/destino 1024:65535 en la primera y segunda regla? Imaginemos que un hacker logra acceso a la máquina de la DMZ. Si no especificamos el puerto de destino en esas dos reglas, el hacker puede abrir CUALQUIER puerto de la LAN siempre que pueda establecer como puerto origen suyo el tcp/3389, cosa fácil para un hacker que sepa algo de C o que tenga el programa pertinente a

mano. De todas formas el hacker tendría que saber que existe ese tipo de reglas, si es listo probara con puertos de gestión o con puertos netbios. El problema es que se deja un vínculo con la LAN bien para administrarlo remotamente o para establecer relaciones de confianza y ahí es donde reside el peligro.

En las conexiones "legales" no se usa como puerto origen nada por debajo del 1024; cuando alguien se conecta a otro puerto en su extremo abre un puerto por encima del 1024. Especificándolo en la regla de firewall protegeremos un poco mejor la LAN, aunque los puertos por encima de 1024 estarán en peligro.

Firewall de una LAN con salida a internet y VPNS

En principio este caso no nos tendría que dar mayor problema, aunque la primera vez que lo montemos, el enmascaramiento nos jugará una mala pasada. Por eso conviene echar un vistazo en este caso.



esquema de firewall entre red local e internet con zona DMZ y delegaciones que acceden a DMZ.

Supongamos que entre los routers ya se ha establecido un tunel (con Ciscos se haría creando un interfaz Tunnel), y que si el firewall nos deja podríamos llegar de la central a las delegaciones y viceversa usando las IPs privadas. Vaya que se puede hacer un ping desde la central a 192.168.30.x y nos responde. Para ello es imprescindible que el router de la central tenga una ruta metida para llegar a 192.168.10.0/24 y por supuesto cada una ruta para cada delegación. Antes de meterse en el firewall hay que asegurar la visibilidad entre los routers y poder llegar a sus IPs privadas haciendo ping.

Supongamos también que en la central esta el servidor de correo que lógicamente debe tener el puerto 25 accesible desde internet, y debe ser accesible desde las delegaciones para puerto 25, 110 (pop3) o 143(imap). La salida a internet (web, ftp, etc..) cada uno la hace por su lado.

Veamos una posible configuración para este caso.

```
#!/bin/sh
## SCRIPT de IPTABLES - ejemplo del manual de iptables
## Ejemplo de script para firewall entre red-local e internet con DMZ
## y delegaciones. Las delegaciones deben tener acceso al correo de la DMZ
##
## indetec Jose Maria Molina
## www.indetec.tk - jmmolinafuentes@wanadoo.es
echo -n Aplicando Reglas de Firewall...
## FLUSH de reglas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F
## Establecemos politica por defecto
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT
## Empezamos a filtrar
## Nota: eth0 es el interfaz conectado al router y eth1 a la LAN
# Todo lo que venga por el exterior y vaya al puerto 25 lo redirigimos
# a la maquina de la DMZ
iptables -t nat -A PREROUTING -i eth0 \
-p tcp --dport 25 -j DNAT --to 192.168.3.2:25
# Todo lo que venga por el interfaz del router(eth0) y vaya al 110
# siempre que sea una delegacion se acepta y redirije
iptables -t nat -A PREROUTING -s 192.168.20.0/24 -i eth0 \
-p tcp --dport 110 -j DNAT --to 192.168.3.2:110
iptables -t nat -A PREROUTING -s 192.168.30.0/24 -i eth0 \
-p tcp --dport 110 -j DNAT --to 192.168.3.2:110
# Todo lo que venga por el interfaz del router(eth0) y vaya al 110
# siempre que sea una delegacion se acepta y redirije
```



```
iptables -t nat -A PREROUTING -s 192.168.20.0/24 -i eth0 \
-p tcp --dport 143 -j DNAT --to 192.168.3.2:143
iptables -t nat -A PREROUTING -s 192.168.30.0/24 -i eth0 \
-p tcp --dport 143 -j DNAT --to 192.168.3.2:143
# El localhost se deja (por ejemplo conexiones locales a mysql)
iptables -A INPUT -i lo -j ACCEPT
# Al firewall tenemos acceso desde la red local
iptables -A INPUT -s 192.168.10.0/24 -i eth1 -j ACCEPT
# Ahora hacemos enmascaramiento de la red local y de la DMZ
# para que puedan salir hacia fuera
# y activamos el BIT DE FORWARDING (imprescindible!!!!)
# Cuidado con este enmascaramiento.
iptables -t nat -A POSTROUTING -s 192.168.10.0/24 -o eth0 -j MASQUERADE
iptables -t nat -A POSTROUTING -s 192.168.3.0/24 -o eth0 -j MASQUERADE
# Con esto permitimos hacer forward de paquetes en el firewall, o sea
# que otras máquinas puedan salir a través del firewall.
echo 1 > /proc/sys/net/ipv4/ip_forward
# Para que desde la red local se salga hacia fuera hay que ENMASCARAR
# pero que pasa con las delegaciones también están fuera Y NO HAY QUE
# ENMASCARAR, debemos meter una regla FORWARD explícita para que no enmascare
# porque si no una petición de la LAN a otra delegación no se metería
# en el túnel.
iptables -A FORWARD -s 192.168.10.0/24 -d 192.168.20.0/24 -j ACCEPT
iptables -A FORWARD -s 192.168.20.0/24 -d 192.168.10.0/24 -j ACCEPT
iptables -A FORWARD -s 192.168.10.0/24 -d 192.168.30.0/24 -j ACCEPT
iptables -A FORWARD -s 192.168.30.0/24 -d 192.168.10.0/24 -j ACCEPT
# Abrimos el acceso para que se pueda acceder a la DMZ desde la LAN
# a puertos de correo
# En principio lo que va de LAN -> DMZ se acepta
iptables -A FORWARD -s 192.168.10.0/24 -d 192.168.3.0/24 -j ACCEPT
# Luego desde la DMZ a la LAN solo se acepta 25,110,143
iptables -A FORWARD -s 192.168.3.0/24 -p tcp --sport 25 \
-d 192.168.10.0/24 -j ACCEPT
iptables -A FORWARD -s 192.168.3.0/24 -p tcp --sport 143 \
-d 192.168.10.0/24 -j ACCEPT
iptables -A FORWARD -s 192.168.3.0/24 -p tcp --sport 143 \
-d 192.168.10.0/24 -j ACCEPT
# Cerramos el acceso de la DMZ a la LAN
```




```
iptables -A FORWARD -s 192.168.3.0/24 -d 192.168.10.0/24 -j DROP
## Cerramos el acceso de la DMZ al propio firewall
iptables -A INPUT -s 192.168.3.0/24 -i eth2 -j DROP
## Y ahora cerramos los accesos indeseados del exterior:
# Nota: 0.0.0.0/0 significa: cualquier red
# Cerramos el rango de puerto bien conocido
iptables -A INPUT -s 0.0.0.0/0 -p tcp -dport 1:1024 -j DROP
iptables -A INPUT -s 0.0.0.0/0 -p udp -dport 1:1024 -j DROP
# Cerramos un puerto de gestión: webmin
iptables -A INPUT -s 0.0.0.0/0 -p tcp -dport 10000 -j DROP
echo " OK . Verifique que lo que se aplica con: iptables -L -n"
# Fin del script
```

Se han remarcado las reglas FORWARD entre IPs privadas de delegaciones, ya que sin esas reglas y con el enmascaramiento de por medio no se podría acceder a las delegaciones. Cabe resaltar que entre delegaciones no hay visibilidad total, solamente la central vería a todas las demás, y las delegaciones solamente la central.

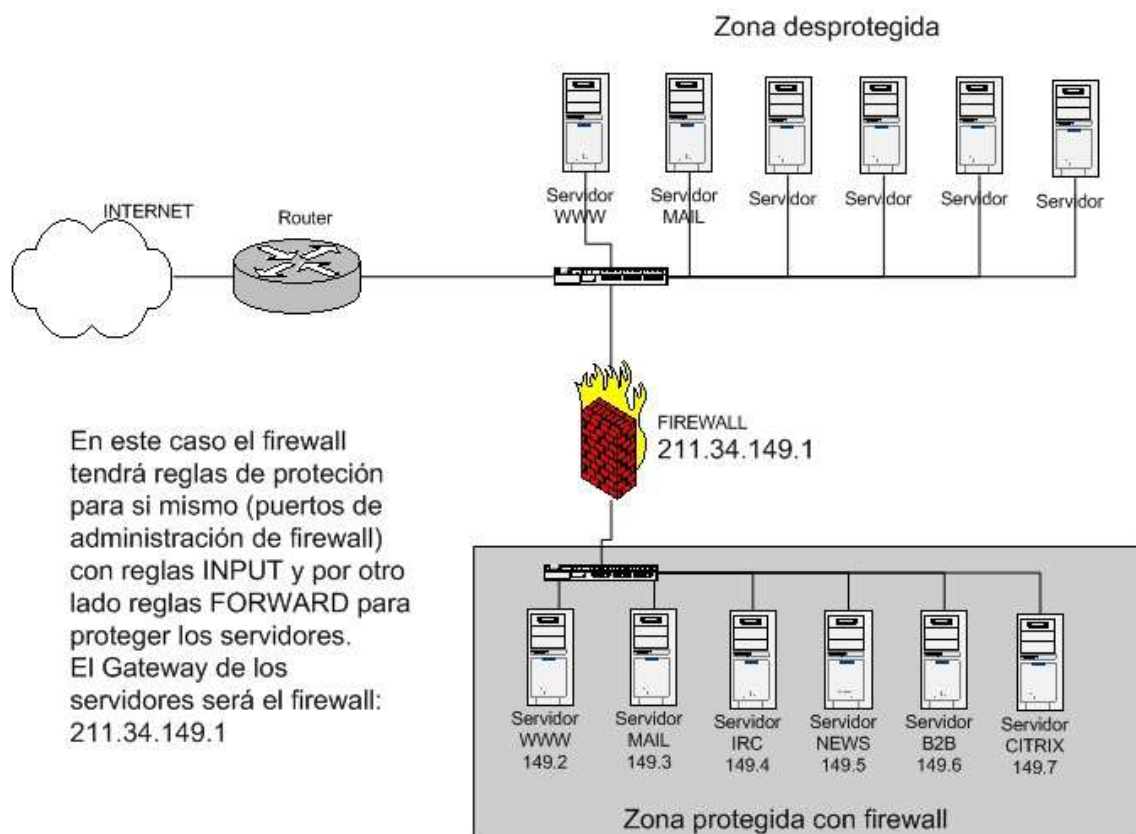
La delegaciones accederían al servidor de correo con una redirección, o sea que ellos se configurarían el servidor de correo como 192.168.10.1, mientras que desde la LAN se accedería directamente. Se puede hacer de distintas maneras.

Lo interesante sería poner ese firewall con DROP por defecto, se tratará de mostrar esa configuración al final.

Firewall puro y duro entre redes

En este caso olvidémonos de redes locales y de NAT. Aquí solo tendremos reglas de filtrado INPUT y FORWARD. Pongamos que tenemos el siguiente escenario:





esquema de firewall entre redes, en la que solo se filtra y no se hace NAT.

En el firewall debemos indicar una serie de reglas para proteger los equipos que están al otro lado de este dispositivo, todos ellos de la red 211.34.149.0/24

Cada uno de ellos da un servicio determinado, y puede estar gestionado desde distintas IPs, lo que significa que habrá que dar acceso a determinados puertos de gestión (22, 3389, etc..).

Este podría ser el aspecto del script del firewall:

```
#!/bin/sh
## SCRIPT de IPTABLES - ejemplo del manual de iptables
## Ejemplo de script para firewall entre redes.
## indetec Jose Maria Molina
## www.indetec.tk - jmmolinafuentes@wanadoo.es
echo -n Aplicando Reglas de Firewall...
## FLUSH de reglas
iptables -F
iptables -X
```

```
iptables -Z
iptables -t nat -F
## Establecemos politica por defecto
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
## Empezamos a filtrar
## Nota: eth0 es el interfaz conectado al router y eth1 a la LAN
# A nuestro firewall tenemos acceso total desde la nuestra IP
iptables -A INPUT -s 210.195.55.15 -j ACCEPT
# Para el resto no hay acceso al firewall
iptables -A INPUT -s 0.0.0.0/0 -j DROP
## Ahora podemos ir metiendo las reglas para cada servidor
## Como serán paquetes con destino a otras máquinas se aplica FORWARD
## Servidor WEB 211.34.149.2
# Acceso a puerto 80
iptables -A FORWARD -d 211.34.149.2 -p tcp --dport 80 -j ACCEPT
# Acceso a nuestra ip para gestionarlo
iptables -A FORWARD -s 210.195.55.15 -d 211.34.149.2 -p tcp --dport 22 -j ACCEPT
# El resto, cerrar
iptables -A FORWARD -d 211.34.149.2 -j DROP
## Servidor MAIL 211.34.149.3
# Acceso a puerto 25, 110 y 143
iptables -A FORWARD -d 211.34.149.3 -p tcp --dport 25 -j ACCEPT
iptables -A FORWARD -d 211.34.149.3 -p tcp --dport 110 -j ACCEPT
iptables -A FORWARD -d 211.34.149.3 -p tcp --dport 143 -j ACCEPT
# Acceso a gestion SNMP
iptables -A FORWARD -s 210.195.55.15 -d 211.34.149.3 -p udp --dport 169 -j ACCEPT
# Acceso a nuestra ip para gestionarlo
iptables -A FORWARD -s 210.195.55.15 -d 211.34.149.3 -p tcp --dport 22 -j ACCEPT
# El resto, cerrar
iptables -A FORWARD -d 211.34.149.3 -j DROP
## Servidor IRC 211.34.149.4
# Acceso a puertos IRC
iptables -A FORWARD -d 211.34.149.4 -p tcp --dport 6666:6668 -j ACCEPT
# Acceso a nuestra ip para gestionarlo
iptables -A FORWARD -s 210.195.55.15 -d 211.34.149.4 -p tcp --dport 22 -j ACCEPT
# El resto, cerrar
```



```
iptables -A FORWARD -d 211.34.149.4 -j DROP
## Servidor NEWS 211.34.149.5
# Acceso a puerto news
iptables -A FORWARD -d 211.34.149.5 -p tcp --dport news -j ACCEPT
# Acceso a nuestra ip para gestionarlo
iptables -A FORWARD -s 213.194.68.115 -d 211.34.149.5 -p tcp --dport 22 -j ACCEPT
# El resto, cerrar
iptables -A FORWARD -d 211.34.149.5 -j DROP
## Servidor B2B 211.34.149.6
# Acceso a puerto 443
iptables -A FORWARD -d 211.34.149.6 -p tcp --dport 443 -j ACCEPT
# Acceso a una ip para gestionarlo
iptables -A FORWARD -s 81.34.129.56 -d 211.34.149.6 -p tcp --dport 3389 -j ACCEPT
# El resto, cerrar
iptables -A FORWARD -d 211.34.149.6 -j DROP
## Servidor CITRIX 211.34.149.7
# Acceso a puerto 1494
iptables -A FORWARD -d 211.34.149.7 -p tcp --dport 1494 -j ACCEPT
# Acceso a una ip para gestionarlo
iptables -A FORWARD -s 195.55.234.2 -d 211.34.149.7 -p tcp --dport 3389 -j ACCEPT
# acceso a otro puerto quiza de BBDD
iptables -A FORWARD -s 195.55.234.2 -d 211.34.149.7 -p tcp --dport 1434 -j ACCEPT
# acceso a otro puerto quiza de BBDD
iptables -A FORWARD -s 195.55.234.2 -d 211.34.149.7 -p udp --dport 1433 -j ACCEPT
# El resto, cerrar
iptables -A FORWARD -d 211.34.149.7 -j DROP
echo " OK . Verifique que lo que se aplica con: iptables -L -n"
# Fin del script
```

Con esta firewall y sobretodo gracias a las reglas de DROP que metemos tras especificar lo que dejamos abiertos, protegeremos de manera eficaz todos los puertos abiertos de las máquinas.

Firewall con política por defecto DROP

Aquí llega la sección para los auténticos administradores de pelo en pecho.

- ¿Qué supone el hecho de establecer como política por defecto la denegación?
- Se debe explicitar cada conexión permitida en los dos sentidos.



- Se debe conocer perfectamente qué debe estar abierto y qué no.
- Es muchos más difícil de mantener y si se hace conviene hacerlo desde el principio.
- No todo es más trabajo: también supone un firewall mucho más seguro.

En el ejemplo de la DMZ ya se presentaba esta situación en las reglas forward de una a otra red. Para ilustrar el DROP por defecto, vamos a mostrar la configuración del ejemplo anterior de firewall entre redes pero con política por defecto DROP.

```
#!/bin/sh
## SCRIPT de IPTABLES - ejemplo del manual de iptables
## Ejemplo de script para firewall entre redes con DROP por defecto
## indetec Jose Maria Molina fuentes
## www.indetec.tk – jmmolinafuentes@wanadoo.es
echo -n Aplicando Reglas de Firewall...
## FLUSH de reglas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F
## Establecemos politica por defecto: DROP!!!
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
## Empezamos a filtrar
## Nota: eth0 es el interfaz conectado al router y eth1 a la LAN
# A nuestro firewall tenemos acceso total desde la nuestra IP
iptables -A INPUT -s 210.195.55.15 -j ACCEPT
iptables -A OUTPUT -d 210.195.55.15 -j ACCEPT
# Para el resto no hay acceso al firewall
# En principio esta de más, pero si rebajamos los permisos temporalmente
# nos cubre las espaldas
iptables -A INPUT -s 0.0.0.0/0 -j DROP
## Ahora podemos ir metiendo las reglas para cada servidor
## Como serán paquetes con destino a otras máquinas se aplica FORWARD
## Servidor WEB 211.34.149.2
# Acceso a puerto 80
iptables -A FORWARD -d 211.34.149.2 -p tcp --dport 80 -j ACCEPT
```



```
iptables -A FORWARD -s 211.34.149.2 -p tcp --sport 80 -j ACCEPT
# Acceso a nuestra ip para gestionarlo
iptables -A FORWARD -s 210.195.55.15 -d 211.34.149.2 -p tcp --dport 22 -j ACCEPT
iptables -A FORWARD -s 211.34.149.2 -d 210.195.55.15 -p tcp --sport 22 -j ACCEPT
## Servidor MAIL 211.34.149.3
# Acceso a puerto 25, 110 y 143
iptables -A FORWARD -d 211.34.149.3 -p tcp --dport 25 -j ACCEPT
iptables -A FORWARD -s 211.34.149.3 -p tcp --sport 25 -j ACCEPT
iptables -A FORWARD -d 211.34.149.3 -p tcp --dport 110 -j ACCEPT
iptables -A FORWARD -s 211.34.149.3 -p tcp --sport 110 -j ACCEPT
iptables -A FORWARD -d 211.34.149.3 -p tcp --dport 143 -j ACCEPT
iptables -A FORWARD -s 211.34.149.3 -p tcp --sport 143 -j ACCEPT
# Acceso a gestion SNMP
iptables -A FORWARD -s 210.195.55.15 -d 211.34.149.3 -p udp --dport 169 -j ACCEPT
iptables -A FORWARD -s 211.34.149.3 -d 210.195.55.15 -p udp --sport 169 -j ACCEPT
# Acceso a nuestra ip para gestionarlo
iptables -A FORWARD -s 210.195.55.15 -d 211.34.149.3 -p tcp --dport 22 -j ACCEPT
iptables -A FORWARD -s 211.34.149.3 -d 210.195.55.15 -p tcp --sport 22 -j ACCEPT
## Servidor IRC 211.34.149.4
# Acceso a puertos IRC
iptables -A FORWARD -d 211.34.149.4 -p tcp --dport 6666:6668 -j ACCEPT
iptables -A FORWARD -s 211.34.149.4 -p tcp --sport 6666:6668 -j ACCEPT
# Acceso a nuestra ip para gestionarlo
iptables -A FORWARD -s 210.195.55.15 -d 211.34.149.4 -p tcp --dport 22 -j ACCEPT
iptables -A FORWARD -s 211.34.149.4 -d 210.195.55.15 -p tcp --sport 22 -j ACCEPT
## Servidor NEWS 211.34.149.5
# Acceso a puerto news
iptables -A FORWARD -d 211.34.149.5 -p tcp --dport news -j ACCEPT
iptables -A FORWARD -s 211.34.149.5 -p tcp --sport news -j ACCEPT
# Acceso a nuestra ip para gestionarlo
iptables -A FORWARD -s 213.194.68.115 -d 211.34.149.5 -p tcp --dport 22 -j ACCEPT
iptables -A FORWARD -s 211.34.149.5 -d 213.194.68.115 -p tcp --sport 22 -j ACCEPT
# El resto, cerrar
iptables -A FORWARD -d 211.34.149.5 -j DROP
## Servidor B2B 211.34.149.6
# Acceso a puerto 443
iptables -A FORWARD -d 211.34.149.6 -p tcp --dport 443 -j ACCEPT
iptables -A FORWARD -s 211.34.149.6 -p tcp --sport 443 -j ACCEPT
```



```
# Acceso a una ip para gestionarlo
iptables -A FORWARD -s 81.34.129.56 -d 211.34.149.6 -p tcp --dport 3389 -j ACCEPT
iptables -A FORWARD -s 211.34.149.6 -d 81.34.129.56 -p tcp --sport 3389 -j ACCEPT
## Servidor CITRIX 211.34.149.7
# Acceso a puerto 1494
iptables -A FORWARD -d 211.34.149.7 -p tcp --dport 1494 -j ACCEPT
iptables -A FORWARD -s 211.34.149.7 -p tcp --sport 1494 -j ACCEPT
# Acceso a una ip para gestionarlo
iptables -A FORWARD -s 195.55.234.2 -d 211.34.149.7 -p tcp --dport 3389 -j ACCEPT
iptables -A FORWARD -s 211.34.149.7 -d 195.55.234.2 -p tcp --sport 3389 -j ACCEPT
# acceso a otro puerto quiza de BBDD
iptables -A FORWARD -s 195.55.234.2 -d 211.34.149.7 -p tcp --dport 1434 -j ACCEPT
iptables -A FORWARD -s 211.34.149.7 -d 195.55.234.2 -p tcp --sport 1434 -j ACCEPT
# acceso a otro puerto quiza de BBDD
iptables -A FORWARD -s 195.55.234.2 -d 211.34.149.7 -p udp --dport 1433 -j ACCEPT
iptables -A FORWARD -s 211.34.149.7 -d 195.55.234.2 -p udp --sport 1433 -j ACCEPT
echo " OK . Verifique que lo que se aplica con: iptables -L -n"
# Fin del script
```

Ya esta, hemos levantado un verdadero muro entre internet y el conjunto de servidores que esta

Tras el firewall. No se puede ni hacer un ping a las máquinas, salvo que se haya dado acceso total a una ip. Si quisieramos dar acceso al ping, pondríamos algo así:

Es más llevadero aplicar el DROP por defecto cuando el firewall es para la propia máquina. El primer escenario de esta manual trataba sobre este caso, ahora lo revisamos con la política por defecto drop.

```
#!/bin/sh
## SCRIPT de IPTABLES - ejemplo del manual de iptables
## Ejemplo de script para proteger la propia máquina
## con política por defecto DROP
## indetec Jose Maria Molina
## www.indetec.tk - jmmolinafuentes@wanadoo.es
echo -n Aplicando Reglas de Firewall...
## FLUSH de reglas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F
```



```
## Establecemos politica por defecto
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
## Empezamos a filtrar
# El localhost se deja (por ejemplo conexiones locales a mysql)
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT
# A nuestra IP le dejamos todo
iptables -A INPUT -s 195.65.34.234 -j ACCEPT
iptables -A OUTPUT -d 195.65.34.234 -j ACCEPT
# A un colega le dejamos entrar al mysql para que mantenga la BBDD
iptables -A INPUT -s 231.45.134.23 -p tcp --dport 3306 -j ACCEPT
iptables -A OUTPUT -d 231.45.134.23 -p tcp --sport 3306 -j ACCEPT
# A un diseñador le dejamos usar el FTP
iptables -A INPUT -s 80.37.45.194 -p tcp --dport 20:21 -j ACCEPT
iptables -A OUTPUT -d 80.37.45.194 -p tcp --sport 20:21 -j ACCEPT
# El puerto 80 de www debe estar abierto, es un servidor web.
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 80 -j ACCEPT
# Aquí están las reglas de cerrar. Como hemos comentado en la configuración
# anterior conviene tener esto escrito por si en algún momento se relaja el
# firewall y s cambia a de DROP a ACCEPT por defecto
# Cerramos rango de los puertos privilegiados. Cuidado con este tipo de
# barreras, antes hay que abrir a los que si tienen acceso.
iptables -A INPUT -p tcp --dport 1:1024
iptables -A INPUT -p udp --dport 1:1024
# Cerramos otros puertos que estan abiertos
iptables -A INPUT -p tcp --dport 3306 -j DROP
iptables -A INPUT -p tcp --dport 10000 -j DROP
iptables -A INPUT -p udp --dport 10000 -j DROP
echo " OK . Verifique que lo que se aplica con: iptables -L -n"
# Fin del script
```

Cómo depurar el funcionamiento del firewall



Programas Útiles

IPTRAF. Sin duda alguna uno de los programas más prácticos para depurar el firewall es iptables, ya que con el podemos observar si la conexiones se establecen o no; es un programa de consola que es aconsejable controlar ya que muestra en tiempo real el tráfico que atraviesa nuestra máquina con todo lujo de detalles: origen/destino de ips y puertos, tráfico total o tráfico total según el interfaz de red, etc? Si vemos muchas conexiones simultaneas y nos perdemos, existe la posibilidad de aplicar filtros para captar solo aquello que nos interesa.

NMAP. La herramienta para escanear puertos por excelencia, rechace imitaciones. Es una herramienta de consola rápida, efectiva y con multitud de opciones. Podemos usarla desde máquinas ajenas a nuestra red para comprobar si realmente el firewall esta filtrando correctamente y en cierta manera para hacernos una idea de que "visión" pueden tener los hackers de nuestro sistema.

SHELL. En el propio script del firewall podemos añadir algunas opciones para descubrir fallos de sintaxis en las reglas. Claro, imaginemos que tenemos un firewall de 40 lineas y una de ellas falla cuando ejecutamos el script. ¿Cuál es? Es probable que el mensaje de error no aclare lo suficiente, por eso se puede añadir algo así al final de cada regla:

```
...
iptables -A INPUT -s 195.55.234.2 -j ACCEPT && echo "regla-21 ok"
iptables -A INPUT -s 213.62.89.145 -j ACCEPT && echo "regla-22 ok"
...
```

Si la regla se ejecuta bien mostrará el mensajito de ok.

Otra opción algo mas cutre sería ir eliminando o comentando reglas hasta dar con la regla que tiene la sintaxis incorrecta. Cabe reseñar que puede fallar una regla, pero a partir de ella el resto se ejecutan con normalidad.

4.3 Introducción A NAT

La Traducción de Direcciones de Red, o NAT (*Network Address Translation*), es un sistema que se utiliza para asignar una red completa (o varias redes) a una sola dirección IP. NAT es necesario cuando la cantidad de direcciones

IP que nos haya asignado nuestro proveedor de Internet sea inferior a la cantidad de ordenadores que queramos que accedan a Internet.

NAT nos permite aprovechar los bloques de direcciones reservadas que se describen en el [RFC 1918](#). Generalmente, una red interna se suele configurar para que use uno o más de estos bloques de red. Estos bloques son:

10.0.0.0/8	(10.0.0.0 - 10.255.255.255)
172.16.0.0/12	(172.16.0.0 - 172.31.255.255)
192.168.0.0/16	(192.168.0.0 - 192.168.255.255)

Un sistema UNIX, Windows, Linux o MacOS configurado para NAT tendrá como mínimo dos adaptadoras de red, una para Internet y la otra para la red interna. NAT se encargará de traducir los requerimientos desde la red interna, de modo que parezca que todos provienen del sistema de origen en el que se encuentra configurado NAT.

Cómo Funciona NAT

Cuando un cliente en la red interna contacta con un máquina en Internet, envía paquetes IP destinados a esa máquina. Estos paquetes contienen toda la información de direccionamiento necesaria para que puedan ser llevados a su destino. NAT se encarga de estas piezas de información:

- Dirección IP de origen (por ejemplo, 192.168.1.35)
- Puerto TCP o UDP de origen (por ejemplo, 2132)

Cuando los paquetes pasan a través de la pasarela de NAT, son modificados para que parezca que se han originado y provienen de la misma pasarela de NAT. La pasarela de NAT registra los cambios que realiza en su tabla de estado, para así poder: a) invertir los cambios en los paquetes devueltos, y b) asegurarse de que los paquetes devueltos pasen a través del cortafuegos y no sean bloqueados. Por ejemplo, podrían ocurrir los siguientes cambios:

- IP de origen: sustituida con la dirección externa de la pasarela (por ejemplo, 24.5.0.5)
- Puerto de origen: sustituido con un puerto no en uso de la pasarela, escogido aleatoriamente (por ejemplo, 53136)

Ni la máquina interna ni el anfitrión de Internet se dan cuenta de estos pasos de traducción. Para la máquina interna, el sistema NAT es simplemente una pasarela a Internet. Para el anfitrión de Internet, los paquetes parecen venir directamente del sistema NAT; ni siquiera se da cuenta de que existe la estación interna.

Cuando el anfitrión de Internet responde a los paquetes internos de la máquina, los direcciona a la IP externa de la pasarela de NAT (24.5.0.5) y a su puerto de traducción (53136). La pasarela de NAT busca entonces en la tabla de estado para determinar si los paquetes de respuesta concuerdan con alguna conexión establecida. Entonces encontrará una única concordancia basada en la combinación de la dirección IP y el puerto, y esto indica a PF que los paquetes pertenecen a una conexión iniciada por la máquina interna 192.168.1.35. Acto seguido PF realiza los cambios

opuestos a los que realizó para los paquetes salientes, y reenvía los paquetes de respuesta a la máquina interna.

La traducción de paquetes ICMP ocurre de forma parecida, pero sin la modificación del puerto de origen.

Activación de NAT

Para activar NAT en una pasarela de Unix, Linux o MacOS, además de [activar PF](#), también hay que activar el reenvío de paquetes IP (*IP forwarding*):

```
# sysctl -w net.inet.ip.forwarding=1
# sysctl -w net.inet6.ip6.forwarding=1 (si se usa IPv6)
```

Para que este cambio sea permanente, hay que añadir las siguientes líneas al fichero [/etc/sysctl.conf](#):

```
net.inet.ip.forwarding=1
net.inet6.ip6.forwarding=1
```

Estas líneas ya existen en la instalación determinada, pero como comentarios (prefijadas con #). Hay que quitar el signo # y guardar el fichero. El reenvío de IP se activará cuando se reinicie la máquina.

Yo dividiría NAT en dos diferentes tipos: **Source NAT** (SNAT, por origen), y **Destination NAT** (DNAT, por destino).

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 (reenvío de puerto), el balanceo de carga y el proxy transparente son formas de DNAT.

