

Kernel IPTables

Fecha:04/10/2006

Fuente: www.lordepsylon.net

El objetivo principal del proyecto es mostrar un ejemplo de configuración del Kernel-IPTables.

<Firewall: también conocido como cortafuegos, es un equipo de hardware o software utilizado en las redes de ordenadores para prevenir algunos tipos de comunicaciones prohibidos por la política de red.>

La principal y más segura herramienta de software utilizada como cortafuegos, implementada en Gnu/Linux de forma nativa, es el Iptables del Kernel. La descripción que da wikipedia es la siguiente:

"Iptables es el nombre de la herramienta del espacio de usuario por medio de la cual el administrador crea reglas para filtrado de paquetes y para hacer NAT. Mientras que técnicamente iptables es solamente la herramienta que controla estos componentes dentro del kernel, el nombre -iptables- se utiliza muchas veces para referirse a toda la infraestructura, incluyendo a netfilter, connection tracking y NAT, como también a la herramienta propiamente dicha."

Por conocer un poco de su historia, también os copio lo siguiente:

"El proyecto -netfilter/iptables- fue comenzado en 1998 por Rusty Russell, también autor del proyecto que precedió: ipchains. A medida que el proyecto creció, fundó el -Netfilter Core Team- en 1999. El software que ellos produjeron está licenciado bajo GPL (GNU General Public License), y fue incorporado al kernel Linux 2.3 en Marzo del 2000. En agosto del 2003 Harald Welte fue designado chairman del Coreteam. En abril del 2004, se produjo una ruptura del proyecto por aquellos que lo usaban embebido en routers sin ajustarse a GPL.

Antes de iptables los programas más usados para crear firewalls en Linux eran -ipchains- en Linux 2.2 e -ipfwadm- en Linux 2.0 que se basaba en el -ipfw- de BSD. Tanto ipchains como ipfwadm alteran el código de red para poder manipular los paquetes, ya que no existía un framework general para el manejo de paquetes, hasta la aparición de netfilter. Iptables mantiene la idea básica introducida en Linux con ipfwadm: lista de reglas con -qué matchear dentro de un paquete- y -qué hacer con el paquete-. Ipchains había agregado el concepto de -cadenas de reglas- (o chains) e iptables extendió esto a la idea de tablas: una tabla era consultada cuando se decidía NAT-ear un paquete, y otra cuando se debía decidir si filtrar el paquete. Adicionalmente, los tres puntos en los que se realiza el filtrado en el viaje de un paquete fue modificado, de modo que un paquete sólo pasa por un punto de filtrado.

Mientras que ipchains e ipfwadm combinan filtrado de paquetes y NAT (específicamente tres tipos de NAT, llamado masquerading, port forwarding y redirection), netfilter hace posible separar las operaciones sobre los paquetes en tres partes: packet filtering, connection tracking, y Network Address Translation. Cada parte se conecta a las reglas de netfilter en diferentes puntos de acceso de los paquetes. Los subsistemas de connection tracking y NAT son más generales y poderosos que los que realizaban ipchains e ipfwadm.

Esta división permite a iptables, en su momento, usar la información que la capa de connection tracking ha determinado acerca del paquete: esta información estaba antes asociada a NAT. Esto hace a iptables superior a ipchains ya que tiene la habilidad de monitorear el estado de una conexión y redirigir, modificar o detener los paquetes de datos basados en el estado de la conexión y no solamente por la fuente, destino o contenido del paquete. Un firewall que utilice iptables de este modo se llama -firewall statefull-contrario a ipchains que sólo puede realizar -firewalls stateless- (excepto en casos limitados). Es posible decir que ipchains no está al tanto del contexto completo del cual un paquete surge, mientras que iptables sí.

Por tanto iptables puede hacer mejores decisiones sobre el futuro de los paquetes y las conexiones.

Iptables, el subsistema NAT y el subsistema de connection tracking son también extensibles, y muchas extensiones están incluidas en el paquete básico de iptables, tal como la extensión ya mencionada que permite la consulta del estado de la conexión.

Extensiones adicionales se distribuyen junto a la utilidad iptables, como parches al código fuente del kernel junto con una herramienta llamada patch-o-matic.

Una versión de iptables para IPv6 ya fue escrita, llamada ip6tables al igual que la herramienta"

Habiendo entendido su procedencia y su funcionamiento, os voy a copiar un script que he creado, que configura nuestro Kernel-IPTables firewall de forma rápida-sencilla, y que puede servir para el uso general. Podéis crearos vosotros mismos el script, o utilizar los links que se encuentran al final para su descarga.

----- Cortar aquí -----

```
#!/bin/bash
#####
# Kernel-IPTables-2006-Lord Epsilon #
# iFrame06.net #
#--#-#####--#-#####|#?
#1-Sintaxis basica#
###-?##-##-#####

#-i Especificamos un interface de entrada
#-o Especificamos un interface de salida

#-p Especificamos el protocolo

#-s Especificamos la direccion de origen
#-d Especificamos la direcci?n de destino

#-j Especificamos el argumento a ejecutar en el paquete

#--sport Puerto de origen
#--dport Puerto de destino
```

Te recuerdo que para activar la regla, solamente tienes que quitar (descomentar) la #
Si quieres desactivar una regla, solamente tienes que poner (comentar) una # al principio de la línea

```
###-?###-###-#####|#?
```

```
#2-Politica general#
```

```
###-?###-###-#####
```

```
#Borramos todas las reglas existentes  
iptables -F
```

```
#Cerramos todos los puertos y solo dejamos entrar y salir lo solicitado
```

```
iptables -P INPUT DROP
```

```
iptables -P OUTPUT ACCEPT
```

```
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
```

```
iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
```

```
###-?###-###-#####|#?
```

```
#3-Seguridad puertos #
```

```
###-?###-###-#####
```

```
#Permitimos la comunicacion con el servidor dns
```

```
iptables -A INPUT -p UDP --dport 53 -j ACCEPT
```

```
iptables -A INPUT -p TCP --dport 53 -j ACCEPT
```

```
#Esta regla permite el trafico LAN en un determinado equipo de la red (cambia las x por tu ip)
```

```
#iptables -A INPUT -s 192.168.xxx.xxx -j ACCEPT
```

```
#Yo lo tengo asi para mi LAN
```

```
iptables -A INPUT -s 192.168.0.214 -j ACCEPT
```

```
#Permitimos el uso de SSH (puerto 22)
```

```
iptables -A INPUT -p TCP --dport 22 -j ACCEPT
```

```
#Esta regla permite el SSH en un determinado equipo de la red (cambia las x por la ip)
```

```
#iptables -A INPUT -s 192.168.xxx.xxx -p TCP --dport 22 -j ACCEPT
```

```
#Permitimos el acceso a nuestro sitio WEB (puerto 80)
```

```
#iptables -A INPUT -m state --state NEW -p TCP --dport 80 -j ACCEPT
```

```
#Permitimos uso de FTP (puerto 21)
```

```
#iptables -A INPUT -p TCP --dport 21 -j ACCEPT
```

```
#Permitimos acceso POP3 (puerto 110)
```

```
#iptables -A INPUT -p TCP --dport 110 -j ACCEPT
```

```
# Permitimos uso de SMTP (puerto 25)
```

```
#iptables -A INPUT -p TCP --dport 25 -j ACCEPT
```

```

#Permitimos acceso IMAP (puerto 143)
#iptables -A INPUT -p TCP --dport 143 -j ACCEPT
#iptables -A INPUT -p UDP --dport 143 -j ACCEPT

#Permitimos el acceso de localhost (Para trabajar en modo local)
iptables -A INPUT -i lo -j ACCEPT

#---#--#####---#---#####|#?
#4-Seguridad Avanzada#
###-?##----###-#####

#No respondemos a las peticiones broadcast
/bin/echo "1" > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

#Evitamos los pings
/bin/echo "1" > /proc/sys/net/ipv4/icmp_echo_ignore_all

#Nos aseguramos que no hay forward
/bin/echo "0" > /proc/sys/net/ipv4/ip_forward

#Para evitar posibles tecnicas de spoofing comprobamos que la direccion
#origen del paquete recibido viene del sitio correcto.
for interface in /proc/sys/net/ipv4/conf/*/rp_filter; do
/bin/echo "1" > ${interface}
done

#Comprobamos los paquetes ICMPs redirigidos que pueden alterar la tabla de rutas que
usamos
for interface in /proc/sys/net/ipv4/conf/*/accept_redirects; do
/bin/echo "0" > ${interface}
done

#No guardamos registros de los marcianos
/bin/echo "1" > /proc/sys/net/ipv4/conf/all/log_martians

#---#--#####---#---#####|#?
#5-Puesta en marcha #
###-?##----###-#####

#Abrimos un terminal shell

#Damos permisos de ejecucion a este script
#chmod +x kernel-iptables.sh

#Ejecutamos SH
#sh kernel-iptables.sh

#Comprobamos las reglas establecidas
#iptables -L

```

```
#Las establecemos para que se arranquen siempre
#cp kernel-iptables.sh /etc/init.d/
#update-rc.d kernel-iptables.sh defaults
```

```
#---#---#####---#---#####\#?
# Libertad & Copyleft #
###-?##----###-#####
```

----- Cortar aquí -----

En los siguientes links podeis descargaros las instrucciones de instalación y el script que configura nuestro firewall

-Instrucciones de instalación y uso (visitar fuente)

-Script Kernel-IPtables.sh (visitar fuente)

"En la gestión de tus flujos de información, reside la verdadera seguridad"

<Lord Epsilon>