Practica IPTABLES

Lo siguientes comandos son ejecutados en una terminal, necesitan ser ejecutados con privilegios de administrador, yo antepondré sudo a cada comando, ustedes pueden hacerlo igual o evitar usar sudo ejecutando los comandos directamente como root

En el anterior post había explicado que es necesario en un firewall primero denegar todo tráfico entrante, para ello:

```
sudo iptables -P INPUT DROP
```

Luego debemos permitir que nuestro propio ordenador tenga permiso para entrar datos:

```
sudo iptables -A INPUT -i lo -j ACCEPT
```

Así como también aceptar paquetes de peticiones que se originen en nuestro ordenador:

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

Hasta aquí nuestro ordenador puede navegar sin problemas a internet, pero nadie de ningún otro entorno (LAN, internet, Wifi, etc) podrá acceder de ninguna forma a nuestro ordenador. Vamos a empezar ya a configurar iptables según nuestras necesidades.

Usando ulogd para sacar los logs de iptables a otro archivo:

Por defecto los logs de iptables van en el log del kernel, del sistema, o algo así … en Arch por defecto, ahora mismo ni siquiera recuerdo dónde van, es por ello que yo uso ulogd para que los logs de iptables estén en otro archivo.

```
sudo apt-get install ulogd
```

```
sudo iptables -A INPUT -p tcp -m tcp --tcp-flags FIN,SYN,RST,ACK SYN -j ULOG
```

Para ver los registros:

```
/var/log/ulog/syslogemu.log
```

En este archivo que les menciono es donde por defecto ulogd ubica los logs de paquetes rechazados, no obstante si desean que sea en otro archivo y no en este pueden modificar la línea #53 en /etc/ulogd.conf, simplemente cambian la ruta del archivo que muestra esa línea y luego reinician el daemon:

```
sudo /etc/init.d/ulogd restart
```

Si miran atentamente ese archivo podrán ver que hay opciones para inclusive, guardar los logs en una base de datos MySQL, SQLite o PostgreSQL, de hecho los archivos de configuración de ejemplo se encuentran en /usr/share/doc/ulogd/

Ok, ya tenemos los logs de iptables en otro archivo, ahora ¿cómo mostrarlos?

Para ello un simple cat bastaría:

```
cat /var/log/ulog/syslogemu.log
```

Dando acceso a mi servidor privado:

Yo no uso VirtualBox ni nada similar para virtualizar, yo tengo mi servidor privado virtualizado con Qemu+KVM el cual debe poder conectarse a mi laptop como tal, con las reglas de iptables que acabo de especificar arriba no podrá, es por eso que tengo obligatoriamente que darle permiso a la IP de mi servidor virtual para que pueda acceder a mi laptop:

```
sudo iptables -A INPUT -i virbr0 -p tcp -s 192.168.122.88 -j ACCEPT
```

Vamos a detallar esta línea, es importante que ustedes entiendan qué significa cada parámetro, pues se van a repetir mucho de ahora en adelante:

- **-A INPUT** : Estoy diciendo que voy a declarar una regla para tráfico entrante
- **-i virbr0** : Declaro que la interfaz por la cual aceptaré el tráfico no es etho (LAN) ni wlan0 (Wifi), digo específicamente que es mi interfaz virbr0, o sea, la interfaz de red virtual (interna) mediante la cual se comunica mi laptop con mi servidor virtual (y viceversa)
- **-p tcp** : Especifico el protocolo, los más usados son UDP y TCP, aquí en realidad bastaba con no poner esto pero… es una costumbre especificar el tipo de protocolo a aceptar
- **-s 192.168.122.88** : La source, fuente de los paquetes. O sea, la regla se refiere a paquetes que provengan específicamente desde la IP 192.168.122.88
- **-j ACCEPT** : Ya aquí digo lo que quiero hacer con los paquetes que coincidan con lo antes dicho, en este caso aceptar.

O sea y a modo de resumen, voy a aceptar paquetes que provengan desde la IP 192.168.122.88, pero en caso de que deseen entrar paquetes que vengan desde esa IP PERO! entran desde una interfaz que no sea virbr0, o sea, digamos que intentan entrar paquetes desde la IP 192.168.122.88 pero son de un ordenador en nuestra red Wifi, si ese es el caso, los paquetes serán rechazados. ¿por qué? Porque claramente especificamos que sí, aceptamos paquetes desde 192.168.122.88 sí, pero y solo pero, también tienen que entrar desde la interfaz virbr0 (interfaz de red interna, virtual), si los paquetes provienen desde otra interfaz (LAN, RAS, Wifi, etc) entonces no serán aceptados. Especificando la interfaz como pueden apreciar podemos restringir aún más, podemos tener un mejor control sobre lo que entra (o no entra) en nuestro ordenador.

Aceptando ping desde cualquier IP de la Wifi de casa:

Desde algún otro ordenador que se conecte a la Wifi, si intenta hacer ping a mi laptop deseo permitirlo. ¿motivo? La idea también es que en próximas semanas enlazar la PC de la casa de al lado a la red, así compartir información sería menos compleja, más fluida, cuando yo empiece a hacer pruebas para enlazar la desktop a la Wifi, necesitaré hacer ping a mi laptop para comprobar la conectividad, si mi laptop no me devuelve el ping puedo pensar que el AP está fallando, o que hubo algún error al acceder a la Wifi, es por ello que deseo permitir el ping.

```
sudo iptables -A INPUT -i wlo1 -p icmp -s 192.168.1.0/24 -d 192.168.1.51 -j ACCEPT
```

- **-A INPUT** : Lo mismo que antes, hago referencia a tráfico entrante
- **-i wlo1** : Similar a antes. En el caso anterior especifiqué la interfaz virtual, en este caso especifico otra interfaz, la de mi wifi: wlo1
- **-p icmp** : Protocolo icmp, icmp = ping. O sea, no estoy permitiendo SSH ni nada similar, permito solamente ping (icmp)
- **-s 192.168.1.0/24** : La fuente de los paquetes, o sea, siempre que los paquetes vengan desde una IP 192.168.1.? serán aceptados
- **-d 192.168.1.51** : IP de destino, o sea, mi IP.
- **-j ACCEPT** : Indico qué hacer con los paquetes que coincidan con lo anterior, aceptar.

O sea y para explicar esto de forma corrida, acepto que me hagan ping (protocolo icmp) cuyo destino sea específicamente mi IP, siempre y cuando vengan desde alguna IP como 192.168.1.__ pero además, no pueden provenir desde cualquier interfaz de red, tienen que entrar específicamente desde mi interfaz de red de Wifi (wlo1)

Aceptar SSH solo para una IP:

A veces necesito conectarme por SSH desde mi smartphone para controlar la laptop, es por ello que debo permitir el acceso SSH a mi laptop desde las IPs de mi Wifi, para ello:

```
sudo iptables -A INPUT -i wlo1 -p tcp -s 192.168.1.0/24 -d 192.168.1.51 --dport 22 -j ACCEPT
```

De esta línea lo único diferente o que merece destacarse es: –dport 22 (puerto de SSH que uso)

O sea, acepto intentos de conexión a mi laptop por el puerto 22, siempre que estos vengan desde alguna IP de mi wifi, tienen también que tener como destino específico mi IP y además venir por la interfaz wlo1, o sea, la de mi wifi (no la de la lan, etc)

Permitiendo que vean un sitio web suyo:

No es mi caso, pero si alguno de ustedes tiene un sitio web hosteado y no desean denegarle el acceso a nadie, o sea, que todos desde cualquier parte puedan acceder a ese sitio web, es mucho más simple de lo que pueden pensar:

```
sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
```

O sea, aquí están permitiendo todo tráfico entrante (tcp) por el puerto 80. Como ven, no especifico desde qué IPs o red sí permito el acceso, al no especificar un rango de IP a permitir, iptables asume que deseo permitirle el acceso a todos los rangos de IPs existentes, o sea, a todo el mundo :)Firewall Iptables - Introducción.

Acerca de Iptables y Netfilter.

Netfilter es un conjunto de *ganchos* (**Hooks**, es decir, técnicas de programación que se emplean para crear cadenas de procedimientos como manejador) dentro del núcleo de GNU/Linux y que son utilizados para interceptar y manipular paquetes de red. El componente mejor conocido es el cortafuegos, el cual realiza procesos de filtración de paquetes. Los *ganchos* son también utilizados por un componente que se encarga del **NAT** (acrónimo de **N**etwork **A**ddress **T**ranslation o Traducción de dirección de red). Estos componentes son cargados como módulos del núcleo.

Iptables es el nombre de la herramienta de espacio de usuario (**User Space**, es decir, área de memoria donde todas las aplicaciones, en modo de usuario, pueden ser intercambiadas hacia memoria virtual cuando sea necesario) a través de la cual los administradores crean reglas para cada filtrado de paquetes y módulos de **NAT**. **Iptables** es la herramienta estándar de todas las distribuciones modernas de GNU/Linux.

URL: http://www.netfilter.org/

Procedimientos.

Cadenas.

Las cadenas pueden ser para tráfico entrante (INPUT), tráfico saliente (OUTPUT) o tráfico reenviado (**FORWARD**).

Reglas de destino.

Las reglas de destino pueden ser aceptar conexiones (**ACCEPT**), descartar conexiones (**DROP**), rechazar conexiones (**REJECT**), encaminamiento posterior (**POSTROUTING**), encaminamiento previo (**PREROUTING**), **SNAT**, **NAT**, entre otras.

Políticas por defecto.

Establecen cual es la acción a tomar por defecto ante cualquier tipo de conexión. La opción -P cambia una política para una cadena. En el siguiente ejemplo se descartan (**DROP**) todas las conexiones que ingresen (INPUT), todas las conexiones que se reenvíen (**FORWARD**) y todas las conexiones que salgan (OUTPUT), es decir, se descarta todo el tráfico que entre desde una red pública y el que trate de salir desde la red local.

```
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT
```

Limpieza de reglas específicas.

A fin de poder crear nuevas reglas, se deben borrar las existentes, para el tráfico entrante, tráfico reenviado y tráfico saliente así como el NAT.

```
iptables -F INPUT
iptables -F FORWARD
iptables -F OUTPUT
iptables -F -t nat
```

Reglas específicas.

Las opciones más comunes son:

- -A añade una cadena, la opción -i define una interfaz de tráfico entrante
- -o define una interfaz para trafico saliente
- -j establece una regla de destino del tráfico, que puede ser **ACCEPT**, **DROP** o **REJECT**. La
- -m define que se aplica la regla si hay una coincidencia específica
- --state define una lista separada por comas de distinto tipos de estados de las conexiones (INVALID, ESTABLISHED, NEW, RELATED).
- --to-source define que IP reportar al tráfico externo
- -s define trafico de origen
- -d define tráfico de destino
- --source-port define el puerto desde el que se origina la conexión
- --destination-port define el puerto hacia el que se dirige la conexión
- -t tabla a utilizar, pueden ser nat, filter, mangle o raw.

Ejemplos de reglas.

Reenvío de paquetes desde una interfaz de red local (eth1) hacia una interfaz de red pública (eth0):

```
iptables -A FORWARD -i eth1 -o eth0 -j ACCEPT
```

Aceptar reenviar los paquetes que son parte de conexiones existentes (ESTABLISHED) o relacionadas de tráfico entrante desde la interfaz eth1 para tráfico saliente por la interfaz eth0:

```
iptables -A FORWARD -i eth0 -o eth1 -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Permitir paquetes en el propio muro cortafuegos para tráfico saliente a través de la interfaz eth0 que son parte de conexiones existentes o relacionadas:

```
iptables -A INPUT -i eth0 -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Permitir (**ACCEPT**) todo el tráfico entrante (INPUT) desde (**-s**) cualquier dirección (0/0) la red local (eth1) y desde el retorno del sistema (lo) hacia (**-d**) cualquier destino (0/0):

```
iptables -A INPUT -i eth1 -s 0/0 -d 0/0 -j ACCEPT
iptables -A INPUT -i lo -s 0/0 -d 0/0 -j ACCEPT
```

Hacer (-j) SNAT para el tráfico saliente (**-o**) a tráves de la interfaz eth0 proveniente desde (**-s**) la red local (**192.168.0.0/24**) utilizando (--to-source) la dirección IP **w.x.y.z**.

```
iptables -A POSTROUTING -t nat -s 192.168.0.0/24 -o eth0 -j SNAT --to-source x.y.z.c
```

Descartar (**DROP**) todo el tráfico entrante (**-i**) desde la interfaz eth0 que trate de utilizar la dirección IP pública del servidor (**w.x.y.z**), alguna dirección IP de la red local (**192.168.0.0/24**) o la dirección IP del retorno del sistema (127.0.01)

```
iptables -A INPUT -i eth0 -s w.x.y.x/32 -j DROP
iptables -A INPUT -i eth0 -s 192.168.0.0/24 -j DROP
iptables -A INPUT -i eth0 -s 127.0.0.0/8 -j DROP
```

Aceptar (**ACCEPT**) todos los paquetes SYN (--syn) del protocolo TCP (-p tcp) para los puertos (**--destination-port**) de los protocolos SMTP (25), HTTP(80), HTTPS (443) y SSH (22):

```
iptables -A INPUT -p tcp -s 0/0 -d 0/0 --destination-port 25 --syn -j ACCEPT
iptables -A INPUT -p tcp -s 0/0 -d 0/0 --destination-port 80 --syn -j ACCEPT
iptables -A INPUT -p tcp -s 0/0 -d 0/0 --destination-port 443 --syn -j ACCEPT
iptables -A INPUT -p tcp -s 0/0 -d 0/0 --destination-port 22 --syn -j ACCEPT
```

Aceptar (**ACCEPT**) todos los paquetes SYN (--syn) del protocolo TCP (-tcp) para los puertos (**--destination-port**) del protocolos SMTP (25) en el servidor (**w.x.y.z**/32), desde (**-s**) cualquier lugar (0/0) hacia (**-d**) cualquier lugar (0/0).

```
iptables -A INPUT -p tcp -s 0/0 -d w.x.y.z/32 --destination-port 25 --syn -j ACCEPT
```

Aceptar (**ACCEPT**) todos los paquetes SYN (--syn) del protocolo TCP (-p tcp) para los puertos (**--destination-port**) de los protocolos POP3 (110), POP3S (995), IMAP (143) y IMAPS (993):

```
iptables -A INPUT -p tcp -s 0/0 -d 0/0 --destination-port 110 --syn -j ACCEPT
iptables -A INPUT -p tcp -s 0/0 -d 0/0 --destination-port 995 --syn -j ACCEPT
iptables -A INPUT -p tcp -s 0/0 -d 0/0 --destination-port 143 --syn -j ACCEPT
iptables -A INPUT -p tcp -s 0/0 -d 0/0 --destination-port 993 --syn -j ACCEPT
```

Aceptar (**ACCEPT**) el tráfico entrante (**-i**) proveniente desde la interfaz eth1 cuando las conexiones se establezcan desde el puerto (--sport) 67 por protocolos (**-p**) TCP y UDP.

```
iptables -A INPUT -i eth1 -p tcp --sport 68 --dport 67 -j ACCEPT
iptables -A INPUT -i eth1 -p udp --sport 68 --dport 67 -j ACCEPT
```

Aceptar (**ACCEPT**) conexiones de tráfico entrante (INPUT) por protocolo (**-p**) UDP cuando se establezcan desde (**-s**) el servidor DNS 200.33.145.217 desde el puerto (**--source-port**) 53 hacia (**-d**) cualquier destino (0/0):

```
iptables -A INPUT -p udp -s 200.33.146.217/32 --source-port 53 -d 0/0 -j ACCEPT
```

Cerrar accesos.

Descartar (**DROP**) el tráfico entrante (INPUT) para el protocolo (**-p**) TCP hacia los puerto (**--destination-port**) de SSH (22) y Telnet (23):

```
iptables -A INPUT -p tcp --destination-port 22 -j DROP
iptables -A INPUT -p tcp --destination-port 23 -j DROP
```

Descartar (**DROP**) todo tipo de conexiones de tráfico entrante (INPUT) desde (**-s**) la dirección IP a.b.c.d:

```
iptables -A INPUT -s a.b.c.d -j DROP
```

Rechazar (**REJECT**) conexiones hacia (OUTPUT) la dirección IP a.b.c.d desde la red local:

```
iptables -A OUTPUT -d a.b.c.d -s 192.168.0.0/24 -j REJECT
```

Eliminar reglas.

En general se utiliza la misma regla, pero en lugar de utilizar -A (append), se utiliza -D (delete).

Elininar la regla que descarta (**DROP**) todo tipo de conexiones de tráfico entrante (INPUT) desde (**-s**) la dirección IP a.b.c.d:

```
iptables -D INPUT -s a.b.c.d -j DROP
```

Mostrar la lista de cadenas y reglas.

Una vez cargadas todas las cadenas y reglas de **iptables** es posible visualizar éstas utilizando el mandato **iptables** con las opciones **-n**, para ver las listas en formato numérico, y -**L**, para solicitar la lista de éstas cadenas.

```
iptables -nL
```

Cuando no hay reglas ni cadenas cargadas, la salida **debe** devolver lo siguiente:

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

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

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

Cuando hay cadenas presentes, la salida, suponiendo que se utilizarón los ejemplos de este documento, debe devolver algo similar a lo siguiente:

```
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
DROP       all  --  192.168.1.64         0.0.0.0/0
DROP       all  --  172.16.0.0/24        0.0.0.0/0
DROP       all  --  127.0.0.0/8          0.0.0.0/0
……
```