Laboratorio 11: Cortafuegos en Linux

Objetivos:

1. Aprender los conceptos básicos para crear, para un servidor en Linux, un bastión con `iptables`.
2. Aprender a realizar configuraciones básicas de cortafuegos en Linux con `iptables`.

Topología de trabajo

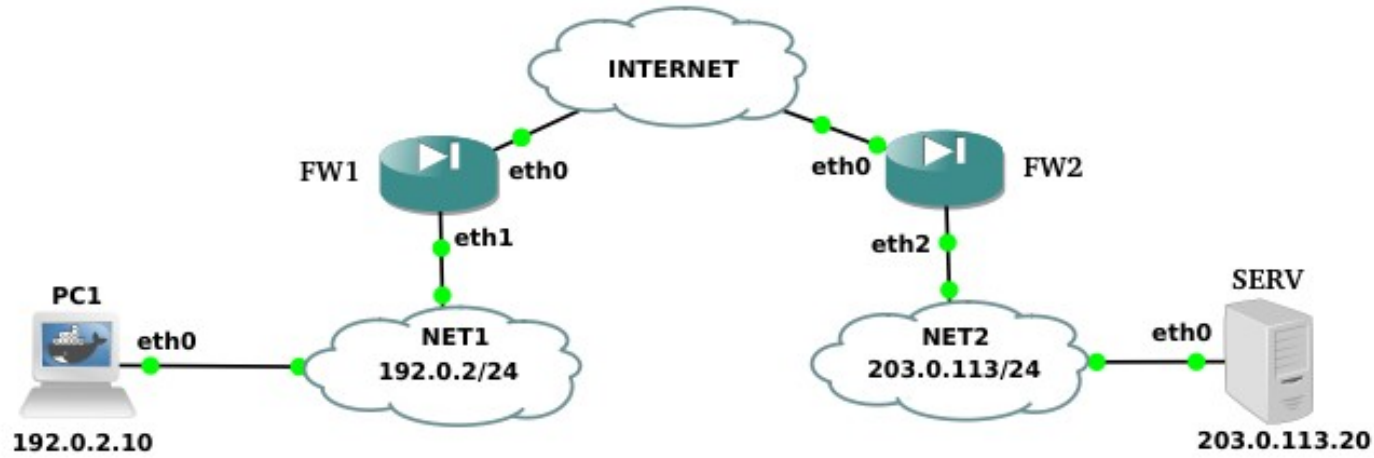


Figura 1 : Red utilizada

Tareas

Preparativos:

1. Para empezar, desde *e-gela*, descarga la configuración de la red, y descomprímela en el directorio `GNS3/projects`.

   Ejecuta en la máquina virtual: `docker images` Si no aparece una línea con el nombre `alpineplus`, entonces, descarga desde el repositorio/*e-gela* el fichero `alpineplus.tar.gz`, que es el contenedor para trabajar con `iptables`. Cópialo en la máquina virtual y ejecuta el siguiente comando `docker load < alpineplus.tar.gz`

   Para habilitar la recepción de registros desde `GNS3` ejecuta el siguiente comando en la máquina virtual:

   ```
   $ sudo sysctl -w net.netfilter.nf_log_all_netns=1
   ```

2. Lanza `GNS3`, abre la red descargada. Verifica el nombre de las interfaces, pon en marcha todas las máquinas y abre todas las consolas.
3. Para verificar que la red funciona correctamente, haz `ping` desde `PC1` a `SERV`, desde `FW1` a `SERV`, y desde `FW2` a `PC1`.
4. Verifica que en el servidor `SERV` están disponibles el servicio `telnet` y `web`. Prueba a realizar lo siguiente desde `PC1`:
   a) Establecer una sesión `telnet` con el servidor `SERV`. Para finalizar la conexión, `exit`.

b) Repetir la conexión `telnet`, pero ahora al puerto `80`, donde escucha el servidor web. Para comunicarte con el servidor escribe la siguiente solicitud `HTTP`:

```
GET / HTTP/1.1
```
[línea en blanco]

Si algo no funciona comentárselo al docente.

Configuración de un bastión:

5. Supongamos que en la máquina `SERV` se ha definido la siguiente política de seguridad:
   - Solo se permite el tráfico `ICMP`, y solo desde su propia red.
   - Descartar el resto de tráfico y guardar en un registro (*log*) el tráfico descartado con un comentario descriptivo.

   Utilizando `iptables`, escribe un *script* (esquema al final) en el servidor para implementar los criterios de seguridad anteriores.
6. Ejecuta el *script* y verifica que el contenido de la tabla de filtrado sea el esperado.
7. Comprueba que la configuración realizada cumple la política de seguridad definida, para eso:
   a) Haz `ping` de `FW2` a `SERV`. Si no funciona, revisa lo hecho hasta el momento.
   b) Haz `ping` de `PC1` a `SERV`. No debe funcionar, si lo hace repasa lo hecho. Analiza los registros (*logs*) para verificar que este tráfico ha sido interceptado. Ejecuta `traceroute` para comprobar la ruta seguida por el `ping`.
   c) Haz `ping` de `SERV` a `PC1`. No debe funcionar, si lo hace repasa lo hecho. Analiza los registros para verificar que este tráfico ha sido interceptado.
   d) Comprueba también que no puedes establecer conexión telnet o web desde `PC1` a `SERV`
   e) Revisa las estadísticas de `netfilter`.
8. Cambia la configuración de `netfilter`, para que los clientes de cualquier red puedan conectarse con el servidor web que se ejecuta en `SERV`. Compruébala estableciendo una sesión `HTTP` desde `PC1`.

Configuración de un cortafuegos

Supongamos que las redes `Net1` y `Net2` pertenecen a una misma entidad, que se comunican a través de Internet.

9. En la red `Net1` se ha definido la siguiente política de seguridad:
   - Está permitido que las máquinas de la red se comuniquen con servidores HTTP externos.
   - Está permitido que las máquinas de la red puedan establecer conexiones `telnet` con las máquinas de la red `Net2`.
   - Ningún otro tráfico de la red `Net1` puede salir a Internet. De igual modo, ningún otro tráfico externo puede ingresar a `Net1`.
   - Debe rechazar cualquier tráfico externo enviado al propio cortafuegos.

   Escribe un *script* en `FW1` que implemente esta política de seguridad y registre los accesos no permitidos. Ejecútalo.

10. Comprueba que la configuración realizada cumple los criterios de seguridad indicados, para eso:
    a) Haz `ping` de `PC1` a `FW2`. No debe funcionar. Analiza los registros (*logs*) para confirmar que este tráfico ha sido interceptado por `FW1`. Ejecuta `traceroute` para comprobar la ruta seguida por el `ping`.
    b) Desde `PC1` establece una sesión web con `SERV` y otra con `FW2`. Ambas deben funcionar.
    c) Intenta hacer un `ping` de `FW2` a `FW1`. No debe funcionar. Revisa los registros.

11. En la red `Net2` se ha definido la siguiente política de seguridad:
    - Solo se permitirá el tráfico de sesiones `HTTP` y `telnet` si provienen de la red `Net1`.
    - No se permitirá ningún otro tráfico entrante/saliente a/desde la red.
    - `FW2` únicamente aceptará tráfico `ICMP` y solo con máquinas de `Net2`.

    Escribe un *script* en `FW2` que implemente esta política de seguridad y registre los accesos no permitidos. Ejecútalo.

12. Comprueba que la configuración realizada cumple los criterios de seguridad indicados, para eso repite las pruebas anteriores y también haz lo siguiente:
    a) Establece una sesión web de `PC1` con `SERV`. Debe funcionar, igual que lo hacía anteriormente.
    b) Sesión web y `telnet` de `FW1` con `SERV`. Si `FW1` está en `Net1` ¿por qué no funciona?.
    c) Haz `ping` de `SERV` a `FW2`. Debe funcionar.
    d) Establece una sesión web de `PC1` con `FW2/eth0`. Antes funcionaba pero ahora no debe funcionar (filtrado por `FW2`). Y si la intentas establecer con la interfaz `FW2/eth2` ¿qué sucederá? Si funciona, razona porqué puede suceder; si no funciona, plantea una posible solución para que lo consiga sobre esa interfaz y pruébala.

Esquema *script*:

```
# Declaración direcciones
NET1=192.0.2/24
# Borrar reglas y contadores
iptables -F
iptables -Z
# Aceptar tráfico entrante y saliente por loopback
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT
# Reglas de filtrado
iptables -A cadena -d $NET1 ...
...
# Rechazar todo lo demás, y guardar registro
iptables -A cadena -j LOG --log-prefix "DROP-HOST-CADENA"
iptables -A cadena -j DROP
```

Nota: en *cursiva* valores a sustituir

ANEXO: Información técnica

Netfilter

El kernel de nuestros sistemas Linux está construido con *netfilter*, un software para interceptar y manipular el tráfico de red. Cuando se pasa un datagrama a *netfilter*, lo que éste hace es revisar un conjunto de reglas buscando aquellas que sean de aplicación al datagrama en cuestión. Una regla tiene dos partes: una **condición** (por ejemplo, que el datagrama vaya dirigido a un puerto en concreto), y una **acción** a aplicar en caso de que el datagrama cumpla la condición (por ejemplo, ser desechado).

Las reglas están organizadas en **tablas** (o, en inglés, *tables*), entre otras: la tabla de filtrado (`filter table`), la tabla de traducciones (`nat table`), y la tabla de transformaciones (`mangle table`). En este laboratorio trabajamos sobre la tabla de filtrado.

Dentro de cada tabla, las reglas se agrupan en **cadenas** (*chains*) según el tipo de datagramas a los que se las aplica. La tabla de filtrado está organizada en tres cadenas:

- `INPUT`: se aplica a los datagramas dirigidos a nuestra propia máquina
- `OUTPUT`: se aplica a los datagramas generados por nuestra máquina
- `FORWARD`: cuando la máquina está configurada para actuar como encaminador, se aplica a los datagramas reenviados.

Cada datagrama se coteja con las reglas de la cadena que corresponda, una por una. Cuando cumple la condición de una regla, se completará la acción establecida por esa regla. Hay dos tipos de evento:

- Final del procesamiento del datagrama. Es lo que ocurre con las acciones `DROP`, `REJECT` y `ACCEPT`, entre otros. Con la acción `ACCEPT`, el datagrama supera el filtro, es decir, será enviado (cadena `OUTPUT`), recibido (cadena `INPUT`) o reenviado (cadena `FORWARD`, si corresponde). Con la acción `DROP`, el datagrama es rechazado. También con la acción `REJECT`, pero mando mensaje al origen comunicándolo.
- No supone el final del procesamiento del datagrama. En este caso, una vez ejecutada la acción especificada, se sigue cotejando el datagrama con la siguiente regla de la cadena. Es el caso de la acción `LOG` que se usa cuando se desea registrar la siguiente acción. Para ello, en la tabla hay dos reglas que tienen la misma condición con dos acciones: por un lado, la acción `LOG` que registra el evento, y por otro lado, las acciones `ACCEPT`, `REJECT` o `DROP` para el cumplimiento de la acción.

Si el datagrama alcanza el final de la cadena se le aplica la acción establecida por defecto para dicha cadena, que puede ser definida con el comando *`-P (policy)`*. Si no se ha definido una acción por defecto, se aplica `ACCEPT`.

`iptables`

Es la aplicación usada para configurar y gestionar *netfilter*. Su uso puede ser bastante complejo, por lo que es habitual acudir al manual on-line para consultar detalles u opciones menos habituales. Su sintaxis es:

```
iptables [-t nombre_tabla] comando nombre_cadena parámetro_1 argumento_1 ...
    parámetro_N argumento_N -j acción
```

- `-t`, parámetro para indicar con qué tabla quieres trabajar. Por defecto, la tabla usada es `filter`, por lo que no especificaremos la opción `-t` en este laboratorio.

- `comando`, más usados son `-A` (añadir una regla a una cadena), `-D` (borrar una regla de una cadena), `-F` (borrar todas las reglas de una cadena), `-I` (insertar una regla en la cadena), y `-L` (mostrar las reglas de una cadena). Los parámetros y argumentos varían según el comando.

Ejemplo: Como guardar un evento en el registro del sistema:

```
iptables -A FORWARD -p tcp -j LOG --log-prefix "Tráfico TCP detectado y rechazado"
iptables -A FORWARD -p tcp -j DROP
```

Se aplican ambas reglas, la primera detecta el tráfico `TCP` reenviado, y lo guarda en el fichero *log* del sistema[1]; la segunda detecta el mismo tráfico y lo rechaza.

Existe una herramienta para grabar y cargar la configuración automáticamente, llamada `iptables-save` e `iptables-restore`, aunque es más recomendable utilizar un *script*.

Más información:

- http://www.ticarte.com/contenido/iptables-conceptos-generales-para-configurar-un-cortafuegos
- https://help.ubuntu.com/community/IptablesHowTo
- https://www.digitalocean.com/community/tutorials/a-deep-dive-into-iptables-and-netfilter-architecture
- https://artica.es/docs/Guia_Netfilter.pdf
- *man iptables*

Conocer `iptables` (para hacer en casa):

1. Describe lo que hace cada una de estas reglas:

```
iptables -F
iptables -L
iptables -A OUTPUT -p icmp -j ACCEPT
iptables -D INPUT 2
iptables -A INPUT -j LOG --log-prefix "Filtrado INPUT:"
iptables -I INPUT 3 -j DROP
iptables -A FORWARD -p icmp -j ACCEPT
iptables -A FORWARD -d 158.227.112.1 -p tcp --dport 23 -j ACCEPT
iptables -A FORWARD -s 158.227.112.0/24 -j DROP
iptables -A FORWARD -p icmp --icmp-type echo-request -j ACCEPT
iptables -L FORWARD -n -x -v --line-numbers
```

2. ¿Cuál es la diferencia entre las dos acciones siguientes?

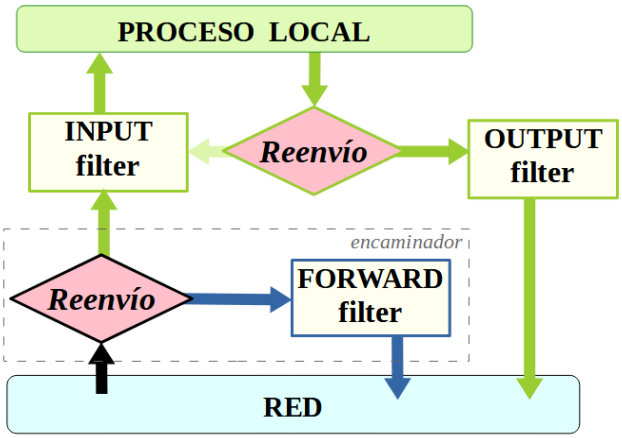
```
iptables -A OUTPUT -j DROP
iptables -A OUTPUT -j REJECT
```

3. ¿Qué efecto tiene la siguiente regla colocada la última en su cadena?

```
iptables -A nombre_cadena -j ACCEPT
```

[1] *En principio, los mensajes logs se graban en el lugar indicado en la configuración syslog del sistema (/etc/syslog.conf). En nuestra simulación los mensajes logs se guardan en el fichero /var/log/syslog en la máquina virtual.*

Recorrido (simplificado) de un datagrama a través de las tablas de filtrado:



A través de una cadena: