ASICI

Asociación Internacional de Ciberseguridad

Guía Comandos del Firewall iptables

Firewall iptables

`iptables` es una herramienta de línea de comandos utilizada para configurar las tablas de reglas de filtrado de paquetes en el kernel de Linux. Es el firewall de facto para la mayoría de las distribuciones Linux y proporciona un control extremadamente granular sobre cómo el tráfico de red entra, sale y se reenvía a través de un sistema. A diferencia de UFW, que es una abstracción, `iptables` interactúa directamente con el marco `netfilter` del kernel, lo que le otorga una inmensa flexibilidad y potencia, aunque también una mayor complejidad.

1. Conceptos Fundamentales de `iptables`

Para entender `iptables`, es crucial comprender sus componentes principales:

- **Tablas (Tables):** Son colecciones de cadenas que definen diferentes tipos de procesamiento de paquetes. Las tablas más comunes son:
    - **`filter` (por defecto):** Es la tabla más utilizada para el filtrado de paquetes. Decide si un paquete debe ser permitido o bloqueado. Si no se especifica una tabla, `filter` es la que se usa.
    - **`nat` (Network Address Translation):** Se usa para la traducción de direcciones de red (NAT), permitiendo que múltiples dispositivos compartan una única dirección IP pública, o para redireccionar puertos.
    - **`mangle`:** Se usa para modificar la cabecera de los paquetes IP (por ejemplo, alterar el TTL).
    - **`raw`:** Se usa para marcar paquetes que deben eximirse del seguimiento de conexión (`conntrack`).
- **Cadenas (Chains):** Son listas de reglas que el kernel examina en un orden predefinido. Cada paquete que llega o sale del sistema atraviesa una o más cadenas. Las cadenas predefinidas más importantes son:
    - **`INPUT`:** Para paquetes destinados al propio sistema.
    - **`OUTPUT`:** Para paquetes generados por el propio sistema y que salen.
    - **`FORWARD`:** Para paquetes que no están destinados al propio sistema, sino que deben ser reenviados a otro destino (enrutamiento).
    - **`PREROUTING` (en tablas `nat`/`mangle`/`raw`):** Para paquetes que entran al sistema *antes* de que se tome la decisión de enrutamiento.
    - **`POSTROUTING` (en tablas `nat`/`mangle`):** Para paquetes que salen del sistema *después* de que se haya tomado la decisión de enrutamiento.
- **Reglas (Rules):** Son condiciones específicas que un paquete debe cumplir para que se aplique una determinada **acción (Target)**. Una regla se compone de:
    - **Condiciones/Criterios:** Especifican qué paquetes coinciden con la regla (ej. dirección IP de origen/destino, puerto, protocolo, interfaz).
    - **Acción/Objetivo (Target):** Qué hacer con el paquete si coincide con la regla.
        - **`ACCEPT`:** Permite el paquete.
        - **`DROP`:** Descarta silenciosamente el paquete (el remitente no recibe ninguna notificación). Es como si el paquete nunca hubiera existido.

    - **`REJECT`:** Descarta el paquete y envía un mensaje de error al remitente (ej. ICMP Port Unreachable).
    - **`LOG`:** Registra el paquete en los logs del sistema (normalmente `/var/log/syslog` o `/var/log/messages`).
    - **`RETURN`:** Detiene el procesamiento en la cadena actual y vuelve a la cadena que la llamó.
    - **`SNAT` (Source NAT):** Modifica la dirección IP de origen del paquete (tabla `nat`, cadena `POSTROUTING`).
    - **`DNAT` (Destination NAT):** Modifica la dirección IP de destino del paquete (tabla `nat`, cadena `PREROUTING`).
    - **`MASQUERADE`:** Una forma de SNAT para conexiones con direcciones IP dinámicas (ej. conexiones DSL o DHCP).

2. Sintaxis Básica de `iptables`

La sintaxis general es:

iptables -t <tabla> <comando> <cadena> [criterios] -j <acción>

- `sudo` es casi siempre necesario para ejecutar comandos `iptables`.
- Si no se especifica `-t <tabla>`, se asume la tabla `filter`.

3. Comandos de Gestión de Reglas

Estos comandos son para añadir, eliminar, listar y manipular reglas.

- **`-A` / `--append`:** Añade una regla al final de una cadena.
    - Ejemplo (permitir SSH entrante): `sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT`
- **`-I` / `--insert`:** Inserta una regla en una posición específica de una cadena (por defecto, la primera posición).
    - Ejemplo (insertar en la posición 1): `sudo iptables -I INPUT 1 -s 192.168.1.100 -j ACCEPT`
- **`-D` / `--delete`:** Elimina una regla. Puedes especificar la regla completa o su número.
    - Ejemplo (eliminar por regla): `sudo iptables -D INPUT -p tcp --dport 22 -j ACCEPT`
    - Ejemplo (eliminar por número): `sudo iptables -D INPUT 3` (primero, usa `-L --line-numbers` para ver los números).
- **`-R` / `--replace`:** Reemplaza una regla existente.
    - Ejemplo: `sudo iptables -R INPUT 3 -p tcp --dport 80 -j DROP`
- **`-L` / `--list`:** Lista todas las reglas en una o todas las cadenas de una tabla.
    - Ejemplo (listar todas las reglas de la tabla `filter`): `sudo iptables -L`

    - Ejemplo (listar reglas con detalles y números): `sudo iptables -L --line-numbers -v`
    - Ejemplo (listar reglas de la tabla `nat`): `sudo iptables -t nat -L`
- **`-F` / `--flush`:** Elimina todas las reglas de una cadena o de todas las cadenas de una tabla. **¡CUIDADO!** Esto puede dejar tu sistema expuesto o inaccesible.
    - Ejemplo (eliminar todas las reglas de la cadena `INPUT`): `sudo iptables -F INPUT`
    - Ejemplo (eliminar todas las reglas de todas las cadenas en la tabla `filter`): `sudo iptables -F`
    - Ejemplo (eliminar todas las reglas en todas las tablas): `sudo iptables -t nat -F; sudo iptables -t mangle -F; sudo iptables -t filter -F`
- **`-Z` / `--zero`:** Pone a cero los contadores de paquetes y bytes para las reglas.
    - Ejemplo: `sudo iptables -Z`
- **`-N` / `--new-chain`:** Crea una nueva cadena definida por el usuario.
    - Ejemplo: `sudo iptables -N LOG_AND_DROP`
- **`-X` / `--delete-chain`:** Elimina una cadena definida por el usuario (debe estar vacía y no referenciada).
    - Ejemplo: `sudo iptables -X LOG_AND_DROP`
- **`-P` / `--policy`:** Establece la política por defecto para una cadena predefinida. Esto es lo que sucede si un paquete no coincide con ninguna regla en la cadena.
    - Ejemplo (bloquear todo lo entrante por defecto): `sudo iptables -P INPUT DROP`
    - Ejemplo (permitir todo lo saliente por defecto): `sudo iptables -P OUTPUT ACCEPT`
    - Ejemplo (bloquear todo lo reenviado por defecto): `sudo iptables -P FORWARD DROP`

4. Criterios para Coincidir Paquetes (Matching Criteria)

Estos son los criterios que puedes usar para que una regla coincida con un paquete.

- **`-s` / `--source <dirección_IP>`:** IP de origen (puede ser una IP individual o una red CIDR, ej. `192.168.1.0/24`).
    - Ejemplo: `iptables -A INPUT -s 10.0.0.5 -j ACCEPT`
- **`-d` / `--destination <dirección_IP>`:** IP de destino.
    - Ejemplo: `iptables -A OUTPUT -d 8.8.8.8 -j ACCEPT`
- **`-p` / `--protocol <protocolo>`:** Protocolo (tcp, udp, icmp, all).
    - Ejemplo: `iptables -A INPUT -p tcp -j ACCEPT`
- **`--sport <puerto>`:** Puerto de origen TCP/UDP.
    - Ejemplo: `iptables -A INPUT -p tcp --sport 80 -j ACCEPT`
- **`--dport <puerto>`:** Puerto de destino TCP/UDP.
    - Ejemplo: `iptables -A INPUT -p tcp --dport 22 -j ACCEPT`

- **`-i` / `--in-interface <interfaz>`:** Interfaz de red de entrada (ej. `eth0`, `lo`).
    - Ejemplo: `iptables -A INPUT -i eth0 -p tcp --dport 80 -j ACCEPT`
- **`-o` / `--out-interface <interfaz>`:** Interfaz de red de salida (ej. `eth0`, `lo`).
    - Ejemplo: `iptables -A OUTPUT -o eth0 -p udp --dport 53 -j ACCEPT`
- **`--tcp-flags <máscara> <flags_a_coincidir>`:** Coincide con flags TCP específicos.
    - Ejemplo (paquetes SYN, para detectar escaneos): `iptables -A INPUT -p tcp --tcp-flags ALL SYN -j LOG`
- **`--syn`:** Un atajo para `--tcp-flags SYN,RST,ACK SYN`. Coincide con paquetes que inician una conexión TCP.
    - Ejemplo: `iptables -A INPUT -p tcp --dport 80 --syn -m limit --limit 25/minute --limit-burst 100 -j ACCEPT` (permite nuevas conexiones HTTP, limitando la tasa).
- **`-m state --state <estados>`:** Módulo para coincidir con estados de conexión.
    - Estados: `NEW` (nueva conexión), `ESTABLISHED` (parte de una conexión ya establecida), `RELATED` (conexión relacionada con una establecida, ej. FTP DATA), `INVALID` (paquete inválido).
    - Ejemplo (permitir respuestas a conexiones salientes): `iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT` (CRUCIAL para que las conexiones salientes funcionen).
- **`-m mac --mac-source <dirección_MAC>`:** Coincide con la dirección MAC de origen.
    - Ejemplo: `iptables -A INPUT -m mac --mac-source 00:00:00:00:00:01 -j DROP`
- **`-m limit --limit <tasa>[/seg|min|hora|día] --limit-burst <ráfaga>`:** Limita el número de veces que una regla puede ser activada. Útil para proteger contra ataques de fuerza bruta o de denegación de servicio.
    - Ejemplo: `iptables -A INPUT -p tcp --dport 22 -m state --state NEW -m limit --limit 3/minute --limit-burst 3 -j ACCEPT` (permite 3 intentos de conexión SSH por minuto).

5. Configuración de NAT (Network Address Translation)

La tabla `nat` se utiliza para reescribir direcciones IP y puertos.

- **`PREROUTING` (DNAT):** Para cambiar la IP/puerto de *destino* de los paquetes entrantes *antes* de que lleguen al sistema.
    - Ejemplo (redireccionar tráfico del puerto 80 del host al puerto 8080 de un servidor web interno 192.168.1.10): `sudo iptables -t nat -A PREROUTING -p tcp --dport 80 -j DNAT --to-destination 192.168.1.10:8080`

- **`POSTROUTING` (SNAT/MASQUERADE):** Para cambiar la IP de *origen* de los paquetes salientes *después* de que el enrutamiento haya ocurrido.
    - **`SNAT`:** Se usa cuando tienes una IP pública estática. `sudo iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to-source <tu_ip_publica>`
    - **`MASQUERADE`:** Es un tipo especial de SNAT para cuando tu IP pública es dinámica (ej. DHCP). `sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE` (muy común para compartir internet).

6. Persistencia de Reglas

Las reglas de `iptables` son volátiles, lo que significa que se pierden al reiniciar el sistema. Necesitas guardarlas para que persistan.

- **Guardar reglas:**
    - Debian/Ubuntu: `sudo iptables-save > /etc/iptables/rules.v4`
    - Red Hat/CentOS (usando `iptables-services`): `sudo service iptables save` o `sudo /sbin/iptables-save > /etc/sysconfig/iptables`
- **Restaurar reglas (al inicio del sistema):**
    - Debian/Ubuntu: Configurar un script de inicio o usar el paquete `netfilter-persistent`. `sudo apt install netfilter-persistent` (esto creará y usará los archivos `/etc/iptables/rules.v4` y `rules.v6`).
    - Red Hat/CentOS: El servicio `iptables` ya lo hace automáticamente al iniciar. `sudo systemctl enable iptables` `sudo systemctl start iptables`

7. Ejemplos de Configuración Común

Configuración de Firewall Básico y Seguro:

1. **Establecer políticas por defecto a DROP (muy importante):** `sudo iptables -P INPUT DROP` `sudo iptables -P FORWARD DROP` `sudo iptables -P OUTPUT ACCEPT`
    - *Explicación:* Bloquea todo lo entrante y reenviado. Permite todo lo saliente (para que el sistema pueda funcionar y hacer peticiones).
2. **Permitir tráfico local (loopback):** `sudo iptables -A INPUT -i lo -j ACCEPT` `sudo iptables -A OUTPUT -o lo -j ACCEPT`
    - *Explicación:* Permite que las aplicaciones se comuniquen consigo mismas.
3. **Permitir conexiones ya establecidas y relacionadas:** `sudo iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT`

    - *Explicación:* Crucial. Permite que las respuestas a tus conexiones salientes vuelvan, y permite que las conexiones relacionadas (como el canal de datos de FTP) funcionen.
4. **Permitir SSH (puerto 22) desde cualquier lugar (Opcional: limitar):** `sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT`
    - *Mejorado (limitando SSH para evitar fuerza bruta):* `sudo iptables -A INPUT -p tcp --dport 22 -m state --state NEW -m limit --limit 3/minute --limit-burst 3 -j ACCEPT` `sudo iptables -A INPUT -p tcp --dport 22 -j DROP`
        - *Explicación:* La primera regla permite un máximo de 3 nuevas conexiones SSH por minuto. La segunda regla `DROP` descarta cualquier intento que exceda ese límite.
5. **Permitir HTTP (puerto 80) y HTTPS (puerto 443) entrantes:** `sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT` `sudo iptables -A INPUT -p tcp --dport 443 -j ACCEPT`
6. **Registrar paquetes descartados (para depuración y seguridad):** `sudo iptables -A INPUT -m limit --limit 5/min -j LOG --log-prefix "iptables-dropped: "` `sudo iptables -A FORWARD -m limit --limit 5/min -j LOG --log-prefix "iptables-forward-dropped: "`
7. **Guardar las reglas (después de configurarlas):** `sudo iptables-save > /etc/iptables/rules.v4` (para Debian/Ubuntu con `netfilter-persistent`)

Configuración de NAT (Compartir Internet):

1. **Habilitar el reenvío IP en el kernel:** `sudo sysctl -w net.ipv4.ip_forward=1`
    - Para que sea persistente, edita `/etc/sysctl.conf` y descomenta `net.ipv4.ip_forward=1`, luego `sudo sysctl -p`.
2. **Configurar MASQUERADE en la interfaz de salida a Internet (ej. `eth0`):** `sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE`
    - *Explicación:* Permite que las IPs de la red interna accedan a Internet a través de la IP de `eth0`.
3. **Permitir el reenvío de tráfico en la tabla `filter`:** `sudo iptables -A FORWARD -i eth1 -o eth0 -j ACCEPT` (permite tráfico de la red interna `eth1` a Internet `eth0`) `sudo iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT` (permite el retorno de las conexiones)

8. Diferencias con UFW

- **Nivel de Abstracción:** UFW es una interfaz de alto nivel, `iptables` es de bajo nivel.
- **Sintaxis:** UFW usa una sintaxis simple (ej. `allow 80`), `iptables` requiere especificar tablas, cadenas, protocolos, puertos, etc.

- **Complejidad:** `iptables` es mucho más complejo de aprender y usar, pero ofrece control absoluto.
- **NAT/Enrutamiento:** `iptables` es necesario para configurar NAT y reglas de enrutamiento avanzadas; UFW solo ofrece algunas opciones básicas de enrutamiento.
- **Persistencia:** `iptables` requiere un paso manual o un servicio para guardar/restaurar reglas. UFW lo hace automáticamente.

Consideraciones al Usar `iptables`

- **¡Peligro de Bloqueo!:** Un error en la configuración de `iptables` (especialmente al usar `DROP` en las políticas por defecto o `-F`) puede dejar tu servidor inaccesible. Siempre prueba las reglas con precaución y ten un plan de recuperación (consola física, acceso remoto alternativo).
- **Estado de la Conexión:** El módulo `--state` es fundamental para que el firewall funcione correctamente y permita el retorno de las conexiones salientes.
- **Orden de las Reglas:** El orden es crítico. Las reglas se evalúan de arriba hacia abajo en una cadena. Una vez que un paquete coincide con una regla, la acción se realiza y no se evalúan más reglas en esa cadena (a menos que el objetivo sea una cadena personalizada que luego retorne).
- **Módulos:** `iptables` utiliza varios módulos (`-m <módulo>`) para capacidades adicionales como el seguimiento de estado (`state`), limitación de tasa (`limit`), o coincidencia MAC (`mac`).
- **`ip6tables`:** Existe un comando análogo, `ip6tables`, para gestionar el firewall IPv6.

`iptables` es la espina dorsal del firewall de Linux. Dominarlo te da un poder inmenso para proteger y controlar tu infraestructura de red. Sin embargo, su complejidad exige un conocimiento profundo y un uso cuidadoso.