Day 42

Exploitation Analyst

Firewalls and TCP wrappers:

Iptables-persistent:

What is Iptables-persistent?

iptables-persistent is a helper package that makes your firewall rules permanent across reboots.

Normally, when you add rules with iptables, they live only in the system’s memory. As soon as you reboot, they’re gone. iptables-persistent solves this by:

1. **Saving rules** → When you run sudo netfilter-persistent save, it writes the current active iptables rules to /etc/iptables/rules.v4 (IPv4) and /etc/iptables/rules.v6 (IPv6).
2. **Restoring rules at boot** → During system startup, it automatically loads those saved rules back into the kernel’s firewall (netfilter).

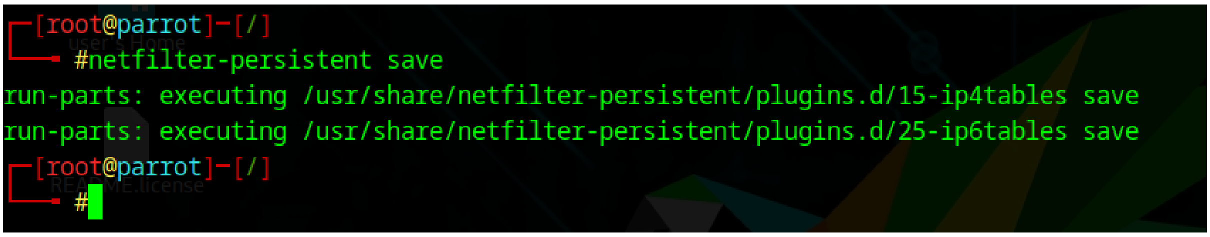
Exploring Iptables-persistent:

Steps:

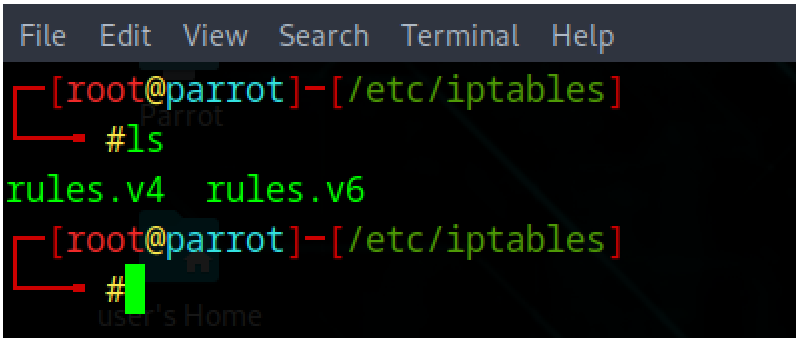
Command:

sudo netfilter-persistent save

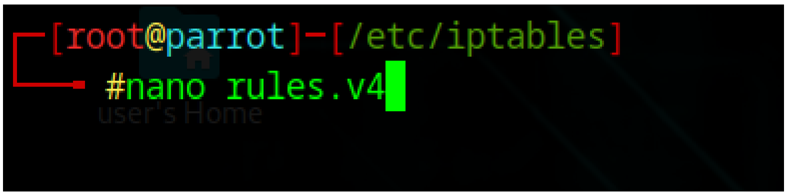
- This takes your **current in-memory iptables rules** and writes them into /etc/iptables/rules.v4 (for IPv4) and /etc/iptables/rules.v6 (for IPv6).
- So if you reboot later, the firewall will restore from those saved files.



```
┌─[root@parrot]─[/]
└──╼ #netfilter-persistent save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/15-ip4tables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/25-ip6tables save
┌─[root@parrot]─[/]
└──╼ #
```

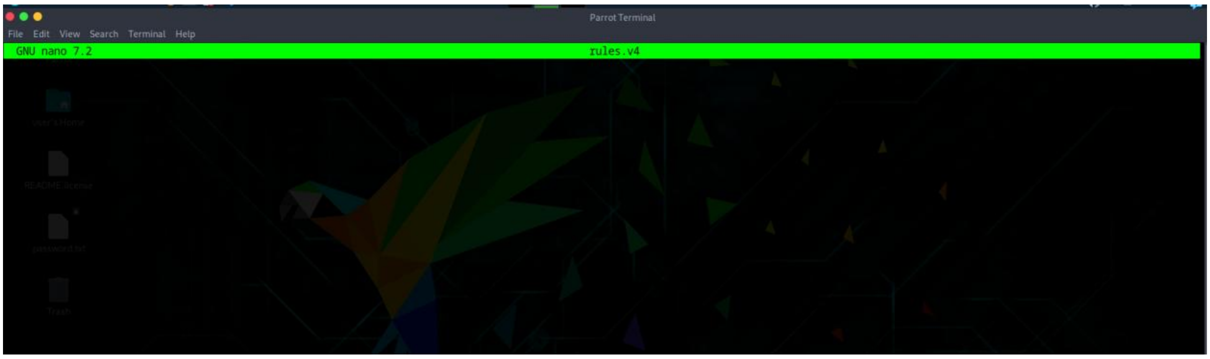


```
┌─[root@parrot]─[/etc/iptables]
└──╼ #ls
rules.v4  rules.v6
┌─[root@parrot]─[/etc/iptables]
└──╼ #
```

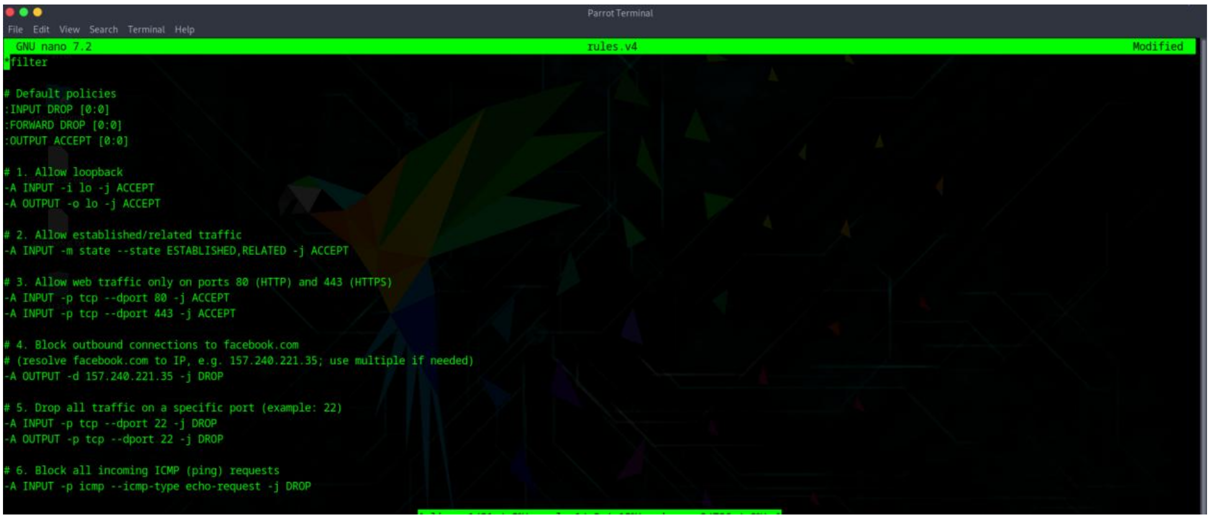


```
┌─[root@parrot]─[/etc/iptables]
└──╼ #nano rules.v4
```

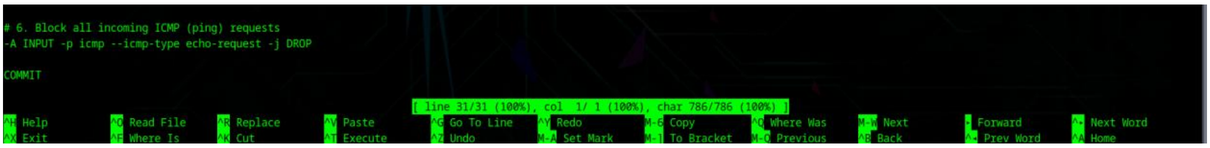
The opened editor will show it empty:



Write like this:



```
*filter

# Default policies
:INPUT DROP [0:0]
:FORWARD DROP [0:0]
:OUTPUT ACCEPT [0:0]

# 1. Allow loopback
-A INPUT -i lo -j ACCEPT
-A OUTPUT -o lo -j ACCEPT

# 2. Allow established/related traffic
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# 3. Allow web traffic only on ports 80 (HTTP) and 443 (HTTPS)
-A INPUT -p tcp --dport 80 -j ACCEPT
-A INPUT -p tcp --dport 443 -j ACCEPT

# 4. Block outbound connections to facebook.com
# (resolve facebook.com to IP, e.g. 157.240.221.35; use multiple if needed)
-A OUTPUT -d 157.240.221.35 -j DROP

# 5. Drop all traffic on a specific port (example: 22)
-A INPUT -p tcp --dport 22 -j DROP
-A OUTPUT -p tcp --dport 22 -j DROP

# 6. Block all incoming ICMP (ping) requests
-A INPUT -p icmp --icmp-type echo-request -j DROP
```



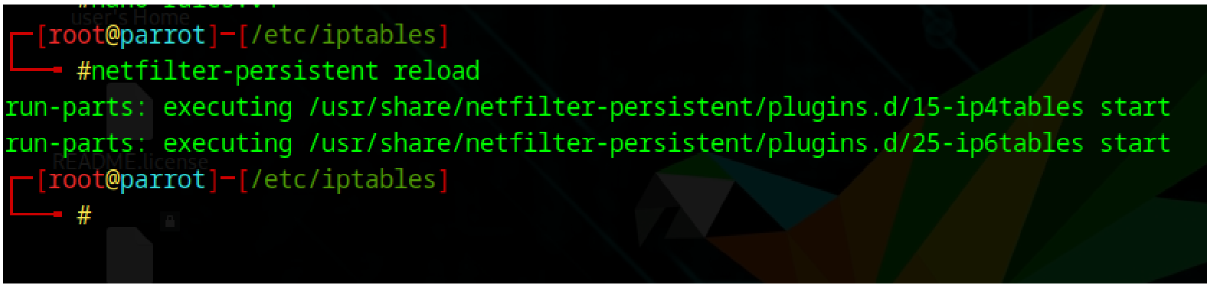
```
# 6. Block all incoming ICMP (ping) requests
-A INPUT -p icmp --icmp-type echo-request -j DROP

COMMIT
                                     [ line 31/31 (100%), col  1/ 1 (100%), char 786/786 (100%) ]
```

- *filter → means we’re working in the **filter table** (used for packet filtering).
- :INPUT DROP [0:0] → set default policy for INPUT chain to **DROP**. Any packet not matching a rule will be dropped.
- :FORWARD DROP [0:0] → default policy for FORWARD chain is **DROP** (good if your system is not a router).
- :OUTPUT ACCEPT [0:0] → default policy for OUTPUT is **ACCEPT** (outgoing traffic allowed unless explicitly blocked).

1. -A INPUT -i lo -j ACCEPT / -A OUTPUT -o lo -j ACCEPT
   → Allow loopback traffic (localhost communication, 127.0.0.1).

2. -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
   → Allow responses to connections you initiated (like replies from a website).
3. -A INPUT -p tcp --dport 80 -j ACCEPT and --dport 443
   → Allow only HTTP (80) and HTTPS (443) connections from outside.
4. -A OUTPUT -d 157.240.221.35 -j DROP
   → Block outbound traffic to **facebook.com's IP**.
5. -A INPUT -p tcp --dport 22 -j DROP and -A OUTPUT -p tcp --dport 22 -j DROP
   → Block all SSH traffic (port 22) both inbound and outbound.
6. -A INPUT -p icmp --icmp-type echo-request -j DROP
   → Drop all ping requests (server won't reply to pings).

- COMMIT → saves these rules to the kernel when restoring.



```
┌─[root@parrot]─[/etc/iptables]
└──╼ #netfilter-persistent reload
run-parts: executing /usr/share/netfilter-persistent/plugins.d/15-ip4tables start
run-parts: executing /usr/share/netfilter-persistent/plugins.d/25-ip6tables start
┌─[root@parrot]─[/etc/iptables]
└──╼ #
```

- This clears the current in-memory firewall rules.
- Then reloads rules fresh from /etc/iptables/rules.v4 and /etc/iptables/rules.v6.
- Useful after editing the rules file manually.

What happens in background?

When you run sudo netfilter-persistent save, it uses iptables-save to take a snapshot of the current firewall rules stored in the kernel's netfilter tables and writes them into /etc/iptables/rules.v4 and /etc/iptables/rules.v6 for persistence. These files are not just plain text but instructions that rebuild the firewall rules in the kernel whenever they are reloaded. During sudo netfilter-persistent reload, the system flushes any active rules in memory and re-applies the saved configuration using iptables-restore, repopulating the kernel's netfilter hooks exactly as written in the file. Once active, every packet entering or leaving the system passes through netfilter inside the Linux kernel, where it is checked against chains (INPUT, OUTPUT, FORWARD) and evaluated rule by rule. If a packet matches a rule, the defined action such as ACCEPT, DROP, or REJECT is taken immediately; if nothing matches, the chain's default policy is applied. For example, if someone pings your server, the ICMP packet is caught by the DROP rule in the INPUT chain and silently discarded, while an outbound request to a blocked Facebook IP hits the OUTPUT chain and is denied. In essence, iptables-persistent ensures that rules you configure once are permanently saved and automatically enforced at boot, allowing the kernel's netfilter engine to continuously filter packets in real time.

--The End--