IPTABLES

The Linux Firewall

Presented By

Emin Asif A S

Introduction

- Network security is a primary consideration in any decision to host a website as the threats are becoming more widespread and persistent every day.
- We can convert a Linux server into:

A firewall while simultaneously being our home website's mail, web and DNS server.

A router that will use NAT and port forwarding to both protect your home network and have another web server on your home network while sharing the public IP address of ourfirewall.

What Is Iptables?

Originally, the most popular firewall/NAT package running on Linux was ipchains, but it had a number of shortcomings. To rectify this, the Netfilter organization decided to create a new product called iptables, giving it such improvements as:

- Better integration with the Linux kernel.
- Stateful packet inspection.
- Filtering packets.
- System logging.
- Better network address translation.

Considered a faster and more secure alternative to ipchains, iptables has become the default firewall package installed under RedHat and Fedora Linux.

Managing the iptables Server

Different Linux distributions use different daemon management systems.

- The most commonly used daemon management systems are SysV and Systemd.
- The daemon is **iptables**.

Armed with this information we can know how to:

- Start the daemons automatically on booting
- Stop, start and restart them later on during troubleshooting or when a configuration file change needs to be applied.

Packet Processing In iptables

All packets inspected by iptables pass through a sequence of built-in tables (queues) for processing.There are three tables in total.

- The first is the mangle table which is responsible for the alteration of quality of service bits in the TCP header.
- The second table is the filter queue which is responsible for packet filtering. It has three built-in chains in which you can place your firewall policy rules. These are the:

  Forward chain: Filters packets to servers protected by the firewall.

  Input chain: Filters packets destined for the firewall.

  Output chain: Filters packets originating from the firewall.

- The third table is the nat queue which is responsible for network address translation. It has two built-in chains; these are:

  Pre-routing chain: NATs packets when the destination address of the packet needs to be changed.

Post-routing chain: NATs packets when the source address of the packet needs to be changed

Check if iptables installed

- $ rpm -q iptables

  iptables-1.4.7-5.1.el6_2.x86_64
- use the -L switch to inspect the currently loaded rules:

  # iptables -L
- If iptables is not running, you can enable it by running:

  # system-config-securitylevel

Switch Operations

`-t <-table->`	tables include: filter, nat, mangle
`-j <target>`	Jump to the specified target chain when the packet matches the current rule.
`-A`	Append rule to end of a chain
`-F`	Flush. Deletes all the rules in the selected table
`-p <protocol-type>`	icmp, tcp, udp, and all
`-s <ip-address>`	source IP address
`-d <ip-address>`	destination IP address
`-i <interface-name>`	"input" interface on which the packet enters.
`-o <interface-name>`	"output" interface on which the packet exits

Targets And Jumps

- **ACCEPT** iptables stops further processing. The packet is **handed over** to the end application or the operating system for processing.
- **DROP** iptables **stops** further processing. The packet is **blocked**.
- **LOG** The packet information is sent to the syslog daemon for logging iptables continues processing with the next rule in the table.
- **REJECT** Works like the **DROP** target, but will also **return an error message** to the host sending the packet that the packet was blocked.
- **DNAT** Used to do **destination network address translation**. ie. rewriting the destination IP address of the packet.
- **SNAT** Used to do **source network address translation** rewriting the source IP address of the packet. The source IP address is user defined
- **MASQUERADE** Used to do Source Network Address Translation. By default the source IP address is the same as that used by the firewall's interface

Interfaces

#iptables -A INPUT -i <u>lo</u> -j ACCEPT

/*allows localhost <u>interface</u> 127.0.0.1*/

#iptables -A INPUT -i <u>eth0</u> -j ACCEPT

/*allows eth0 which is our internal LAN connection, (eth0 and eth1 are ethernet interfaces, they can be either internet or private network interfaces)*/

#iptables -A INPUT -i <u>ppp0</u> -j ACCEPT

/*allows ppp0 dialup modem which is our external internet connection*/

Common Extended Match Criteria

`-m multiport --sports <port, port>`

A variety of TCP/UDP source ports separated by commas. Unlike when `-m` isn't used, they do not have to be within a range.

`-m multiport --dports <port, port>`

A variety of TCP/UDP destination ports separated by commas. Unlike when `-m` isn't used, they do not have to be within a range.

`-m multiport --ports <port, port>`

A variety of TCP/UDP ports separated by commas. Source and destination ports are assumed to be the same and they do not have to be within a range.

`-m --state <state>`

The most frequently tested states are:

ESTABLISHED: The packet is part of a connection that has seen packets in both directions

NEW: The packet is the start of a new connection

RELATED: The packet is starting a new secondary connection. This is a common feature of such protocols such as an FTP data transfer, or an ICMP error.

Accept packets from trusted IP addresses

- To allow the packets from a single IP

```
    iptables -A INPUT -s 192.168.0.4 -j ACCEPT
```

[-s   source   -j   jump to the target action (here ACCEPT)]

- To allow incoming packets from a range of IP addresses

```
   iptables -A INPUT -s 192.168.0.0/24 -j ACCEPT
                 or
   iptables -A INPUT -s 192.168.0.0/255.255.255.0 -j ACCEPT
```

Ports and Protocols

```
[  -p   protocol  (tcp,udp,icmp,all) ]

[ --dport  destination port ]     [--sport   source port ]
```

- Accept tcp packets on destination port 6881 (bittorrent)

```
#iptables -A INPUT -p tcp --dport 6881 -j ACCEPT
```

- To include a port range

  Accept tcp packets on destination ports 6881-6890

```
#iptables -A INPUT -p tcp --dport 6881:6890 -j ACCEPT
```

Writing a Simple Rule Set

iptables -<u>P</u> INPUT ACCEPT

 //If connecting remotely we must first temporarily set the default <u>policy</u> on the INPUT chain to ACCEPT,otherwise we will be locked out of our server once we flush the current rules.

iptables -<u>F</u>

//to <u>flush</u> all existing rules so we start with a clean state from which to add new rules.

iptables -A INPUT -i <u>lo</u> -j ACCEPT   //to communicate with the localhost adaptor.

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

//to allow only the incoming packets that are part of an already established connection or related to and already established connection.

Writing a Simple Rule Set (Contd.)

# iptables -A INPUT -p tcp --dport 22 -j ACCEPT   // to prevent accidental lockouts when working on remote systems over an SSH connection.

# iptables -P INPUT DROP   //if an incoming packet does not match one of the following rules it will be dropped.

# iptables -P FORWARD DROP   //set the default policy on the FORWARD chain to DROP as we're not using our computer as a router so there should not be any packets passing through our computer.

# iptables -P OUTPUT ACCEPT   //set the default policy on the OUTPUT chain to ACCEPT as we want to allow all outgoing traffic (as we trust our users).

# iptables -L -v   //we can list (-L) the rules we've just added to check they've been loaded correctly.

# /sbin/service iptables save   //to save our rules so that next time we reboot our computer our rules are automatically reloaded

Masquerading (Many to One NAT)

Traffic from all devices on one or more protected networks will appear as if it originated from a single IP address on the Internet side of the firewall.

echo 1 > /proc/sys/net/ipv4/ip_forward   //to enable routing between internet & private network interfaces of the firewall.

Masquerading has been achieved using the POSTROUTING chain of the nat table,

```
#iptables -A POSTROUTING -t nat -o eth0 -s 192.168.1.0/24 -d 0/0 -j MASQUERADE
```

Use the FORWARD chain of the filter table. NEW and ESTABLISHED connections will be allowed outbound to the Internet,

```
#iptables -A FORWARD -t filter -o eth0 -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
```

But only packets related to ESTABLISHED connections will be allowed inbound.

```
#iptables -A FORWARD -t filter -i eth0 -m state --state ESTABLISHED,RELATED -j ACCEPT
```

This helps to protect the home network from anyone trying to initiate connections from the Internet

Recovering From A Lost Script

Sometimes the script you created to generate iptables rules may get corrupted or lost, To recover:

Export the iptables-save output to a text file named firewall-config

```
[root@bigboy tmp]# iptables-save > firewall-config
[root@bigboy tmp]# cat firewall-config
```

We can reload it into the active firewall rule set with the iptables-restore command.

```
[root@bigboy tmp]# iptables-restore < firewall-config
[root@bigboy tmp]# service iptables save
```

Troubleshooting iptables

- Checking The Firewall Logs

  Log and drop packets to the /var/log/messages file.

```
iptables -A OUTPUT -j LOG
iptables -A INPUT -j LOG
iptables -A FORWARD -j LOG

iptables -A OUTPUT -j DROP
iptables -A INPUT -j DROP
iptables -A FORWARD -j DROP
```

Allowing DNS Access To Your Firewall

```
#iptables -A OUTPUT -p udp -o eth0 --dport 53 --sport 1024:65535 \
    -j ACCEPT

#iptables -A INPUT -p udp -i eth0 --sport 53 --dport 1024:65535 \
    -j ACCEPT
```

Allowing WWW And SSH Access To Your Firewall

- Interface eth0 is the internet interface

```
#iptables -A OUTPUT -o eth0 -m state --state ESTABLISHED,RELATED -j ACCEPT
```

- Allow port 80 (www) and 22 (SSH) connections to the firewall

```
#iptables -A INPUT -p tcp -i eth0 --dport 22 --sport 1024:65535
  -m state --state NEW -j ACCEPT
#iptables -A INPUT -p tcp -i eth0 --dport 80 --sport 1024:65535
  -m state --state NEW -j ACCEPT
```

Allowing Your Firewall To Access The Internet

- Allow port 80 (www) and 443 (https) connections from the firewall

```
#iptables -A OUTPUT -j ACCEPT -m state --state NEW,ESTABLISHED,RELATED
  -o eth0 -p tcp -m multiport --dports 80,443 --sport 1024:65535
```

- Allow previously established connections

  Interface eth0 is the internet interface

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

Allow Your Home Network To Access The Firewall

Allow all bidirectional traffic from your firewall to the protected network

Interface eth1 is the private network interface

```
#iptables -A INPUT   -j ACCEPT -p all -s 192.168.1.0/24 -i eth1
#iptables -A OUTPUT  -j ACCEPT -p all -d 192.168.1.0/24 -o eth1
```

THANK YOU