Linux Firewalls

Frank Kuse, AfNOG 2018

About this presentation

Based on a previous talk by Kevin Chege and Chris Wilson, with thanks!

You can access this presentation at:

- Online: http://afnog.github.io/sse/firewalls/

- Local: http://www.ws.afnog.org/afnog2017/sse/firewalls/index.html

- Github: https://github.com/afnog/sse/blob/master/firewalls/presentation.md

- Download PDF:
  http://www.ws.afnog.org/afnog2017/sse/firewalls/presentation.pdf

  Download Exercises: http://www.ws.afnog.org/afnog2017/sse/firewalls/Exercises.pdf

WhatisaFirewall?

AdvancedFirewalls

- Basic firewalls are packet filters

- Can't always make a decision based on one packet (examples?)

- Stateful firewalls (connection table)

- Application layer (L7) filtering/inspection/IDS

- Redundant firewalls with synchronisation

- VPNs and SSL "VPNs"

StatefulFirewalls

.................▶ unusual event

──────────▶ client/receiver path

──────────▶ server/sender path

▶Step 2 of the 3wayhandshake◀   SYN/SYN+ACK

(Start) **CLOSED**

CONNECT/SYN ▶Step 1 of the 3wayhandshake◀

LIST **EN/**

CLOSE/

CLOSE/

CLOSE/

LISTEN
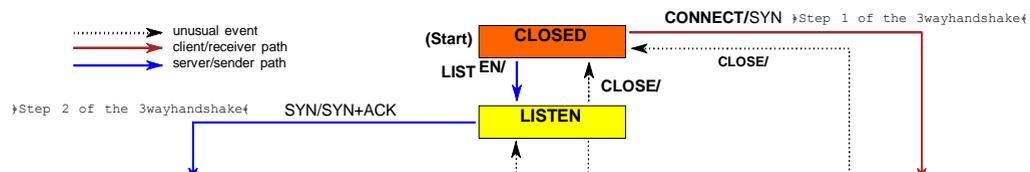
LimitationsofFirewalls

BlockingWebsites
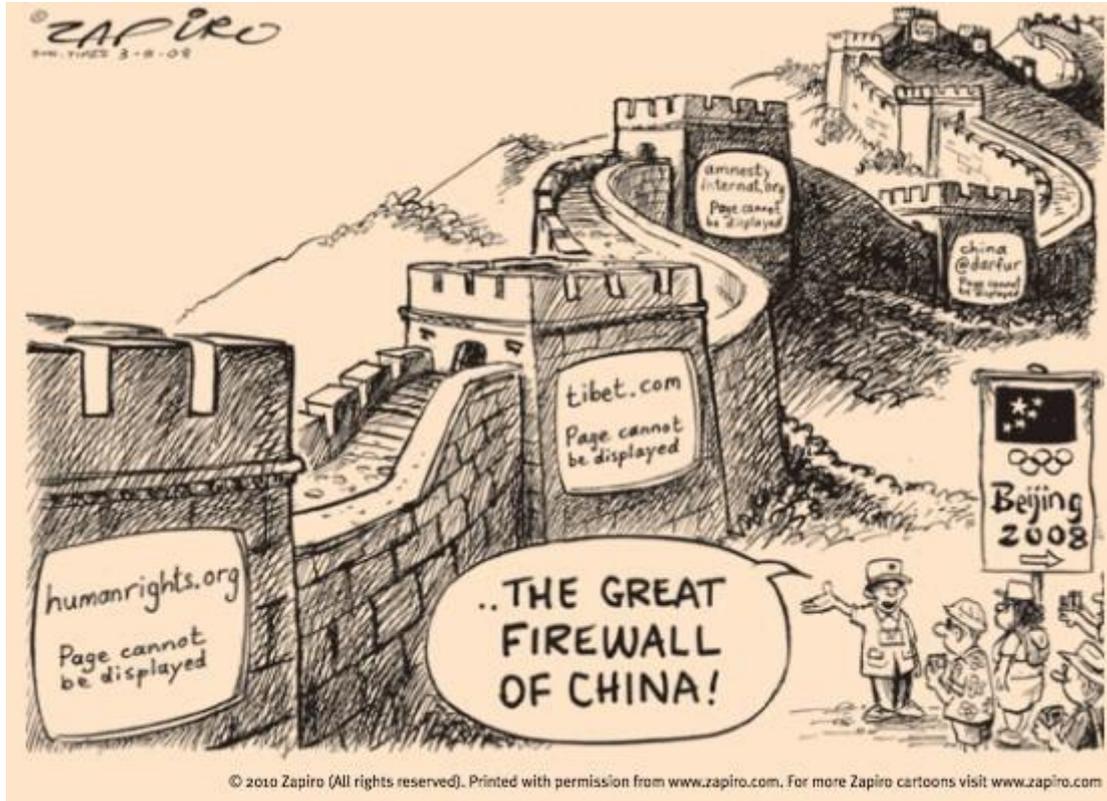
Whatdofirewallsfilter?

Src IP	Dst IP	Pro	Spt	Dpt	Flg
1.2.3.4	5.6.7.8	TCP	xx	80	SYN

Typical features

- Rulesets (lists of rules, read in order)

- Rules (IF this THEN that)

- Match conditions
    - interface, IP address, protocol, port, time, contents

- Actions
    - accept, drop, reject, jump to another table, return

- Default policy

iptables/netfilter
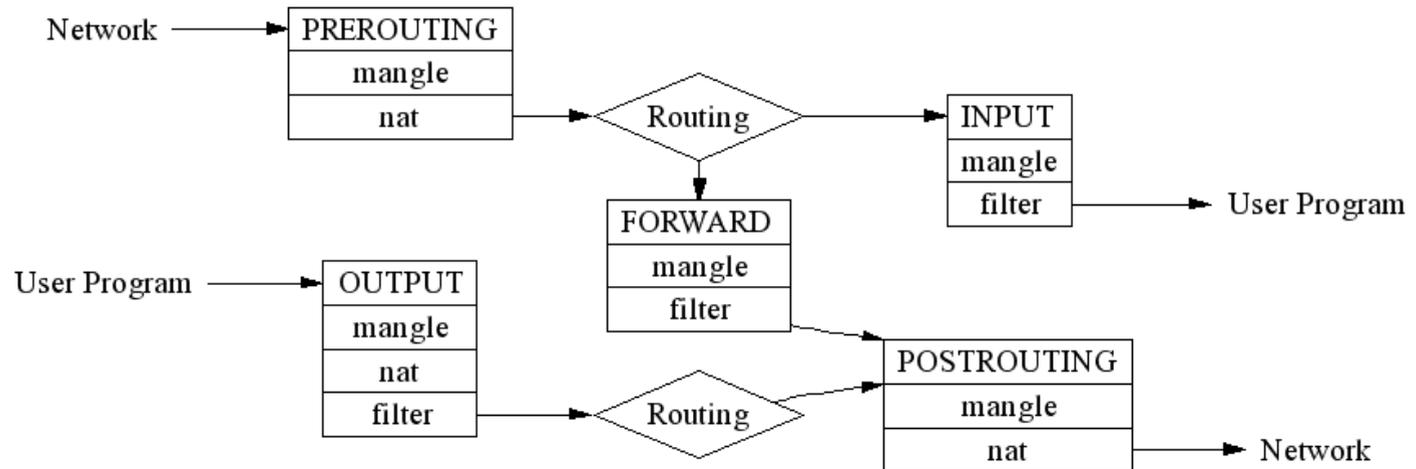
Listingcurrentrules

We use the `iptables` command to interact with the firewall (in the kernel):

```
$ sudo apt install iptables
```

```
$ sudo iptables -L -nv
```

```
Chain INPUT (policy ACCEPT 119 packets, 30860 bytes)
 pkts bytes target     prot opt in     out     source
  destination
```

```
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source
  destination
```

```
Chain OUTPUT (policy ACCEPT 36 packets, 1980 bytes)
 pkts bytes target     prot opt in     out     source
  destination
```

Yourfirstruleset

Configure your firewall to allow ICMP packets.

```
$ sudo iptables -A INPUT -p icmp -j ACCEPT

$ sudo iptables -L INPUT -nv

Chain INPUT (policy ACCEPT 4 packets, 520 bytes)

 pkts bytes target     prot opt in      out      source
  destination

    0     0 ACCEPT     icmp --  *       *        0.0.0.0/0
  0.0.0.0/0
```

What effect will this have?

What are the numbers?

Testingrules

How can you test it?

```
$ ping -c4 127.0.0.1

PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.

64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.058 ms

...

$ sudo iptables -L INPUT -nv

Chain INPUT (policy ACCEPT 220 packets, 218K bytes)
 pkts bytes target     prot opt in      out     source
  destination

    8   672 ACCEPT     icmp --  *       *       0.0.0.0/0
  0.0.0.0/0
```

Why do we see 8 packets against the rule, instead of 4?

You can use `iptables -L INPUT -nZ` to `Z`ero the counters.

Blockingpings

Add another rule:

```
$ sudo iptables -A INPUT -p icmp -j DROP
```

```
$ sudo iptables -L INPUT -nv
```

```
Chain INPUT (policy ACCEPT 12 packets, 1560 bytes)
```

```
 pkts bytes target     prot opt in      out      source
  destination
```

```
    8   672 ACCEPT     icmp --  *       *        0.0.0.0/0
 0.0.0.0/0
```

```
    0     0 DROP       icmp --  *       *        0.0.0.0/0
 0.0.0.0/0
```

```
$ ping -c1 127.0.0.1
```

```
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.067 ms
```

Is that what you expected?

Ruleprecedence

Insert a DROP rule **before** the ACCEPT rule with `-I` :

```
$ sudo iptables -I INPUT -p icmp -j DROP
```

```
$ sudo iptables -L INPUT -nv
```

```
Chain INPUT (policy ACCEPT 12 packets, 1560 bytes)
```

```
 pkts bytes target      prot opt in      out      source
  destination
```

```
   0     0 DROP        icmp --  *       *        0.0.0.0/0
  0.0.0.0/0
```

```
  10   840 ACCEPT      icmp --  *       *        0.0.0.0/0
  0.0.0.0/0
```

```
   0     0 DROP        icmp --  *       *        0.0.0.0/0
  0.0.0.0/0
```

Ruleprecedencetesting

```
$ ping -c1 127.0.0.1

PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.

^C

--- 127.0.0.1 ping statistics ---

1 packets transmitted, 0 received, 100% packet loss, time 0ms
```

Listruleswithindexes

Use the iptables `-L --line-numbers` options:

```
$ sudo iptables -L INPUT -nv --line-numbers

Chain INPUT (policy ACCEPT 15 packets, 1315 bytes)

num    pkts bytes target     prot opt in     out    source
  destination

1        0     0 DROP       icmp -- *      *      0.0.0.0/0
  0.0.0.0/0

2        0     0 ACCEPT     icmp -- *      *      0.0.0.0/0
  0.0.0.0/0

3        0     0 DROP       icmp -- *      *      0.0.0.0/0
  0.0.0.0/0
```

DeletingRules

Delete rule by index:

```
$ sudo iptables -D INPUT 3
```

Delete rule by target:

```
$ sudo iptables -D INPUT -p icmp -j ACCEPT
```

Check the results:

```
$ sudo iptables -L INPUT -nv --line-numbers

Chain INPUT (policy ACCEPT 9 packets, 835 bytes)

num    pkts bytes target     prot opt in      out      source
  destination

1       0     0 DROP         icmp --  *       *        0.0.0.0/0
  0.0.0.0/0
```

PersistentRules

What happens when you reboot?

PersistentRules

What happens when you reboot?

The rules that we created are only in the kernel's memory. They will be lost on reboot.

How can we make them permanent? Could be as simple as:

```
/sbin/iptables-save > /etc/default/iptables
```

```
/sbin/iptables-restore < /etc/default/iptables
```

Or install `iptables-persistent` which automates this a little.

ConnectionTracking

Every packet is tracked by default (made into a connection).

You can see them with `conntrack =L`:

```
sudo /usr/sbin/conntrack -L
```

```
tcp      6 431999 ESTABLISHED src=196.200.216.99
  dst=196.200.219.140 sport=58516 dport=22
```

```
src=196.200.219.140 dst=196.200.216.99 sport=22 dport=58516
  [ASSURED] mark=0 use=1
```

What does this mean?

ConnectionTracking

```
sudo /usr/sbin/conntrack -L
```

```
tcp      6 431999 ESTABLISHED src=196.200.216.99
  dst=196.200.219.140 sport=58516 dport=22
```

```
src=196.200.219.140 dst=196.200.216.99 sport=22 dport=58516
  [ASSURED] mark=0 use=1
```

- ESTABLISHED is the connection state
    - What are valid states?

- src=196.200.216.99 is the source address of the tracked connection

- dst=196.200.219.140 is the destination address
    - Which one is the address of this host? Will it always be?

- sport=58516: source port

- dport=22: destination port

- Another set of addresses: what is this?

ConnectionTracking

How do we use it?

- `iptables -A INPUT -m state --state ESTABLISHED -j ACCEPT`

    - You normally want this!

Can you see any problems?

ConnectionTrackingProblems

What happens if someone hits your server with this?

```
sudo hping3 --faster --rand-source -p 22 196.200.219.140 --syn
```

Or if you run a server that has thousands of clients?

ConnectionTrackingProblems

Add a rule to block all connection tracking to a particular port:

```
sudo /sbin/iptables -t raw -A PREROUTING -p tcp --dport 22 -j
  NOTRACK
```

Write your rules so that connection tracking is **not needed** (allow traffic both ways).

You probably want to do this for your DNS server. How?

ConnectionTrackingProblems

Add a rule to block all connection tracking to a particular port:

```
sudo /sbin/iptables -t raw -A PREROUTING -p tcp --dport 22 -j
  NOTRACK
```

Write your rules so that connection tracking is **not needed** (allow traffic both ways).

You probably want to do this for your DNS server. How?

```
sudo /sbin/iptables -t raw -A PREROUTING -p udp --dport 53 -j
  NOTRACK
```

Standardsimpleruleset

This is one of the first things I set up on any new box:

```
iptables -P INPUT ACCEPT
```

```
iptables -F INPUT
```

```
iptables -A INPUT -m state --state ESTABLISHED -j ACCEPT
```

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

```
iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
```

```
iptables -A INPUT -p tcp --dport 22 -j ACCEPT
```

```
iptables -A INPUT -m limit --limit 5/min -j LOG --log-prefix
  'Rejected INPUT '
```

Check that I can access the server without triggering a "Rejected INPUT"
message in the logs, and then lock it down:

```
iptables -P INPUT DROP
```

Exercise

Install `nmap`:

```
sudo apt install nmap
```

Scan your system:

```
sudo nmap -sS pcXX.sse.ws.afnog.org
```

- Which ports are open?

- How would you block them?

You will probably lock yourself out of your PC. That is OK, we can fix it :)

- As long as the changes have NOT been made permanent, we can reboot the system to restore access.

Exercise

The correct answer is:

```
iptables -I INPUT 2 -p tcp --dport 22 -j DROP
```

Which prevents new connections, but as long as rule 1 allows ESTABLISHED connections you will not be locked out (unless you lose your connection).

The output of `iptables -L -nv` should look like:

```
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target prot opt in out source      destination
  151 11173 ACCEPT all  --  *   *   0.0.0.0/0 0.0.0.0/0   state
  ESTABLISHED
    0     0         tcp  --  *   *   0.0.0.0/0 0.0.0.0/0   tcp
  dpt:22
```

FIN

Any questions?

(yeah, right!)