Firewalls
SS-E 2019

Kevin Chege
ISOC

What's a Firewall?

- Computer network security device to protect devices, or restrict access to or from a network
- Analyzes traffic coming in or going out (or through it) and determines a course of action based on a pre-defined rule set
- Firewalls can be found anywhere:
  - On your laptop OS
  - On routers
  - On server OS
  - On network hardware appliances

Types of firewalls

- Packet Filters – analyze network packets and decide a course of action based on configuration
- Stateful Filters – track network “conversations” and maintain a table of which connections are in an active conversations
- Application layer – aka Layer 7 firewalls are able to detect if an unwanted protocol is attempting to bypass the firewall on an allowed port

Keeping State vs Stateless

- Stateful inspection refers to ability to track the state, or progress, of a network connection
- By storing information about each connection in a state table, a firewall is able to quickly determine if a packet passing through the firewall belongs to an already established connection.
- If it does, it is passed through the firewall without going through ruleset evaluation saving time and avoiding extra processing.

Typical features of a Firewall

- Rule Syntax
- NAT control
- Able to pass, redirect or drop traffic based on the rules
- Logging feature – to allow audit of activities and of traffic
- Stateful inspection - not all and may need to be enabled with extra config options
- Ability to be either inclusive or exclusive - An exclusive firewall allows all traffic through except for the traffic matching the ruleset (default is to allow). Inclusive firewall does the reverse (default is to block)

FreeBSD Firewalls

- FreeBSD ships with 3 Main firewalls:
  - IPFW – IP FireWall is (by default) a stateless firewall. FreeBSD sponsored firewall software application authored and maintained by FreeBSD volunteer staff members.
  - IPF – IP Filter can be configured as stateful or stateless. Open source application and has been ported to FreeBSD, NetBSD, OpenBSD, SunOS™, HP/UX, and Solaris™ operating systems. IPFILTER is actively being supported and maintained, with updated versions being released regularly.
  - PF – Packet Filter can be configured as stateful or stateless. Maintained by OpenBSD Project

Linux IPTables

- **iptables** is a user-space utility program that allows a system administrator to configure the tables provided by the Linux kernel firewall which are implemented as different Netfilter modules
- Netfilter offers various functions and operations for packet filtering, network address translation, and port translation, which provide the functionality required for directing packets through a network and prohibiting packets from reaching sensitive locations within a network.

More on “iptables”

$ sudo apt-get install man

$ man iptables

What about default deny/permit?

- The recommended practice when setting up a firewall is to take a "default deny" approach.
- That is, to deny *everything* and then selectively allow certain traffic through the firewall.
- This approach is recommended because it errs on the side of caution and also makes writing a ruleset easier. the first two filter rules should be:
- **HOWEVER**, you may opt to approach your firewall rules differently depending on the scenario

Some iptables examples

sudo iptables -A INPUT -p icmp -j ACCEPT

- **-A** - Append one or more rules to the end of the selected chain
- **INPUT** - The filter table is the default table. It contains the actual firewall filtering rules. The built-in chains include these INPUT, OUTPUT, FORWARD
- **-p icmp**– Protocol (tcp, udp,, icmp, all, among others**)**
- **-j ACCEPT** – Jump -This specifies the target of the rule; i.e., what to do if the packet matches it: either ACCEPT or DROP

sudo iptables -A INPUT -p icmp -j ACCEPT

- **-A** - Append one or more rules to the end of the selected chain
- **INPUT** - The filter table is the default table. It contains the actual firewall filtering rules. The built-in chains include these INPUT, OUTPUT, FORWARD
- **-p icmp**– Protocol (tcp, udp,, icmp, all, among others**)**
- **-j ACCEPT** – Jump -This specifies the target of the rule; i.e., what to do if the packet matches it: either ***ACCEPT*** or DROP

sudo iptables -I INPUT -p icmp -j DROP

- **-I** - Inserts a rule at the beginning of the chain
- **INPUT** - The filter table is the default table. It contains the actual firewall filtering rules. The built-in chains include these INPUT, OUTPUT, FORWARD
- **-p icmp**– Protocol (tcp, udp,, icmp, all, among others**)**
- **-j DROP** – Jump -This specifies the target of the rule; i.e., what to do if the packet matches it: either ACCEPT or ***DROP***

Show the order of the rules

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

```
Chain INPUT (policy DROP 0 packets, 0 bytes)
num   pkts bytes target     prot opt in     out     source               destinatio
1        1    60 DROP       icmp --  *      *       0.0.0.0/0            0.0.0.0/0
2      207 12468 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0
3     4390  324K ACCEPT     all  --  *      *       0.0.0.0/0            0.0.0.0/0
```

Delete a rule

sudo iptables -D INPUT 1

```
afnog@pc35:~$ sudo iptables -D INPUT 1
afnog@pc35:~$ sudo iptables -L INPUT -nv --line-numbers
Chain INPUT (policy DROP 0 packets, 0 bytes)
num   pkts bytes target     prot opt in     out     source
1      207 12468 ACCEPT     icmp --  *      *       0.0.0.0/0
2     4491  332K ACCEPT     all  --  *      *       0.0.0.0/0
ate ESTABLISHED
```

FreeBSD Firewalls

- FreeBSD ships with 3 Main firewalls:
  - IPFW – IP FireWall is (by default) a stateless firewall. FreeBSD sponsored firewall software application authored and maintained by FreeBSD volunteer staff members.
  - IPF – IP Filter can be configured as stateful or stateless. Open source application and has been ported to FreeBSD, NetBSD, OpenBSD, SunOS™, HP/UX, and Solaris™ operating systems. IPFILTER is actively being supported and maintained, with updated versions being released regularly.
  - PF – Packet Filter can be configured as stateful or stateless. Maintained by OpenBSD Project

PF (Packet Filter)

- Was initially developed for OpenBSD
- Has been successfully ported to many other operating systems including all the other BSDs and Mac OS X
- Written by Daniel Hartmeier
- Derived its rule syntax from IPFilter
- Has many features

Features

- Can do both stateless or state-full firewalling
- Can do Network Address Translation
  - Additionally can do Bidirectional NAT aka One to One NAT
- Combined with ALTQ (ALTernate Queueing framework for BSD) can perform QoS
  - Priority queuing – assign certain traffic a higher priority than others before forwarding
  - Class Based Queuing – assigning bandwidth to certain queues and reducing bandwidth for others
- Can be configured for automatic fail-over between 2 boxes using CARP – Common Address Redundancy Protocol

Features cont'd

- FTP-proxy integration to handle FTP firewalling
- Configurable logging per rule to pflogd
  - Logs can be further monitored with tcpdump
- Simple IP Filter rule syntax
  - Eg: *pass in quick on em0 inet proto tcp all*
- Macro definition – to simplify rule creation
  - Eg identify an interface as "LAN" instead of "em0"
- Support for transparent proxying with SQUID
  - Redirect all traffic destined for a port 80 to the Squid port 8080 for Squid to process
- Among many others

Working with PF

- Installed by default on FreeBSD since FreeBSD 5.3 but is disabled
- Can start in from boot by adding the following to /etc/rc.conf: ***pf_enable=YES***
  - Or by ***kldload pf.ko***
- Start it by doing
  - ***/etc/rc.d/pf start*** OR ***pfctl –e***
- You may want to compile pf support into the kernel to enable:
  - Pfsync seudo device
  - CARP for automatic failover
  - ALTQ – for prioritization, bandwidth throttling

Options in rc.conf

- **pf_enable="YES**" # Enable PF (load module if required)
- **pf_rules="/etc/pf.conf"** # rules definition file for pf
- **pf_flags=""** # additional flags for pfctl startup
- **pflog_enable="YES"** # start pflogd(8)
- **pflog_logfile="/var/log/pflog"** # where pflogd should store the logfile
- **pflog_flags=""** # additional flags for pflogd startup
- You will also want to enable packet forwarding between interfaces and this can be done by
  - **gateway_enable="YES"** in /etc/rc.conf

Working with PF

- **pfctl –e** *Enable PF*
- **pfctl -d** *Disable PF*
- **pfctl -F all -f /etc/pf.conf** *Flush all rules (nat, filter, state, table, etc.) and reload*
- **pfctl -s [rules | nat | state]** *Report on the filter rules, nat rules, or state table*
- **pfctl -vnf /etc/pf.conf** *Check /etc/pf.conf for errors, but do not load ruleset*

Packet Filtering with PF

- Rules are loaded from a file usually **/etc/pf.conf**
- Packets can be passed, redirected or dropped as they pass through an interface
- PF inspects packets based on Layer 3 (IPv4/IPV6) and Layer 4 headers (TCP, UDP, ICMP/v6)
- Can check for source/destination address, protocol (Layer 4) and source/destination port
- Rules evaluated in sequential order – top to bottom of the file

Packet Filtering with PF cont'd

- A packet is evaluated against all the rules UNLESS the key word ***quick*** is specified
- If ***quick*** is not specified then the last rule to match wins and action is taken on the packet
- There is an implicit pass all at the beginning meaning that if a packet does not match any rule then it will be passed
- You are free to circumvent this feature if you want by having a "block all" at the top of the file

Rule Syntax

- *action [direction] [log] [quick] [on interface] [af] [proto protocol] [from src_addr [port src_port]] [to dst_addr [port dst_port]] [flags tcp_flags] [state]*
- **action** – pass or block
- **direction** – in or out
- **log** – should this be logged or not
- **quick** – specified action is taken immediately
- **on interface** – name of the interface
- **inet** – address family, inet6 for ipv6
- **protocol** – tcp, udp, icmp, icmp6 or others in **/etc/protocols**
- **src_addr/dst_addr** – source port or destination address
- **src_port/dst_port** – Number between 1 – 65535 (**/etc/services**
- **tcp_flags** – eg flags S/SA look only for SYN and ACK
- **state** – whether to check state. PF checks state by default

Good practice

- Recommended to have default deny at the beginning of the file so that what you do not specify is denied by default.
  - i.e. to make it an exclusive firewall
- This is to counter the default pass rule
- Done by adding the below at the top of the file
  - **block in all**
- Also good idea to leave out the loopback interface and link local addresses
  - **set skip on lo0**
  - You can set a macro eg: ipv6_ll="fe80::/10"

Some PF Examples

```
good_ports="{ 22, 443, 80 }"
me="192.168.0.1"
set skip on lo0
block in all
pass out all
pass in on em0 inet proto tcp from any to $me    port $good_ports
```

##This is sufficient to allow any communication that the server initiates (pass out all), allow all incoming tcp traffic to the good ports and block all other incoming traffic. The "pass out all" is needed despite PF having an implicit pass rule. Removing it will mean traffic out will not match any rule but incoming replies to conversations initiated by the server will be matched against the "block in all" rule.

References and more reading

- http://en.wikipedia.org/wiki/PF_%28firewall%29
- http://www.openbsd.org/faq/pf/filter.html
- http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/firewalls-pf.html
- http://en.wikipedia.org/wiki/Firewall_%28computing%29
- http://www.informit.com/articles/article.aspx?p=421057&seqNum=4