How Packet Firewall (PF) Can Protect Your Enterprise

Hello all,
Welcome to the talk about the PF firewall.

How many people today have used or heard of pf?

For those who have I hope you enjoy the talk and maybe it will talk about something you heard or seen before.

For those who have I hope it provides the framework for you to implement pf for your clients or enterprise.

Who am I

Who am I?

Jason Miller
Father
IT Nerd
Big Bang Theory Enthusiast
Sports ball enthusiast
Endpoint Engineer @ Lawrence
Berkeley National Lab
Board Member of **Macbrained**
jason@jasonkmiller.com
@jasonkmiller

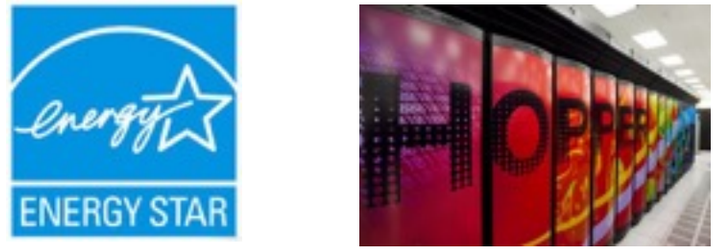
macbrained.org

Mac and iOS-focused community for developers, engineers and administrators-alike

Who is Jason?

Lawrence Berkeley National Lab?

Some people may be wondering where and what this is?



ALS - Electron bunches traveling at nearly the speed of light, when forced into a circular path by magnets, emit bright ultraviolet and x-ray light that shines down beamlines to experiment endstations. ALS produces light in the x-ray region of electromagnetic spectrum that is one billion times brighter than the sun.
Energy Star was created in 1992.
JGI - Help Develop and define the human Genome with other National Labs.
Nersc - Home to Cray XC30 supercomputer, Edison and Crazy XE6 supercomputer Hopper. Currently working on Cori system which is suppose to deliver 10x the computing capacity of the Hopper supercomputer.
ESNET - High-performance, unclassified national network built to support scientific research. Manage backbone for over 140 research and commercial networks.

Why are we here?

PF, Packet Filter On OS X

Brief History

Why Use PF!

What is PF?

Additionally PF Can Do?

Build Some Rules

Brief History - Originally developed by Daniel Hartmeir and now maintained and developed by OpenBSD Team.
Why use PF! - Filtering TCP/IP Traffic to clients and server and/or Network Address Translation (NAT).
What is PF? - Openbsd Packet Firewall, that is included with OS X. Have to use pf because ipfw (internet packet firewall) was deprecated in 10.9. Command line based tool, however, there are GUI based options.
Additionally PF can

- Provide Bandwidth Control
- Packet Prioritization

Use Case

server/client protection

- From Outside World
- From Internal Clients
- From Internal Threats
- Understand who and what is trying to talk to your clients.

Think of one your users using public wifi in a coffee shop
- HR
- Executives
- Admins

Explain the reason why you started using PF Firewall:
- Outside World
- Internal clients
- Internal threats
- Seeing that we are a national laboratory, users from all over the world attempt to access our systems. With pf can log the attempts and IPs.
- Infosec Mandate to help secure a server that had control and access to hundreds of clients

general use cases

- Servers
- Workstations
  - Desktops
  - Laptops
- Gateway for Windows XP based clients

Let’s Break it down

Let’s break it down

pf background

the basics

pfctl

- **pfctl**

Basic Flags

Enable	Disable	Load the pf.conf file	Parse but don't load
-e	-d	-f	-nf

Current Rulset	Current State Table	Filter Stats & Counters	All of the things
-sr	-ss	-si	-sa

Running PF on the command line will not work. pf is controlled by pfctl.
Man pfctl to grab additional resources

pf files

```
9:09 jmiller@hueyfreeman /Users/jmiller
% ls /etc | grep pf
pf.anchors
pf.conf
pf.os
```

It is located /etc/ has three files:

- pf.conf - 5 parts
  - Macros - User defined, Hold IP Address, interface names
  - Tables - Used to hold a list of IP addresses
  - Options - Flags to control PF
  - Queuing - Bandwidth control and packet prioritization
  - Filter Rules - User defined rules about packets allowed to talk over selected interfaces
  - This is the file that is being evaluated when applying rules
- pf.anchors - Sub rulesets that can be modified on the fly using pfctl. It is attached to the main ruleset.
  - anchor is collection of rules, tables, and potentially other anchors. Processing continues in the pf.conf(main ruleset) unless the packet matches a filter rule, that uses the quick option.
- pf.os - fingerprint file, determines the remote operating system by comparing TCP SYN packet.

pf background

the basics

Create Ruleset

What is a Ruleset?

It is collection of tables, lists, and macros to help you manage inbound and outbound packets.

By default apple has excepts its exceptions list included in the pf rules included inside of pf.anchors com.apple

pf ruleset makeup

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 - action that should be taken for matching packets, either pass or blocks.
direction - direction the packet is moving on the interface in or out
log - where the packet should be logged
quick - Rule is considered last matching rule and the specified action is taken.
interface - name/group of network interface the packet is moving through.
af - address family, IPv4 or IPv6, which is designated by inet or inet6
protocol - Layer 4 protocol of the packet

- tcp
- udp
- icmp
- icmp6
- A valid protocol name from /etc/protocols
- A protocol number between 0 and 255
- A set of protocols using a list

tcp_flags

state

- no state
- keep state
- modulate state
- synproxy state

Let’s talk about lists,
macros, and tables
oh my!

lists & macros

- Create and maintain Rulesets **EASIER!**
- Macros can be a single IP, or Hostname
  - “{“ helps to determine that a list is within a macro
  - “$” shows you a Macro is to be expanded
- List and Macros are the same but different

Lists: Allows for the specification of multiple similar criteria within a rule. Contain nested lists.

- Using for list of IP addresses

Macros: User-defined variables:

- IP address port numbers, interfaces names
- Names must start with letter & may contain letters, digits, and underscores.
- Cannot be pass, out or queue, which we will discuss later.
- Using Macro’s is great for network interfaces.

Difference between List and a Macro is that a list cannot contain a Macro but a Macro can contain a list.

lists & macros

```
block in log on en0  inet from 127.0.0.0
block in log on en0  inet from 192.168.0.0
block in log on en0  inet from 172.16.0.0
```

Sample List

```
block in quick log on en0 from { 127.0.0.0.
192.168.0.0, 172.16.0.0} to any
```

Sample Macro

```
ext_if = "fxp0"
```

Sample Macro that contains a list

```
icmp_types = "{ echorep, echoreq, timex, unreach }"
```

Break down this slide.

- block
- on
- log
- on
- en0
- inet
- from
- IPs

Let’s Talk about Tables

tables

table <block_hosts> { 192.0.2.0/24 }
table <my_infrastructure> const { 192.168.0.0/16,
172.16.0.0/12, \
  10.0.0.0/8 }
table <> persist

block in on fxp0 from { <rfc1918>, <spammers> } to any
pass in on fxp0 from <goodguys> to any

Manipulate Tables on the fly
pfctl -t <table name> -T add 123.56.78.91/16

Tables are a method to help group together a range of IP address.
Lists of IP addresses which can be manipulated without needing to reload the entire rule set, and where fast lookups are desirable.
Tables are also enclosed by <>
const - contents of the table cannot be changed once the table is created.
persist - causes the kernel to keep the table in memory even when no rules refer to it. If your table doesn’t have this option, the kernel will automatically remove the table when the last rule referencing it is flushed.

the rules

```
#allow out the tcp and udp traffic
pass in log proto tcp from <mpsg_host> to port $tcp_services
pass in log proto udp from <mpsg_host> to port $udp_services
#pass in lo                                                    l_bigfix_udp
pass in log #Trust all outbound                               .ssh
pass in log                                                   .ssh
pass in log pass out all keep state                           .ssh
pass in log proto udp from <mpsg_host> to port $casper_ssh
pass in log proto tcp from any to port $casper_comms
pass in log proto tcp from <mpsg_host> to port $casper_filerep
```

- Disable Filtering
- Default blocking in all
- *If necessary block out all

Inbound Rules
Outbound Rules
Ruleset Allowing tcp and udp traffic

Loopback feedback and vmnet work traffic we will not monitor
Explain:
- Block
- in
- log
- all

Explain:
- pass
- in
- log
- quick
- proto
- tcp/udp
- list
- macros

TEST!

Second to Last Step of Ruleset

pf files

```
scrub-anchor "com.apple/*"
nat-anchor "com.apple/*"
rdr-anchor "com.apple/*"
dummynet-anchor "com.apple/*"
anchor "com.apple/*"
load anchor "com.apple" from "/etc/pf.anchors/com.apple"
```

Edit the /etc/pf.conf file to add in your custom anchor.

- scrub-anchor “”
- anchor
- load anchor

Last step

test

pf background

the basics

Create Ruleset

Enable Logging

Logging is a fun story!

Maybe not these logs

What about the logs?

pf logging

No Logging
(default)

Create your own logging
- tcpdump
- permissions
- create launchdaemon

By default pf has no logging mechanism included. You must piece together the monitoring, logging, and launchdaemon in order to monitor your systems because you want to have a text log of things getting blocked, otherwise how do you know its working?

SO how is this done!

pf log setup

enable syslogging of local2 to /var/log/pf.log

```
echo -e "# gather PF log data\nlocal2.*\t\t\t/private/var/log/pf.log" >> /etc/syslog.conf
z touch /private/var/log/pf.log
chmod 640 /private/var/log/pf.log
chown root:wheel /private/var/log/pf.log
killall -HUP syslogd
```

1. Enable syslogging of your device to pf.log
2. Set owner & permissions
3. Setup a tcpdump of pflog0 to syslog
    1. Set owner to root
    2. chmod on
4. Create a Launchdaemon to ensure pf turns on at boot
5. Edit syslog.conf to ensure local2 is actually pointing toward pf.log (trust but verify)
6. Changes pfctl to start fully enabled versus on demand
7. Load the plist using launchctl

Use this section if pflogging seems to not be working on devices. Found that the first step of echo -e sometimes didn't add in local2 into syslog file.

#Uses logger, the interface to syslogd, needing an update to its config file

edit /etc/syslog.conf, add this line to the end of the file
local2.* /var/log/pf.log

#last step - switches the pfctl launch daemon to start fully enabled (rather than enabled on demand. Added the -e flag to enable pf. edit /System/Library/LaunchDaemons/com.apple.pfctl.plist

```
 <key>ProgramArguments</key>
      <array>
            <string>pfctl</string>
            <string>-ef</string>
            <string>/etc/pf.conf</string>
**********************************************************************
```

launchctl list l grep pf
launchctl load -w /Library/LaunchDaemons/nameof.plist
launchctl list l grep pf # should see pflog

pf log setup

```
Create a Launch Control daemon to start at boot and ensure it is running

cat >/Library/LaunchDaemons/nameof.plist <<END
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/
DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
        <key>Label</key>                <string>pflog</string>
        <key>ProgramArguments</key>
                <array>
                        <string>/usr/local/bin/pflog.sh</string>
                </array>
        <key>Disabled</key>             <false/>
        <key>RunAtLoad</key>            <true/>
        <key>KeepAlive</key>            <true/>
</dict>
</plist>
END
        chown root:wheel /Library/LaunchDaemons/nameof.plist
        chmod 444 /Library/LaunchDaemons/nameof.plist
```

1. Create a Launchdaemon to ensure pf turns on at boot
2. Edit syslog.conf to ensure local2 is actually pointing toward pf.log (trust but verify)
3. Changes pfctl to start fully enabled versus on demand
4. Load the plist using launchctl

Use this section if pflogging seems to not be working on devices. Found that the first step of echo -e sometimes didn't add in local2 into syslog file.

#Uses logger, the interface to syslogd, needing an update to its config file

edit /etc/syslog.conf, add this line to the end of the file
local2.* /var/log/pf.log

#last step - switches the pfctl launch daemon to start fully enabled (rather than enabled on demand. Added the -e flag to enable pf. edit /System/Library/LaunchDaemons/com.apple.pfctl.plist

```
 <key>ProgramArguments</key>
      <array>
            <string>pfctl</string>
            <string>-ef</string>
            <string>/etc/pf.conf</string>
**********************************************************************
```

launchctl list l grep pf
launchctl load -w /Library/LaunchDaemons/nameof.plist
launchctl list l grep pf # should see pflog

pf background

the basics

Create Ruleset

Enable Logging

Monitor

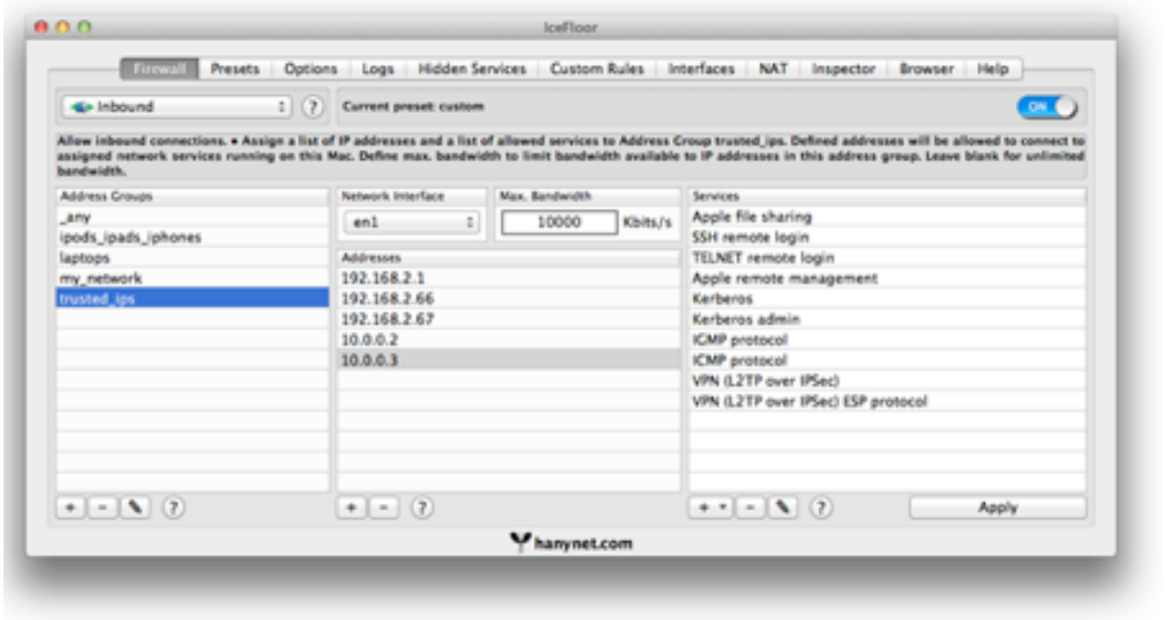
Use any tool that you use to aggregate logs:

- Splunk
- logstash
- scripting

After monitor phase, so what happened to the GUI!

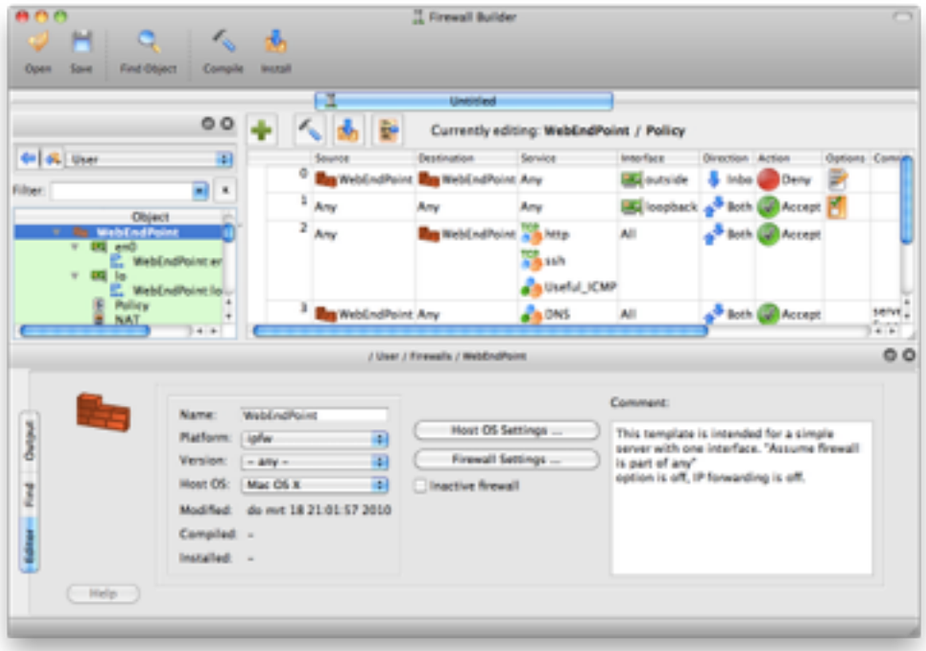
No love the GUI

Icefloor



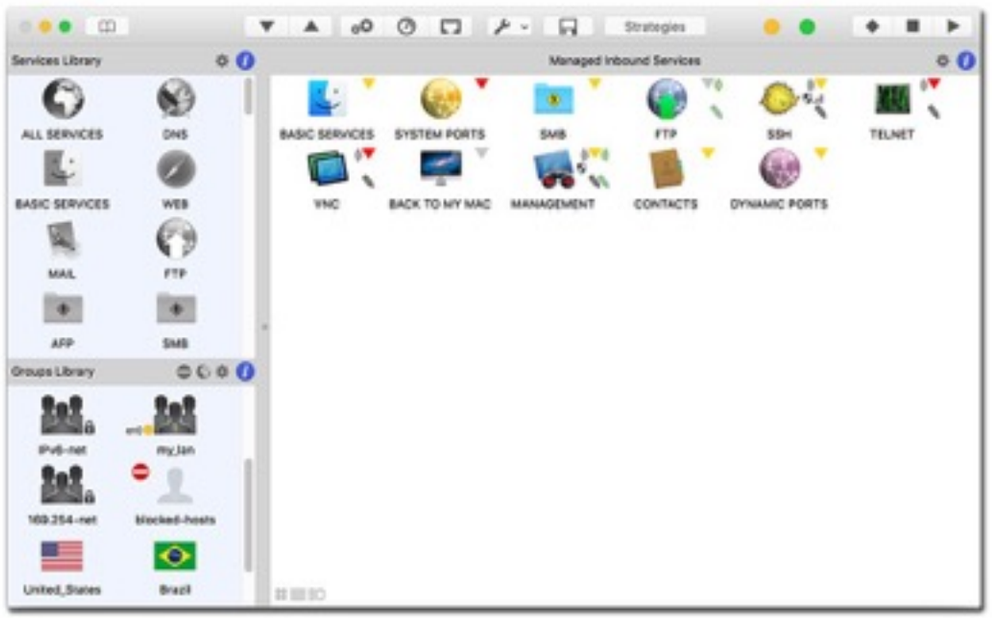
www.hanynet.com/icefloor

Firewall Builder



www.fwbuilder.org

Murus



www.murusfirewall.com

What about Bandwidth Control & Packet Prioritization?

PF does offer this functionality however this is something that I did not configure or play with just yet.

Resources

- http://krypted.com/tag/pf/
- http://www.jasonkmiller.com/search?q=pf
- https://www.nostarch.com/pf3
- http://www.openbsd.org/faq/pf/
- ftp://ftp.openbsd.org/pub/OpenBSD/doc/pf-faq.txt
- https://github.com/millerjasonkyle/osx/blob/master/pf_logging_setup.txt
- https://developer.apple.com/library/mac/documentation/Darwin/Reference/ManPages/man5/pf.conf.5.html
- https://developer.apple.com/library/mac/documentation/Darwin/Reference/ManPages/man8/pfctl.8.html

Go over resources

End closing I want to remind everyone of one thing. Click the button

“Networks are hard. People are soft”

–Taylor Swift

Thank you!


How?
Where?
Which?
When?
What?
Why?
Who?
Where