Packet Filtering: Mac OSX Under the Hood with Apple PF

EVERYTHING IS AWESOME!


EMMET

Johnny Himes
IT Consultant @ PSU
Jxh864@psu.edu
@jhimes830
jhimes830/Slack MacAdmins

Enable PF

- **PF is enabled/disabled through the pfctl command.**
- **# pfctl -e**
- **# pfctl -d**

Rules Location

- /etc/PF.conf
- /etc/PF.os
- etc/PF.anchors

```
# Default PF configuration file.
#
# This file contains the main ruleset, which gets automatically loaded
# at startup.  PF will not be automatically enabled, however.  Instead,
# each component which utilizes PF is responsible for enabling and disabling
# PF via -E and -X as documented in pfctl(8).  That will ensure that PF
# is disabled only when the last enable reference is released.
#
# Care must be taken to ensure that the main ruleset does not get flushed,
# as the nested anchors rely on the anchor point defined here. In addition,
# to the anchors loaded by this file, some system services would dynamically
# insert anchors into the main ruleset. These anchors will be added only when
# the system service is used and would removed on termination of the service.
#
# See pf.conf(5) for syntax.
#

#
# com.apple anchor point
#
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"
```

Basic Commands

pfctl -f /etc/pf.conf

pfctl -nf /etc/pf.conf

pfctl -Nf /etc/pf.conf

pfctl -Rf /etc/pf.conf

pfctl -sn

pfctl -ss

pfctl -si

pfctl -sa

7 Types of statement in PF.conf

- **Macros**
- **Tables**
- **Options**
- **Traffic Normalization**
- **Queueing**
- **Translation**
- **Packet Filtering**

PF - Packet Filtering

General 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]

Default Deny

To create a default deny filter policy, the first two filter rules should be:

block in all

block out all

Pass Through Examples

```
# Pass traffic in on dc0 from the local network, 192.168.0.0/24,
# to the OpenBSD machine's IP address 192.168.0.1. Also,
pass the
# return traffic out on dc0.
pass in on dc0 from 192.168.0.0/24 to 192.168.0.1
pass out on dc0 from 192.168.0.1 to 192.168.0.0/24
# Pass TCP traffic in on fxp0 to the web server running on the
# OpenBSD machine. The interface name, fxp0, is used as the
# destination address so that packets will only match this rule
if
# they're destined for the OpenBSD machine.
pass in on fxp0 proto tcp from any to fxp0 port www
```

“Last Matching” Rule

Wrong:

block in on en0 proto tcp from any to any port ssh

pass in all

Better:

block in quick on en0 proto tcp from any to any port ssh

pass in all

Keeping State

- **TCP**
- **state table**
- **Improve performance**
- **dramatically faster**
- **UDP - Will keep track of how long a matching packet has gone through but has no true start/ end of packet.**

State Tacking Options

```
table <abusive_hosts> persist

block in quick from <abusive_hosts>

pass in on $ext_if proto tcp to $web_server \

port www flags S/SA keep state \

(max-src-conn 100, max-src-conn-rate 15/5,
overload <abusive_hosts>

flush)
```

Blocking Spoofed Packets

antispoof [log] [quick] for interface [af]

antispoof for fxp0 net

block in on ! fxp0 inet from 10.0.0.0/24 to any

block in inet from 10.0.0.1 to any

Lists

Correct way:

block out on en0 from { 192.168.0.1, 10.5.32.6 } to any

block out on fxp0 from 192.168.0.1 to any

block out on fxp0 from 10.5.32.6 to any

Wrong Way:

pass in on fxp0 from { 10.0.0.0/8, !10.1.2.3 }

pass in on fxp0 from 10.0.0.0/8

pass in on fxp0 from !10.1.2.3

Tables

```
table <goodguys> { 192.0.2.0/24 }

table <rfc1918> const { 192.168.0.0/16, 172.16.0.0/12, \

10.0.0.0/8 }

table <spammers> persist

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

pass in on fxp0 from <goodguys> to any

table <spammers> persist file "/etc/spammers"

block in on fxp0 from <spammers> to any
```

Add to Tables

```
# pfctl -t spammers -T add 218.70.0.0/16

# pfctl -t spammers -T show

# pfctl -t spammers -T delete 218.70.0.0/16
```

Quick Rulesets

- **Macros:**

 # define macros for each network interface

IntIF = "dc0"

ExtIF = "fxp0"

DmzIF = “fxp1"

 # define our networks

IntNet = "192.168.0.0/24"

ExtAdd = "24.65.13.4"

DmzNet = "10.0.0.0/24"

```
pre = "pass in quick on ep0 inet proto tcp from "
post = "to any port { 80, 6667 } keep state”

# David's classroom
$pre 21.14.24.80 $post

# Nick's home
$pre 24.2.74.79 $post
$pre 24.2.74.178 $post

Expands to:

pass in quick on ep0 inet proto tcp from 21.14.24.80 to any \
port = 80 keep state
pass in quick on ep0 inet proto tcp from 21.14.24.80 to any \
port = 6667 keep state
pass in quick on ep0 inet proto tcp from 24.2.74.79 to any \
port = 80 keep state
pass in quick on ep0 inet proto tcp from 24.2.74.79 to any \
port = 6667 keep state
pass in quick on ep0 inet proto tcp from 24.2.74.178 to any \
port = 80 keep state
pass in quick on ep0 inet proto tcp from 24.2.74.178 to any \
port = 6667 keep state
```

Anchors - “Sub Rulesets”

*** anchor name - evaluates all filter rules in the anchor name**

*** binat-anchor name - evaluates all binat rules in the anchor name**

*** nat-anchor name - evaluates all nat rules in the anchor name**

*** rdr-anchor name - evaluates all rdr rules in the anchor name**

Anchor Example

ext_if = "fxp0"

block on $ext_if all

pass out on $ext_if all keep state

anchor goodguys

Anchor Example

anchor goodguys

load anchor goodguys from “/etc/anchor-goodguys-ssh”

```
# echo "pass in proto tcp from 192.0.2.3 to any port 22" \

| pfctl -a goodguys -f -

anchor "goodguys" {

pass in proto tcp from 192.168.2.3 to port 22

}
```

Manipulate Anchors

List all rules

 # pfctl -a ssh -s rules

Flush all rules

 # pfctl -a ssh -F rules

PF Logging

pass in log (all, to pflog1) on $ext_if inet proto tcp to $ext_if port 22

keep state

View log:

tcpdump -n -e -ttt -r /var/log/pflog0

Reading logs

To view the log file:

 # tcpdump -n -e -ttt -r /var/log/pflog

Real Time View:

 # tcpdump -n -e -ttt -i pflog0

Ice Floor

IceFloor

hanynet.com proudly presents the first PF firewall frontend for OS X. IceFloor 2 is group based, like the old ServerAdmin firewall tool. Control filtering, bandwidth, logs, connections and custom PF configurations.

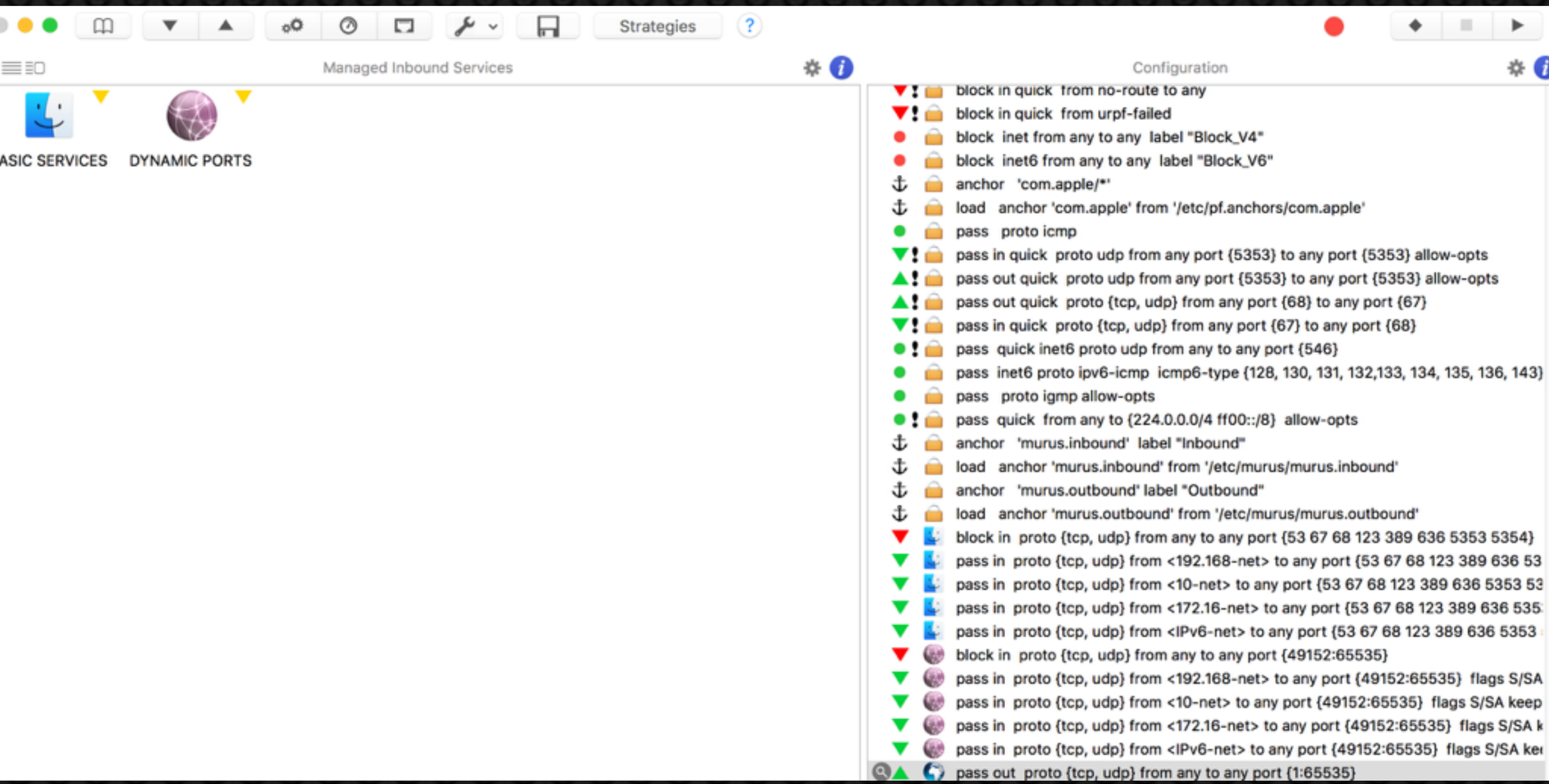
Murus, the new OS X Yosemite PF firewall front end is now available!

Murus Lite is the entry level version of Murus, a totally new PF front end for OS X 10.9 and 10.10 Yosemite. It features a great interface and a lot of cool features. Murus Lite is FREE !!! Download Murus Lite here!

- IceFloor 2 is group based. Create groups and assign addresses, services and parameters to pass or block connections
- IceFloor uses its own set of PF configuration files; default OS X PF configuration files are not modified
- start with IceFloor Wizard to create a basic PF configuration in a few mouse clicks
- use IceFloor interface to set up very complex and customized PF rulesets
- manage inbound and outbound connections with filtering and bandwidth rules for your Mac and NAT clients
- hide services using port knocking, list and block connections on the fly using Inspector
- create custom PF presets including custom rules, options, filtering and bandwidth rules
- mix IceFloor PF rules with your custom PF rules, interact with external applications like sshguard
- share Internet connection using PF NAT, assign per-client filtering and bandwidth rules and redirections
- browse PF ruleset with the new PF Rules Browser, display filtering, bandwidth and NAT PF rules and pipes
- analyze PF logs with numerical and graphical statistics
- debug and test PF rulesets easily and quickly using IceFloor Menulet
- IceFloor is free and open source. It requires OS X 10.7. Some feature is available only on OS X 10.8 and 10.9. Bandwidth management and other features are not available on OS X 10.10 Yosemite.

Murus Lite

https://www.murusfirewall.com/



My setup

- **/etc/psuiss.sh**
- **com.apple.psuiss.plist**
- **Library/psuiss/pf**

Handy commands

sudo launchctl unload /Library/LaunchDaemons/com.apple.psuiss.plist

sudo launchctl load /Library/LaunchDaemons/com.apple.psuiss.plist

sudo pfctl -a psuiss.inbound -s rules

networksetup -listallhardwareports

Helpful Links

- https://gist.github.com/tracphil/4353170
- https://www.freebsd.org/cgi/man.cgi?query=pfctl&sektion=8
- http://krypted.com/mac-security/a-cheat-sheet-for-using-pf-in-os-x-lion-and-up/
- http://www.hanynet.com/icefloor/
- https://www.murusfirewall.com/

GitHub repo

- https://github.com/jhimes/PF-setup

- This will allow you to setup PF the way I did in my environment.

Thank you for being an “AWESOME” group!

Johnny Himes
IT Consultant @ PSU
Jxh864@psu.edu
@jhimes830
jhimes830/Slack MacAdmins

Feedback: bit.ly/psumac2017-148