Fail2Ban

"…FAILURE IS DELAY, NOT DEFEAT…" – DENIS WAITLEY

Agenda

- What is fail2ban?
- My story
- And then there was DoS
- A look at fail2ban
- Summary

Intrusion Detection vs Prevention

What is fail2ban?

Fail2Ban is an intrusion prevention software framework that protects computer servers from brute-force attacks.

Fail2Ban scans log files and bans IP addresses of hosts that have too many failures within a specified time window.

Think of it as a dynamic firewall. It detects incoming connection failures, and dynamically adds a firewall rule to block that host after too many failures.

My Story

- On my Linux servers, I do not allow username/password authentication
- Users must use SSH with PKI
- But I still didn't like the barrage of remote login attempts
- *My fear was an unknown zero-day, race condition, buffer overflow or other vulnerability was still a threat*
- So I looked for intrusion detection and prevention software
- I found, installed, learned and started using fail2ban to block unwanted ssh connection attempts
- **I was right**. *We fell victim to a previously unknown Denial of Service vulnerability*

Under Attack

- In May of 2016, we suffered a SLOW denial of service attack
- Something was causing our web site to hang every 5-15 minutes
- Restarting Apache would fix the problem, but the site would just hang again in 5-15 minutes
- We did not have an unusually high volume of HTTP GET/POSTs
- We had what seemed like an unusually high amount of Baidu spider traffic

Baidu Spider

- Baidu ignores your robot.txt file, and they do whatever the h*ll they want
- Baidu was 60% of all our bot traffic, 50% more than all the others combined
- Baidu connections primarily come from180.76.15.*, but switches to other IP ranges if not having any success with that IP range
- "Chinese search engines such as Baidu … will merrily spider your sites to oblivion if you let them" - https://searchenginewatch.com/sew/news/2067357/bye-bye-crawler-blocking-parasites
- I recommend you block Baidu

Attack Investigation

- Blocking Baidu traffic did not stop the hanging
- When the site was hung
  - All **ServerLimit** httpd daemons had been allocated
  - None of the httpd daemons were consuming any CPU time
  - All httpd daemons were in the **flock_lock_file_wait** state
- We finally noticed an unusual HTTP GET request
  - It was a request to our shopping cart
  - They were all "delete" type requests
  - It was supposedly from a googlebot
- Why is a bot sending random delete requests to our shopping cart?
- Blocking the 3 IP addresses used by these unusual requests stopped the hanging!

Example requests

- 64.150.181.58 - - [13/May/2016:11:46:29 -0500] "GET /checkout/cart/**delete**/id/14816/uenc/aHR0cDovL3d3dy5uYXRpb25hbGN5Y2xlLmNvbS9jYXRhbG9nL3Byb2R1Y3Qvdmlldy9pZC82MzMv/ HTTP/1.1" 200 1724015 "http://www.domain.com/catalog/product/view/id/633/" "Mozilla/5.0 (compatible; **Googlebot**/2.1; +http://www.google.com/bot.html)"

- 69.64.95.112 - - [13/May/2016:12:19:09 -0500] "GET /checkout/cart/**delete**/id/14835/uenc/aHR0cDovL3d3dy5uYXRpb25hbGN5Y2xlLmNvbS9uMTM1MS5odG1s/ HTTP/1.1" 200 1688239 "http://www.domain.com/n1351.html" "Mozilla/5.0 (compatible; **Googlebot**/2.1; +http://www.google.com/bot.html)"

Attack Forensics

- Apache logs indicated they were Googlebot requests; but they were sending a bogus **User-Agent** string
- The IP address of the "Googlebot" request mapped back to **bluechipbacklinks.com**
- Blue Chip Back Links is a shady outfit that sells you expired domains to create SEO PBNs (Private Blog Networks). They are used to create backlinks to a website to increase Google page ranking
- Each of these HTTP GET requests would HANG one httpd daemon forever by putting it into a **flock_lock_file_wait** state
- We were getting roughly one of these DoS HTTP requests every 10-20 seconds
- Very difficult to:
  - Identify why so many httpd daemons were getting allocated
  - Realize that httpd daemons were running but hung
  - Figure out a way to show which/if HTTP daemons were hung
  - Finally what was causing them to hang

Diverting Attack

- Manually blocked 3 IP addresses with iptables
- Created a fail2ban filter to identify and block these unusual HTTP requests
- Remove manual iptable entries
- Monitor fail2ban and iptables
- Review system logs for this and other persistent threats that needed to be blocked

Let's Look at fail2ban

Features

- client/server
- multi-threaded
- autodetection of datetime format
- lots of predefined support
  - services – sshd, apache, qmail, proftpd, sasl, asterisk, squid, vsftpd, assp, etc
  - actions – iptables, tcp-wrapper, shorewall, sendmail, ipfw, etc

Requirements

- Python >= 2.4
- Optional
  - iptables
  - shorewall
  - tcp_wrappers
  - mail
  - gamin

Limitations

- Reaction time – fail2ban is a log parser, so it cannot do anything before something is written to the log file.
- Syslog daemons normally buffer output, so you may want to disable buffering in your syslog daemon
- fail2ban waits 1 second before checking log files for changes, so it's possible to get more failures than specified by maxretry
- A local user could initiate a DoS attack by forging syslog entries with the logger(1) command
- ~~The pattern or regex to match the time stamp is currently not documented, and not available for users to read or set. This is a problem if your log has a timestamp format that fail2ban doesn't expect, since it will then fail to match any lines~~

Components

Directories

- /etc/fail2ban/action.d
- /etc/fail2ban/fail2ban.d
- /etc/fail2ban/filter.d
- /etc/fail2ban/jail.d

Commands

- fail2ban-server
- fail2ban-client
- fail2ban-regex

Files

- /etc/fail2ban/fail2ban.conf
- /etc/fail2ban/fail2ban.local
- /etc/fail2ban/jail.conf
- /etc/fail2ban/jail.local

Configuration Files

Global Configuration Files

- **fail2ban.conf**

Main configuration options. File should not be modified, customizations are done in fail2ban.local

- **jail.conf**

Declaration(s) of jails that define a combination of Filters and Actions

Local Customizations

- **fail2ban.local**

Settings you would like to override in fail2ban.conf. The .conf file is parsed first and then .local settings are applied

- **jail.local**

New or custom jails to override default jail.conf declarations

Configuration Order

- fail2ban.conf
- fail2ban.d/*.conf (alphabetical)
- fail2ban.local
- fail2ban.d/*.local (alphabetical)

- jail.conf
- jail.d/*.conf (alphabetical)
- jail.local
- jail.d/*.local (alphabetical)

Terminology

fail2ban

Software that bans & unbans IP addresses after a defined number of failures

(un)ban

(Remove)/Add a firewall rule to (un)block an IP address

jail

A jail is the definition of one fail2ban-server thread that watches one or more log file(s), using one filter and can perform one or more actions

filter

Regular expression(s) applied to entries in the jail's log file(s) trying to find pattern matches identifying brute-force break-in attempts

action

One or more commands executed when the outcome of the filter process is true AND the criteria in the fail2ban and jail configuration files are satisfied to perform a ban

fail2ban-server

- A Python program that is
  - multi-threaded
  - listens on Unix sockets for commands
- The server
  - reads log file(s) defined in jails
  - applies a filter defined for the jail and found in filter.d
  - analyzes them using failregex defined in the the filter
  - executes actions defined in actions.d

fail2ban-client

- A command line utility to configure and control the fail2ban-server
  - status [JAIL]
  - start/stop (all jails)
  - start/stop [JAIL]
  - reload [JAIL]
  - ping
  - set/get

Useful commands

Show list of jails

```
# fail2ban-client status
Status
|- Number of jail:      6
`- Jail list:   apache-auth, block-spider, magento-checkout, my-sshd, wp-attacks, wp-
login-attack
```

Status of specific jail

```
# fail2ban-client status my-sshd
Status for the jail: my-sshd
|- Filter
|  |- Currently failed: 23
|  |- Total failed:     7519
|  `- File list:        /var/log/secure
`- Actions
   |- Currently banned: 25
   |- Total banned:     1906
   `- Banned IP list:   200.72.2.200 178.33.189.220 181.49.211.34 212.131.189.111
27.120.94.9 63.247.85.18 185.93.185.239 190.4.63.56 163.172.209.37 221.210.200.245
221.194.47.208 200.216.31.244 221.194.47.249 37.187.137.141 190.181.39.\
15 121.18.238.114 185.78.29.33 119.249.54.88 110.45.144.55 119.249.54.75 71.183.108.45
200.216.31.20 119.249.54.68 181.143.226.67 198.245.49.221
```

Useful commands

List ACTIONS defined for a JAIL

```
# fail2ban-client get wp-attacks actions
The jail wp-attacks has the following actions:
iptables-multiport
```

UNBAN an IP

```
# fail2ban-client set my-sshd unbanip 200.72.2.200
200.72.2.200
```

BAN an IP

```
# fail2ban-client set my-sshd banip 200.72.2.200
200.72.2.200
```

fail2ban-regex

- A command line utility to:
  - Test date format matching
  - Develop and test new "Failregex" strings
  - Develop and test new "ignoreregex" strings
  - Check if your regular expression(s) are parsing log file for lines or files that identify brute-force break-in/attack attempts
  - Test fail2ban filter files on log files
  - Use to expand hierarchical shortcuts

fail2ban-regex testing

Synopsis

fail2ban-regex [options] LOG REGEX [ignoreregex]

Example using command line strings for LOG and REGEX

```
fail2ban-regex 'Oct  9 05:28:52 magento sshd[1304]: Invalid user km999 from 52.208.45.232'
               '^.*sshd\[\d*\]: Invalid user .* from <HOST>$'
Running tests
=============

Use   failregex line : ^.*sshd\[\d*\]: Invalid user .* from <HOST>$
Use      single line : Oct  9 05:28:52 magento sshd[1304]: Invalid user k...

Results
=======

Failregex: 1 total
|-  #) [# of hits] regular expression
|   1) [1] ^.*sshd\[\d*\]: Invalid user .* from <HOST>$
`-

Ignoreregex: 0 total

Date template hits:
|- [# of hits] date format
|  [1] (?:DAY )?MON Day 24hour:Minute:Second(?:\.Microseconds)?(?: Year)?
`-

Lines: 1 lines, 0 ignored, 1 matched, 0 missed [processed in 0.00 sec]
```

fail2ban-regex testing

Synopsis

fail2ban-regex [options] LOG REGEX [ignoreregex]

Example using LOG file and command REGEX

```
fail2ban-regex /var/log/secure '^.*sshd\[\d*\]: Invalid user .* from <HOST>$'

Running tests
=============

Use    failregex line : ^.*sshd\[\d*\]: Invalid user .* from <HOST>$
Use         log file : /var/log/secure
Use         encoding : UTF-8

Results
=======

Failregex: 81 total
|-  #) [# of hits] regular expression
|   1) [81] ^.*sshd\[\d*\]: Invalid user .* from <HOST>$
`-

Ignoreregex: 0 total

Date template hits:
|- [# of hits] date format
|  [549] (?:DAY )?MON Day 24hour:Minute:Second(?:\.Microseconds)?(?: Year)?
`-

Lines: 549 lines, 0 ignored, 81 matched, 468 missed [processed in 0.20 sec]
```

fail2ban-regex testing

Example using LOG file and Filter REGEX

fail2ban-regex /var/log/secure /etc/fail2ban/filter.d/my-sshd.local

```
Running tests
=============
Use    failregex filter file : my-sshd, basedir: /etc/fail2ban
Use         maxlines : 10
Use         log file : /var/log/secure
Use         encoding : UTF-8

Results
=======

Failregex: 283 total
|-  #) [# of hits] regular expression
|   2) [81] ^.*sshd\[\d*\]: Invalid user .* from <HOST>$
|  11) [7] ^\s*(<[^.]+\.[^.]+>)?\s*(?:\S+ )?(?:kernel: \[ *\d+\.\d+\] )?(?:@vserver_\S+
)?(?:(?:\[\d+\])?:\s+[\[\(]?sshd(?:\(\S+\))?[\]\)]?:?|[\[\(]?sshd(?:\(\S+\))?[\]\)]?:?(?:\[\d+\])?:?)?\s(?:\[ID \d+ \S+\])?\s*Received disconnect fr\
om <HOST>: 3: \S+: Auth fail$
|  12) [28] ^\s*(<[^.]+\.[^.]+>)?\s*(?:\S+ )?(?:kernel: \[ *\d+\.\d+\] )?(?:@vserver_\S+
)?(?:(?:\[\d+\])?:\s+[\[\(]?sshd(?:\(\S+\))?[\]\)]?:?|[\[\(]?sshd(?:\(\S+\))?[\]\)]?:?(?:\[\d+\])?:?)?\s(?:\[ID \d+ \S+\])?\s*Received disconnect f\
rom <HOST>: 11: Bye Bye$
|  13) [76] ^\s*(<[^.]+\.[^.]+>)?\s*(?:\S+ )?(?:kernel: \[ *\d+\.\d+\] )?(?:@vserver_\S+
)?(?:(?:\[\d+\])?:\s+[\[\(]?sshd(?:\(\S+\))?[\]\)]?:?|[\[\(]?sshd(?:\(\S+\))?[\]\)]?:?(?:\[\d+\])?:?)?\s(?:\[ID \d+ \S+\])?\s*Received disconnect f\
rom <HOST>: 11:\s*$
|  14) [71] ^\s*(<[^.]+\.[^.]+>)?\s*(?:\S+ )?(?:kernel: \[ *\d+\.\d+\] )?(?:@vserver_\S+
)?(?:(?:\[\d+\])?:\s+[\[\(]?sshd(?:\(\S+\))?[\]\)]?:?|[\[\(]?sshd(?:\(\S+\))?[\]\)]?:?(?:\[\d+\])?:?)?\s(?:\[ID \d+ \S+\])?\s*Connection closed by \
<HOST>\s*$
|  15) [17] ^\s*(<[^.]+\.[^.]+>)?\s*(?:\S+ )?(?:kernel: \[ *\d+\.\d+\] )?(?:@vserver_\S+
)?(?:(?:\[\d+\])?:\s+[\[\(]?sshd(?:\(\S+\))?[\]\)]?:?|[\[\(]?sshd(?:\(\S+\))?[\]\)]?:?(?:\[\d+\])?:?)?\s(?:\[ID \d+ \S+\])?\s*Did not receive ident\
ification string from <HOST>\s*$
|  17) [3] ^\s*(<[^.]+\.[^.]+>)?\s*(?:\S+ )?(?:kernel: \[ *\d+\.\d+\] )?(?:@vserver_\S+
)?(?:(?:\[\d+\])?:\s+[\[\(]?sshd(?:\(\S+\))?[\]\)]?:?|[\[\(]?sshd(?:\(\S+\))?[\]\)]?:?(?:\[\d+\])?:?)?\s(?:\[ID \d+ \S+\])?\s*User .+ from <HOST> no\
t allowed because none of user's groups are listed in AllowGroups\s*$
`-

Ignoreregex: 0 total

Date template hits:
|- [# of hits] date format
|  [549] (?:DAY )?MON Day 24hour:Minute:Second(?:\.Microseconds)?(?: Year)?
`-

Lines: 549 lines, 0 ignored, 283 matched, 266 missed [processed in 1.80 sec]
Missed line(s): too many to print.  Use --print-all-missed to print all 266 lines
```

fail2ban-regex CL options

- **--print-all-matched**

  Print all matched lines

- **--print-all-missed**

  Print all missed lines, no matter how many there are

- **-v**

  Verbose output. Shows timestamp when each IP was banned and the date format style matched

Regular Expressions

- Lines in the log files that fail2ban will process:
  - Must have a date/time stamp
  - Must have an IP address of a host (You can't ban a host without an IP address!)
- In order for a log line to match your failregex, it actually has to match in two parts
  - The beginning of the line has to match a timestamp pattern or regex, and
  - The remainder of the line has to match your failregex. If the failregex is anchored with a leading ^, ***then the anchor refers to the start of the remainder of the line***, after the timestamp and intervening whitespace
  - You must use the special **<HOST>** tag in your failregex as a placeholder for fail2ban to capture the IP address from the log line
- fail2ban is real good at identifying date/time information from a log line no matter how it is formatted
- In the action scripts, the tag **<ip>** will be replaced by the IP address of the host that was matched with the **<HOST>** tag

Custom Filters

- Copy and tweak an existing file instead of trying to create your .local filter from scratch
- **ignoreregex** is a regular expression of IP address(es) that fail2ban should ignore. For example, machines on your LAN and localhost (127.0.0.1)
- [INCLUDES] are definitions of regular expression shortcuts (regex snippets) available for use in your filter
- Regular expressions heavily use hierarchical shortcuts for complex pattern matching

Hierarchical shortcuts

Consider a failregex line:

```
^%(__prefix_line)srefused connect from \S+ \(<HOST>\)\s*$
```

Here is a shortcut defined in common.conf:

```
__prefix_line = \s*%(__bsd_syslog_verbose)s?\s*(?:%(__hostname)s )?(?:%(__kernel_prefix)s
)?(?:@vserver_\S+ )?%(__daemon_combs_re)s?\s%(__daemon_extra_re)s?\s*
```

And

```
_daemon = \S*
__hostname = \S+
__kernel_prefix = kernel: \[ *\d+\.\d+\]
__daemon_combs_re = (?:%(__pid_re)s?:\s+%(__daemon_re)s|%(__daemon_re)s%(__pid_re)s?:?)
__pid_re = (?:\[\d+\])
__daemon_re = [\[\(]?%(_daemon)s(?:\(\S+\))?[\]\)]?:?
__daemon_extra_re = (?:\[ID \d+ \S+\])
__bsd_syslog_verbose = (<[^.]+\.[^.]+>)
```

Hierarchical shortcuts

This failregex:

```
^%(__prefix_line)srefused connect from \S+ \(<HOST>\)\s*$
```

Becomes:

```
^\s*(<[^.]+\.[^.]+>)?\s*(?:\S+ )?(?:kernel: \[ *\d+\.\d+\])?(?:@vserver_\S+)?(?:(?:\[\d+\])?:\s+
[\[\(]?\S*(?:\(\S+\))?[\]\)]?:?|[\[\(]?\S*(?:\(\S+\))?[\]\)]?:?(?:\[\d+\])?:?)?\s(?:\[ID\d+
\S+\])?\s*refused connect from \S+ \(<HOST>\)\s*$
```

Regex Tips

- Use fail2ban-regex to expand hierarchical shortcuts for you!
- Use command line LOG and REGEX to develop your initial **failregex**
- Use actual LOG file with your command line REGEX to test against the actual log file
- Codify your REGEX into a custom .local filter
- Test your filter using fail2ban-regex with the actual LOG file and your FILTER file
- Copy an existing filter .conf file instead of developing from scratch
- Remember to name your filter file using a .local extension

A jail definition

- Must have 3 things
  - A **logpath**
  - A **filter**
  - An **action**
- To use the jail
  - It must also be **enabled**

Jail options

Name	Default	Description
enabled	false	All jails are disabled until explicitly enabled
protocol	tcp	Protocol to be banned
port	0:65535	Ports to be banned
maxretry	3	Number of matches (i.e. value of the counter) which triggers ban action on the IP.
findtime	600 sec	The counter is set to zero if no match is found within "findtime" seconds.
bantime	600 sec	Duration (in seconds) for IP to be banned for. Negative number for "permanent" ban.

Basic jail.local entry

```
[ssh-iptables]
#enabled  = false
enabled  = true
logpath  = /var/log/secure
filter   = sshd
action   = iptables[name=SSH, port=ssh, protocol=tcp]
#          mail-whois[name=SSH, dest=yourmail@mail.com]
maxretry = 5
```

Custom jail.local entry

```
[my-sshd]
enabled   = true
logpath   = /var/log/secure
filter    = my-sshd
banaction = iptables
port      = ssh
findtime  = 86400
bantime   = 86400
maxretry  = 3
```

action vs actionban vs banaction

- **banaction** – used in your jail definition (e.g. jail.local). Defines which **`<action>.conf`** or **`<action>.local`** file to use in the **`action.d`** directory. A variable used in in **`action_*`** definitions.
- **actionban** – used in the **`action.d/<action>.conf`** or **`<action>.local`** file. The actual linux command(s) used to perform a ban if this **banaction** is used by a jail.
- **action** – Mapped to one of the following values in **`jail.local`**. Defines everything you want fail2ban to do when the decision to ban is performed
  - **action_** – ban only
  - **action_mw** – ban & send email with whois to destemail
  - **action_mwl** – ban & send email and relevant log lines to destemail
  - **action_xarf** – ban & send xarf email to abuse contact of IP address & include relevant log lines
  - **action_cf_mwl** – ban IP on CloudFlare & send email with whois report and log lines
  - **action_badips** – Report ban via badips.com, and use as blacklist

Using Fail2Ban

- Install software
- Create a jail definition in **jail.local**
  - Specify **logpath** of log file(s) to monitor
  - Specify **filter** to use
  - Specify **action**(s) to perform
  - Override default settings as necessary
- Test jail using fail2ban-regex:
  - fail2ban-regex **logpath** /path/to/**filter**.[conf|local]
  - debug
  - enable jail
- Start Jail
  - fail2ban-client reload
  - fail2ban-client start <jail>

jail.local

```
[sshd]
enabled         = true
banaction       = iptables
```

paths-fedora.conf

```
before          = paths-common.conf
syslog_authpriv = /var/log/secure
```

paths-common.conf

```
sshd_log        = %(syslog_authpriv)s
```

jail.conf

```
before          = paths-fedora.conf
logpath         = %(syslog_authpriv)s
filter          = %(__name__)s
banaction       = iptables-multiport
action          = %(action_)s
action_         = %(banaction)s[name=%(__name__)s,
                  bantime="%(bantime)s", port="%(port)s",
                  protocol="%(protocol)s", chain="%(chain)s"]
[sshd]
logpath         = %(sshd_log)s
```

Combining everything

jail.local

```
#global setting
action = %(action_)s

#jail definition
[my-sshd]
enabled    = true
port       = ssh
banaction = iptables
filter     = my-sshd
logpath    = /var/log/secure
findtime   = 86400
bantime    = 86400
maxretry   = 3
```

jail.conf

```
action_ = %(banaction)s
[name=%(__name__)s,
bantime="%(bantime)s",
port="%(port)s",
protocol="%(protocol)s",
chain="%(chain)s"]

which becomes:

action_ = iptables
      [name=my-sshd,
      bantime="86400",
      port="ssh",
      protocol="tcp",
      chain="INPUT"]
```

action.d/iptables.conf

```
actionban = <iptables> -I f2b-<name> 1 -s <ip> -j <blocktype>

iptables -I f2b-my-sshd 1 -s 1.2.3.4 -j REJECT -reject-with
cimp-port-unreachable
```

iptables.conf

```
[INCLUDES]
before = iptables-common.conf
```

iptables-common.conf

```
chain      = INPUT
protocol   = tcp
port       = ssh
blocktype  = REJECT -reject-with cimp-port-unreachable
iptables   = iptables <lockingopt>
lockingopt =
```

fail2ban magic

```
__name__    = my-sshd     (filter name)
name        = my-sshd     (jail name)
<HOST>    => ip
```

Action Tags

- <iptables>
- <blocktype>
- <chain>
- <returntype>
- <port>
- <protocol>
- <logpath>
- <keyfile>
- <domain>
- <ttl>
- <sender>
- <sendername>
- <dest>
- <failures>
- <category>
- <email>
- <apikey>
- <service>
- <matches>
- <cftoken>
- <mailcmd>
- <mailargs>
- <message>
- <userid>
- <lines>
- <tmpfile>
- <srcport>
- <myip>
- <tcpflags>
- <maxbufferage>
- <minreportinterval>
- <grepopts>
- <getcmd>
- <mnwurl>
- <nsupdatecmd>
- <loglines>

Predefined Action Tags

Tag	Description
ip	IPv4 IP address to be banned
name	Name of jail
__name__	Name of filter
failures	Number of times the failure occurred
ipfailures	As per failures, but total of all failures for that ip address across all jails from the fail2ban persistent database. Therefore the database must be set for this tag to function
ipjailfailures	As per ipfailures, but total based on the IPs failures for the current jail
time	UNIX (epoch) time of the ban
matches	concatenated string of the log file lines of the matches that generated the ban. Many characters interpreted by shell get escaped to prevent injection, nevertheless use with caution
ipmatches	As per matches, but includes all lines for the IP which are contained with the fail2ban persistent database. Therefore the database must be set for this tag to function

Actions

- It is possible to specify several actions, on separate lines. For example
  - You can react to a SSH break-in attempt by first adding a new firewall rule to block the host
  - Then retrieve some information about the offending host using whois
  - And finally sending an e-mail notification.
- Or maybe you just want to received a notification on your Jabber account when someone accesses the page /donotaccess.html on your web server.

Predefined banactions

- **dummy** – Just log IP bans/unbans to a log file
- **iptables** – watch a single TCP/IP port
- **iptables-multiport** – watches multiple port (like http & https)
- **iptables-multiport-log** – just like **iptables-multiport**, but also logs dropped packets
- **sendmail** – Send banned IP address by email
- **sendmail-whois** – Send whois info for banned IP by email
- **sendmail-buffered** – Send banned IP addresses after each <line> addresses are banned (default 5)

Action Options

- These are various options for an action. They are defined in the `<action>.conf` or `<action>.local` file
  - **actionstart** – the command(s) issued when first starting the action
  - **actionstop** – the command(s) issue to stop the action
  - **actioncheck** – the command(s) executed before each actionban command
  - **actionban** – the command(s) executed when banning an IP
  - **actionunban** – the command(s) execute when unbanning an IP

My Settings

- **findtime** = 86400 (1 day)
- **bantime** = 86400 (1 day)
- **maxretry** = 3

Remediation Results

- Our website has been operating without incident since attack
- We are consistently always blocking 80 IP addresses at any one time for SSH
- However, we're blocking about 3200-3300 IPs for a WordPress login vulnerability
- Baidu is still trying, but failing
- 97% of bans attempt to exercise the XMLRPC vulnerability
- 2.5% of bans attempt to login using SSH
- 0.5% is everything else
- I don't see any more DoS attempts

Our Fail2ban Jails

Jail	Description
magento-checkout	Block specially crafted GET requests that hang httpd
apache-auth	Block hack attempts on the WordPress XMLRPC vulnerability
my-sshd	Custom jail to identify and block additional Failregex's that the default installation does not catch
wp-login-attack	Protect WordPress from brute-force password attempts
wp-attack	Protect WordPress from common vulnerability probes
block-baidu	Blocks the Chinese bot called "Baidu“
apache-fakegooglebot	Blocks “fake” googlebot scans

Apache-fake-googlebot

```
[apache-fakegooglebot]
port       = http,https
logpath    = /var/log/httpd/mag*access.log
maxretry   = 1
findtime   = 172800
bantime    = 172800
enabled    = true
ignorecommand =
%(ignorecommands_dir)s/apache-
fakegooglebot <ip>
```

- Fakegooglebot command
  - Reverse DNS lookup of <ip> to get **name**
  - Forward lookup of **name** to get **googleip**
  - Compare **googleip** to <ip>
  - If the IPs match, a real googlebot, return 0 (False Fake)
  - If IPs don't match, a fake googlebot, return 1 (True Fake)

iptable bans

- Apache-Auth: 5
- Apache-fakegooglebot: 7
- Block-spider (Baidu): 10
- Magento-checkout: 4
- SSH blocks: 58
- WP attack: 5
- WP login attack: 3268

iptables

```
# iptables -L -n
Chain INPUT (policy ACCEPT)
target                prot opt source               destination
f2b-my-sshd           tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:22
f2b-wp-attack         tcp  --  0.0.0.0/0            0.0.0.0/0           multiport dports 80,443
f2b-block-baidu       tcp  --  0.0.0.0/0            0.0.0.0/0           multiport dports 80,443
f2b-apache-auth       tcp  --  0.0.0.0/0            0.0.0.0/0           multiport dports 80,443
f2b-wp-login-attack   tcp  --  0.0.0.0/0            0.0.0.0/0           multiport dports 80,443
f2b-magento-checkout tcp  --  0.0.0.0/0            0.0.0.0/0           multiport dports 80,443
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:80
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:443
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

iptables

```
Chain f2b-apache-auth (1 references)
target     prot opt source               destination
REJECT     all  --  99.89.46.24          0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  99.59.119.114        0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  99.252.102.14        0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  99.174.237.99        0.0.0.0/0           reject-with icmp-port-unreachable

 <130 more entries deleted>

REJECT     all  --  96.40.32.101         0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  92.16.149.24         0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  90.231.113.135       0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  88.182.180.124       0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  86.122.112.218       0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  83.243.219.101       0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  83.160.122.141       0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  83.153.247.131       0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  83.114.107.18        0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  83.112.206.86        0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  107.77.106.24        0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  1.136.96.136         0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  1.124.48.23          0.0.0.0/0           reject-with icmp-port-unreachable
RETURN     all  --  0.0.0.0/0            0.0.0.0/0

Chain f2b-block-baidu (1 references)
target     prot opt source               destination
REJECT     all  --  180.76.15.162        0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  180.76.15.137        0.0.0.0/0           reject-with icmp-port-unreachable
RETURN     all  --  0.0.0.0/0            0.0.0.0/0

Chain f2b-magento-checkout (1 references)
target     prot opt source               destination
RETURN     all  --  0.0.0.0/0            0.0.0.0/0
```

```
Chain f2b-my-sshd (1 references)
target     prot opt source               destination
REJECT     all  --  74.50.142.90         0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  61.178.245.159       0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  52.174.42.74         0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  51.254.46.199        0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  37.187.143.217       0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  27.251.35.202        0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  221.194.47.249       0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  221.194.47.229       0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  221.194.47.224       0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  221.194.47.208       0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  211.144.74.5         0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  204.140.17.62        0.0.0.0/0           reject-with icmp-port-unreachable

  <30 more entries deleted>

REJECT     all  --  113.161.82.184       0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  103.235.234.134      0.0.0.0/0           reject-with icmp-port-unreachable
RETURN     all  --  0.0.0.0/0            0.0.0.0/0

Chain f2b-wp-attack (1 references)
target     prot opt source               destination
RETURN     all  --  0.0.0.0/0            0.0.0.0/0

Chain f2b-wp-login-attack (1 references)
target     prot opt source               destination
REJECT     all  --  85.12.192.40         0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  178.219.88.0         0.0.0.0/0           reject-with icmp-port-unreachable
RETURN     all  --  0.0.0.0/0            0.0.0.0/0
```


Q&A