Fail2Ban

What is Fail2Ban:

fail2ban-logo.jpg
Fail2Ban is an intrusion prevention tool to prevent brute-force attacks or heavy requests that are repetitive and insecure.

Once you create a jail and create a filter for that jail, fail2ban will analyze the regex used in the filter to scan a file for a string that matches and then the jail will ban it using the service specified such as a firewall or a network blackhole (null route) to drop any incoming connections from that IP.

In this introduction to fail2ban you will learn how to create your jails and filters for multiple services, and how to to tweak those functions and make sure they are working correctly.

Why Fail2Ban:

Everyone knows about cphulk, so the first question some of you might ask yourself is why use Fail2Ban when you have cphulk? The advantage of fail2ban over cphulk is that you can install fail2ban on every Linux or BSD distribution, cphulk is also limited to WHM servers, also fail2ban can be highly customized to monitor any service, cphulk is limited to monitor the services that are installed with WHM and that contain password authentication.

Installation:

To install fail2ban on centos you will need the epel-release package installed, once it's installed you can then proceed to install the fail2ban package:

```
yum install epel-release && yum install fail2ban
```

And on Ubuntu you can simply use apt-get to install fail2ban:

```
apt-get install fail2ban
```

Enable and start fail2ban:

For systemd (Ubuntu 16.04+ and Centos 7+):

```
systemctl start fail2ban
systemctl enable fail2ban
```

For upstart (Ubuntu 14.04 and down and Centos 6 and down):

```
service fail2ban start
chkconfig fail2ban on
```

Once it is installed you will want to cp jail.conf into jail.local, the global modifications will be set in the jail.local, jail.conf is used as a template to the global presets, this means if the value is not specified in the jail you created it will refer to the values in jail.local, example IP white listing, or ban time.

```
cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local
```

Here is an example of global configurations you might want to have set for all your jails /etc/fail2ban/jail.local:

```
vi /etc/fail2ban/jail.local
```

> You can add our IP here to make sure you do not get banned.

> [DEFAULT]
> ignoreip = 127.0.0.1 # You can add our IP here to make sure you do not get banned.
> destemail = youraccount@email.com # For alerts
> sendername = Fail2BanAlerts

There are other settings you can change in the jail.local but i would recommend to add them specifically to your jail so the rules change depending on the jail.

Creating a custom access-log jail:

In the directory /etc/fail2ban/jail.d/ you can create new jails.

The best practice is to create 1 jail per rule in the jail.d directory and then create a filter for that jail.

So let's create our first jail that will read the access logs to ban IP’s who try to access a page on a domain in a folder called admin.

```
vi /etc/fail2ban/jail.d/(JAIL_NAME).conf
```

> This jail will look in the apache access logs for a user and then use the filer called block_traffic to add ip’s to iptables.

> “
> [JAIL_NAME] *#You can change this for your jail name*
> enabled = true
> port = http,https *# If the jail you are creating is for another protocol like ssh add it here*
> filter = block_traffic
> banaction = iptables-allports *# Just use iptables and keep it easy*
> logpath = /home/USER/access-logs/* *# You can change it to wherever the access logs are located*
> bantime = 3600 *# Change this however you want, you can change it to -1 for a permanent ban.*
> findtime = 150 *# Refreshes the logs, set time in seconds*
> maxretry = 3 *# If it finds 3 matching strings in the access logs it will ban the ip*

Creating a custom filter for the access-log jail:

This rule will look for any HTTP get or post request for /admin folder, the <HOST> is the IP in the logs the filter will read to add them to a iptables chain. you can replace the word admin for anything, example bot or be wp-admin for wordpress and add the IP's of the customer in the white list of the jail so they can connect to /wp-admin (for example).

The * in the regex and in the jail/filter is a wildcard to grab all the arguments before or after the syntax matching.

```
vi /etc/fail2ban/filter.d/block_traffic.conf
```

> “
> [Definition]
> failregex = ^<HOST> -.*"(GET|POST).*admin.*
> ignoreregex =

XMLRPC filter + jail example:

So here an example i used in the past to create a jail to block xmlrpc request:

```
vi /etc/fail2ban/jail.d/xmlrpc.conf
```

> [xmlrpc]
> enabled = true
> port   = http,https
> filter   = xmlrpc
> banaction = iptables-allports
> logpath = /home/*/access-logs/*
> bantime = 3600
> findtime = 150
> maxretry = 3

And here is what your filter should look like.

```
vi /etc/fail2ban/filter.d/xmlrpc.conf
```

> [Definition]
> failregex = ^<HOST> -.*"(GET|POST).*\/xmlrpc\.php.* HTTP\/.*
> ignoreregex =

Jail for SSH:

Now let’s create a few jail for SSH and Pure-FTPd, We will start by creating a ssh jail:

```
vi /etc/fail2ban/jail.d/ssh.conf
```

> [ssh-iptables]
> enabled = true
> filter   = sshd
> banaction = iptables-allports
> logpath = /var/log/secure
> maxretry = 5

And if you look at /etc/fail2ban/filter.d/ you will see there is already a filter for ssh so no need to do anything else.

Jail for Pure-FTPd:

Now let’s create a jail for Pure-FTPd

```
vi /etc/fail2ban/jail.d/pureftpd.conf
```

> [pureftpd-iptables]
> enabled = true
> port   = ftp
> filter   = pure-ftpd
> logpath = /var/log/messages
> maxretry = 3

And a filter for pure-ftpd.conf

```
vi /etc/fail2ban/filter.d/pure-ftpd.conf
```

> [Definition]
> failregex = pure-ftpd: \(\?@<HOST>\) \[WARNING\] Authentication failed for user
> ignoreregex =

Fail2Ban Client:

You will mostly use the fail2ban-client to unban a customer's IP, or you need to restart a jail after configuration changes, please note it is important to restart the service after every jail change.

Restart the fail2ban service:

```
fail2ban restart
```

To verify what jails are active you can do:

```
fail2ban-client status
```

To reload a jail after doing changes in your conf you can do:

```
fail2ban-client reload <JAIL>
```

To view if the jail is active and how many IP's it has banned:

```
fail2ban-client status <JAIL>
```

If you want to unban an IP:

```
fail2ban-client set <JAIL> unbanip X.X.X.X
```

If you want to add an IP to a jail ban:

```
fail2ban-client -vvv set <JAIL> banip X.X.X.X
```

To start fail2ban in debug mode if fail2ban does not start:

```
cd /usr/src/fail2ban-X.X.X.(VERSION)/
fail2ban-client -vvv start
```

> And here is a list for more Fail2ban commands:
> http://www.fail2ban.org/wiki/index.php/Commands

The default path for logs is: /var/log/fail2ban.log, if ever you have a hard time starting a jail or working with a jail i would recommend you go through the logs

Regex check:

The regex check is used to validate the syntax you will use for your filter, so let’s say you want to create a custom rule to check the access logs you can test the filter regex first by doing:

```
fail2ban-regex '/home/USER/access-logs/* ' '^<HOST> -.*"(GET|POST).*admin.*'
```

With fail2ban-regex you can test to make sure it will read a log file and try to get an ip hit from your failregex, the first ‘single quotes’ is the placement of the logs and the second ‘single quotes’ is the failregex syntax you are testing.

Revision #17
Created 2017-07-03 00:33:34 UTC by Dave
Updated 2020-01-17 08:52:33 UTC by Carl