WP fail2ban Manual

Charles Lecklider

Nov 21, 2018

Contents

WP fail2ban is a WordPress plugin to write a myriad of events to syslog for integration with fail2ban.

CHAPTER 1

Introduction

1.1 History

1.2 Features

1.2.1 CloudFlare and Proxy Servers

WPf2b can be configured to work with CloudFlare and other proxy servers. For a brief overview see *WP_FAIL2BAN_PROXIES*.

1.2.2 Comments

WPf2b can log comments. See *WP_FAIL2BAN_LOG_COMMENTS*.

1.2.3 Pingbacks

WPf2b logs failed pingbacks, and can log all pingbacks. For a brief overview see *WP_FAIL2BAN_LOG_PINGBACKS*.

1.2.4 Spam

WPf2b can log comments marked as spam. See *WP_FAIL2BAN_LOG_SPAM*.

1.2.5 User Enumeration

WPf2b can block user enumeration. See *WP_FAIL2BAN_BLOCK_USER_ENUMERATION*.

1.2.6 Work-Arounds for Broken syslogd

WPf2b can be configured to work around most syslogd weirdness. For a brief overview see *WP_FAIL2BAN_SYSLOG_SHORT_TAG* and *WP_FAIL2BAN_HTTP_HOST*.

1.2.7 Blocking Users

WPf2b can be configured to short-cut the login process when the username matches a regex. For a brief overview see *WP_FAIL2BAN_BLOCKED_USERS*.

1.2.8 *mu-plugins* Support

WPf2b can easily be configured as a must-use plugin.

CHAPTER 2

Configuration

2.1 fail2ban

2.1.1 Filters

1. Copy *wordpress-hard.conf* and *wordpress-soft.conf* to your *fail2ban/filters.d* directory
2. Edit *jail.local* to include something like:

```
[wordpress-hard]
enabled = true
filter = wordpress-hard
logpath = /var/log/auth.log
maxretry = 1
port = http,https

[wordpress-soft]
enabled = true
filter = wordpress-soft
logpath = /var/log/auth.log
maxretry = 3
port = http,https
```

3. Reload or restart *fail2ban*

2.1.2 *wordpress-hard.conf* and *wordpress-soft.conf*

There are some things that are almost always malicious, e.g. blocked users and pingbacks with errors. *wordpress-hard.conf* is designed to catch these so that you can ban the IP immediately.

Other things are relatively benign, like a failed login. You can't let people try forever, but banning the IP immediately would be wrong too. *wordpress-soft.conf* is designed to catch these so that you can set a higher retry limit before banning the IP.

For the avoidance of doubt: you should be using *both* filters.

2.2 *mu-plugins* Support

One of the better ways is to install *WPf2b* as usual and then create a symlink in `mu-plugins`:

```
lrwxr-xr-x  1  www  www  38  4 Nov 16:24 wp-fail2ban.php -> ../plugins/wp-fail2ban/wp-
↪fail2ban.php
```

This has the advantage that you can update *WPf2b* as usual without having to update `mu-plugins` directly. You don't need to activate *WPf2b*, but it won't hurt if you do.

CHAPTER 3

define() Constants

3.1 WP_FAIL2BAN_AUTH_LOG

New in version 2.2.0.

By default, *WPf2b* uses **LOG_AUTH** for logging authentication success or failure. However, some systems use **LOG_AUTHPRIV** instead, but there's no good run-time way to tell. If your system uses **LOG_AUTHPRIV** you should add the following to `wp-config.php`:

```
define('WP_FAIL2BAN_AUTH_LOG', LOG_AUTHPRIV);
```

3.2 WP_FAIL2BAN_BLOCK_USER_ENUMERATION

New in version 2.1.0.

Brute-forcing WP requires knowing a valid username. Unfortunately, WP makes this all but trivial.

Based on a suggestion from *@geeklol* and a plugin by *@ROIBOT*, *WPf2b* can now block user enumeration attempts. Just add the following to `wp-config.php`:

```
define('WP_FAIL2BAN_BLOCK_USER_ENUMERATION', true);
```

3.3 WP_FAIL2BAN_BLOCKED_USERS

New in version 2.0.0.

The bots that try to brute-force WordPress logins aren't that clever (no doubt that will change), but they may only make one request per IP every few hours in an attempt to avoid things like *fail2ban*. With large botnets this can still create significant load.

Based on a suggestion from *@jmadea*, *WPf2b* now allows you to specify a regex that will shortcut the login process if the requested username matches.

For example, putting the following in `wp-config.php`:

```
define('WP_FAIL2BAN_BLOCKED_USERS', '^admin$');
```

will block any attempt to log in as **admin** before most of the core WordPress code is run. Unless you go crazy with it, a regex is usually cheaper than a call to the database so this should help keep things running during an attack.

WPf2b doesn't do anything to the regex other than make it case-insensitive.

If you're running PHP 7, you can now specify an array of users instead:

```
define('WP_FAIL2BAN_BLOCKED_USERS', ['admin', 'another', 'user']);
```

3.4 WP_FAIL2BAN_COMMENT_LOG

New in version 3.5.0.

See *WP_FAIL2BAN_LOG_COMMENTS*.

3.5 WP_FAIL2BAN_HTTP_HOST

New in version 3.0.0.

This is for some flavours of Linux where *WP_FAIL2BAN_SYSLOG_SHORT_TAG* isn't enough.

If you configure your web server to set an environment variable named **WP_FAIL2BAN_SYSLOG_SHORT_TAG** on a per-virtual host basis, *WPf2b* will use that in the syslog tag. This allows you to configure a unique tag per site in a way that makes sense for your configuration, rather than some arbitrary truncation or hashing within the plugin.

Note: This feature has not been tested as extensively as others. While I'm confident it works, FreeBSD doesn't have this problem so this feature will always be second-tier.

3.6 WP_FAIL2BAN_LOG_COMMENTS

New in version 3.5.0.

WPf2b can now log comments. To enable this feature, add the following to `wp-config.php`:

```
define('WP_FAIL2BAN_LOG_COMMENTS', true);
```

By default, *WPf2b* uses **LOG_USER** for logging comments. If you'd rather it used a different facility you can change it by adding something like the following to `wp-config.php`:

```
define('WP_FAIL2BAN_COMMENT_LOG', LOG_LOCAL3);
```

3.7 WP_FAIL2BAN_LOG_PASSWORD_REQUEST

New in version 3.5.0.

3.8 WP_FAIL2BAN_LOG_PINGBACKS

New in version 2.2.0.

Based on a suggestion from *maghe*, *WPf2b* can now log pingbacks. To enable this feature, add the following to `wp-config.php`:

```
define('WP_FAIL2BAN_LOG_PINGBACKS', true);
```

By default, *WPf2b* uses **LOG_USER** for logging pingbacks. If you'd rather it used a different facility you can change it by adding something like the following to `wp-config.php`:

```
define('WP_FAIL2BAN_PINGBACK_LOG', LOG_LOCAL3);
```

3.9 WP_FAIL2BAN_LOG_SPAM

New in version 3.5.0.

WPf2b can now log spam comments. To enable this feature, add the following to `wp-config.php`:

```
define('WP_FAIL2BAN_LOG_SPAM', true);
```

The comment ID and IP will be written to *WP_FAIL2BAN_AUTH_LOG* and matched by *wordpress-hard.conf*.

3.10 WP_FAIL2BAN_OPENLOG_OPTIONS

New in version 3.5.0.

3.11 WP_FAIL2BAN_PINGBACK_LOG

New in version 2.2.0.

See *WP_FAIL2BAN_LOG_PINGBACKS*.

3.12 WP_FAIL2BAN_PROXIES

New in version 2.0.0.

The idea here is to list the IP addresses of the trusted proxies that will appear as the remote IP for the request. When defined:

- If the remote address appears in the **WP_FAIL2BAN_PROXIES** list, *WPf2b* will log the IP address from the *X-Forwarded-For* header

- If the remote address does not appear in the **WP_FAIL2BAN_PROXIES** list, *WPf2b* will return a 403 error
- If there's no *X-Forwarded-For* header, *WPf2b* will behave as if **WP_FAIL2BAN_PROXIES** isn't defined

To set **WP_FAIL2BAN_PROXIES**, add something like the following to `wp-config.php`:

```
define('WP_FAIL2BAN_PROXIES','192.168.0.42,192.168.42.0/24');
```

WPf2b doesn't do anything clever with the list - beware of typos!

3.13 WP_FAIL2BAN_REMOTE_ADDR

New in version 3.6.0.

Some themes and plugins anonymise requests

3.14 WP_FAIL2BAN_SYSLOG_SHORT_TAG

New in version 3.0.0.

Some flavours of Linux come with a *syslogd* that can't cope with the normal message format *WPf2b* uses; basically, they assume that the first part of the message (the tag) won't exceed some (small) number of characters, and mangle the message if it does. This breaks the regex in the *fail2ban* filter and so nothing gets blocked.

Adding:

```
define('WP_FAIL2BAN_SYSLOG_SHORT_TAG', true);
```

to `functions.php` will make *WPf2b* use `wp` as the syslog tag, rather than the normal `wordpress`. This buys you 7 characters which may be enough to work around the problem, but if it's not enough you should look at *WP_FAIL2BAN_HTTP_HOST* or *WP_FAIL2BAN_TRUNCATE_HOST* too.

3.15 WP_FAIL2BAN_TRUNCATE_HOST

New in version 3.5.0.

If you've set *WP_FAIL2BAN_SYSLOG_SHORT_TAG* and defining *WP_FAIL2BAN_HTTP_HOST* for each virtual host isn't appropriate, you can set **WP_FAIL2BAN_TRUNCATE_HOST** to whatever value you need to make *syslog* happy:

```
define('WP_FAIL2BAN_TRUNCATE_HOST', 8);
```

This does exactly what the name suggests: truncates the host name to the length you specify. As a result there's no guarantee that what's left will be enough to identify the site.

3.16 WP_FAIL2BAN_XMLRPC_LOG

New in version 3.6.0.

This is for debugging and future development.

Attackers are doing weird things with XML-RPC, so this logs the raw post data to the file specified:

```
define('WP_FAIL2BAN_XMLRPC_LOG', '/var/log/xml-rpc.log');
```

CHAPTER 4

Filters

4.1 *wordpress-hard.conf*

```
# Fail2Ban filter for WordPress hard failures
# Auto-generated: 2018-11-04T16:40:53+00:00
#

[INCLUDES]

before = common.conf

[Definition]

_daemon = (?:wordpress|wp)

failregex = ^%(__prefix_line)sBlocked authentication attempt for .* from <HOST>$
            ^%(__prefix_line)sBlocked user enumeration attempt from <HOST>$
            ^%(__prefix_line)sSpam comment \d+ from <HOST>$
            ^%(__prefix_line)sXML-RPC multicall authentication failure from <HOST>$
            ^%(__prefix_line)sPingback error .* generated from <HOST>$
            ^%(__prefix_line)sAuthentication attempt for unknown user .* from <HOST>$
            ^%(__prefix_line)sXML-RPC authentication attempt for unknown user .* from
↪<HOST>$

ignoreregex =

# DEV Notes:
# Requires the 'WP fail2ban' plugin:
# https://github.com/invisnet/wp-fail2ban/
#
# Author: Charles Lecklider
```

4.2 *wordpress-soft.conf*

```
# Fail2Ban filter for WordPress soft failures
# Auto-generated: 2018-11-04T16:40:53+00:00
#

[INCLUDES]

before = common.conf

[Definition]

_daemon = (?:wordpress|wp)

failregex = ^%(__prefix_line)sAuthentication failure for .* from <HOST>$
            ^%(__prefix_line)sXML-RPC authentication failure for .* from <HOST>$

ignoreregex =

# DEV Notes:
# Requires the 'WP fail2ban' plugin:
# https://github.com/invisnet/wp-fail2ban/
#
# Author: Charles Lecklider
```

CHAPTER 5

Unit Tests

5.1 Results

```
Code Coverage Report:
  2018-11-04 16:24:19

 Summary:
  Classes:         (0/0)
  Methods:         (0/0)
  Lines:    95.95% (71/74)
```