WP fail2ban Manual

Charles Lecklider

May 11, 2023

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

As with many Open Source projects, *P fail2ban* started as way to scratch a particular itch. I had a dedicated server that was getting some unwelcome attention from various bots, and while it was trivial to configure *fail2ban* for `ssh` etc, WordPress was another story. Thus *WP fail2ban* was born late November 2011.

Since then it's slowly but steadily accumulated features, and much to my surprise, gained a considerable number of installs (30,000+ at the time of writing) - I really had no idea so many other people would be interested!

Between versions 3.5 and 3.6 there was a bit of a delay. I switched my development environment from Windows 10[1] to a FreeBSD workstation and a Linux laptop, life then decided to take its turn and get in the way for a bit, all while the shadow of Gutenberg loomed large over the future of WordPress. With the advent of ClassicPress[2] things started to look sunnier, so I dusted off the repo, put together some better documentation, braved the horrors of `svn`, and in November 2018 released 3.6 as a pseudo 7th anniversary present.

1.2 Future

Version 4 was born from a desire to visualise the things *WPf2b* was logging; being entirely separate and distinct from the core functionality, adding this as freemium features seemed like a good plan. Time will tell.

This logical separation will continue for all future versions - if you were happy with the way 3.6 worked you'll be happy with future versions too.

[1] It took me a while to realise that Microsoft really do want to turn Windows 10 into a toy, but I got there eventually.
[2] In the interests of full disclosure: I'm a Founding Committee Member and at the time of writing, Security Team Lead.

CHAPTER 2

Features

2.1 NEW - Multisite Support

Version 4.3 introduces proper support for multisite networks.

2.2 NEW - Block username logins

Sometimes it's not possible to block user enumeration (for example, if your theme provides Author profiles). Version 4.3 adds support for requiring the use of email addresses for login.

2.3 NEW - Filter for Empty Username Login Attempts

Some bots will try to login without a username. Version 4.3 logs these attempts and provides an "extra" filter to match them.

2.4 NEW - syslog Dashboard Widget

Ever wondered what's being logged? The new dashboard widget shows the last 5 messages; the Premium version keeps a full history to help you analyse and prevent attacks.

2.5 Remote Tools Add-on

The Remote Tools add-on provides extra features without adding bloat to the core plugin. For more details see the add-on page.

2.6 Support for 3rd-party Plugins

Version 4.2 introduced a simple API for authors to integrate their plugins with *WPf2b*, with 2 *experimental* add-ons:

- Contact Form 7
- Gravity Forms

2.7 CloudFlare and Proxy Servers

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

2.8 Comments

WPf2b can log both successful comments (see *WP_FAIL2BAN_LOG_COMMENTS*), and unsuccessful comments (see *WP_FAIL2BAN_LOG_COMMENTS_EXTRA*).

2.9 Pingbacks

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

2.10 Spam

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

2.11 User Enumeration

WPf2b can block user enumeration. See *WP_FAIL2BAN_BLOCK_USER_ENUMERATION*.

2.12 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*.

2.13 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*.

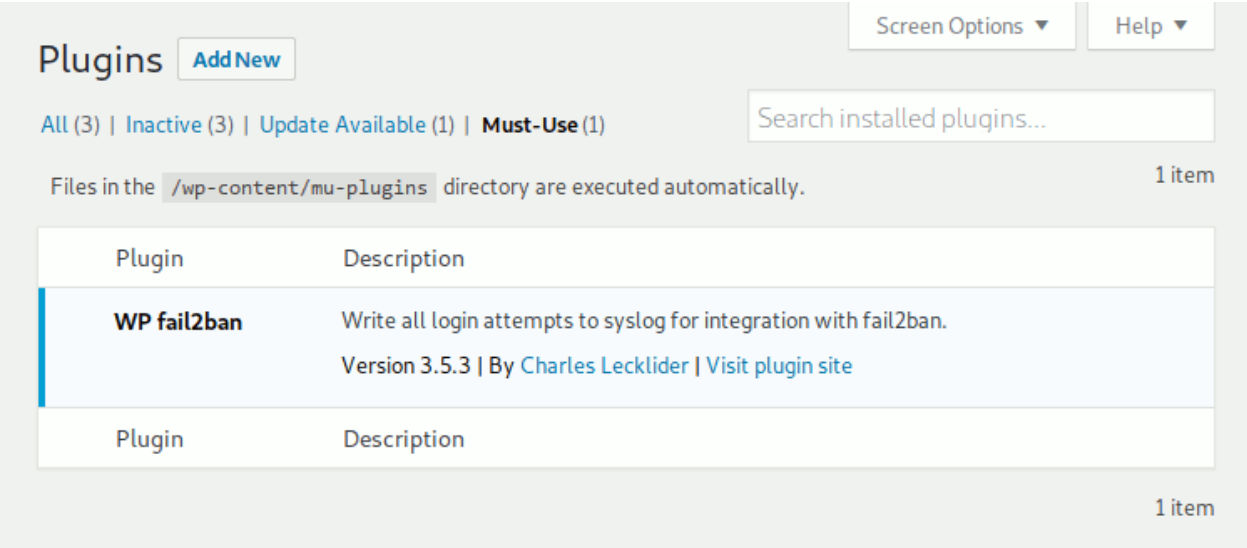
2.14 *mu-plugins* Support

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

CHAPTER 3

Installation

3.1 Is *WP fail2ban* Already Installed?



WP fail2ban pre-installed in *mu-plugins* in a new DigitalOcean WordPress droplet.

3.2 Overview

WPf2b installs just like any other WordPress plugin - you need do nothing differently.

3.2.1 Premium

The Premium version installs via Freemius.

Database

Activating *WPf2b* Premium creates two database tables:

- `wp_fail2ban_log`
- `wp_fail2ban_plugins`

WPf2b Premium never drops the database tables - it's your data.

CHAPTER 4

Configuration

Now you have *WPf2b* installed and activated it's time to make it do something useful.

4.1 WP fail2ban

The Free version of *WPf2b* is configured by defining constants in `wp-config.php`. If you're using the Premium version, or you know your way around `wp-config.php` already, skip ahead to *Logging*.

The first step is to check you can edit your `wp-config.php` file. If you're not sure how to do that you'll need to contact your hosting provider - for now you can skip ahead to configuring *fail2ban*.

The second step is to **take a backup** of `wp-config.php`. We're not going to touch any other part of WordPress, so if anything goes wrong and your site stops working, restoring this backup should get you running again.

4.2 Logging

The key concept behind *WPf2b* is logging *Events* to `syslog`. If *WPf2b* doesn't log an Event, or logs it to the wrong place, `fail2ban` won't work as it should. If in doubt go with the defaults - they should work for most systems, and once you understand how the pieces fit together you can revisit this.

4.2.1 Choosing the Events to Log

If you're unfamiliar with `fail2ban` and `syslog` I recommend **not** enabling any extra logging to start with - skip ahead to configuring *fail2ban*. *WPf2b* automatically handles the most important things with sensible defaults that should work for most systems.

4.2.2 Advanced Users

Events

Over the years *WPf2b* has accumulated a lot of logging ability (and there're even more on the way):

Event	Reference
Auth OK	*WP_FAIL2BAN_AUTH_LOG*
Auth Fail	
Blocked User	*WP_FAIL2BAN_BLOCKED_USERS*
Blocked User Enumeration	*WP_FAIL2BAN_BLOCK_USER_ENUMERATION*
Blocked Username Login	*WP_FAIL2BAN_BLOCK_USERNAME_LOGIN*
Comment	*WP_FAIL2BAN_LOG_COMMENTS*
Comment: Spam	*WP_FAIL2BAN_LOG_SPAM*
Attempted Comment: Post not found	*WP_FAIL2BAN_LOG_COMMENTS_EXTRA*
Attempted Comment: Closed post	*WP_FAIL2BAN_LOG_COMMENTS_EXTRA*
Attempted Comment: Trash post	*WP_FAIL2BAN_LOG_COMMENTS_EXTRA*
Attempted Comment: Draft post	*WP_FAIL2BAN_LOG_COMMENTS_EXTRA*
Attempted Comment: Password-protected post	*WP_FAIL2BAN_LOG_COMMENTS_EXTRA*
Pingback	*WP_FAIL2BAN_LOG_PINGBACKS*
Pingback error	*WP_FAIL2BAN_PINGBACK_ERROR_LOG*

You should consider enabling *Comment: Spam* and *Attempted Comment: Closed post*, and, if you don't use WordPress's commenting system at all, you should enable **all** the *Attempted Comment* Events.

Facilities

By default, *WPf2b* uses the following `syslog` Facilities and *Levels*:

What	Default	Level
Auth OK	*LOG_AUTH*	INFO
Auth Fail		NOTICE
Blocked User		
Blocked User Enum		
Comment	*LOG_USER*	INFO
Comment: Spam	*LOG_AUTH*	NOTICE
Comment: Post not found		
Comment: Closed post		
Comment: Trash post		
Comment: Draft post		
Comment: Password-protected post		
Pingback	*LOG_USER*	INFO
Pingback error	*LOG_AUTH*	NOTICE

Unfortunately, there is no way of knowing *a priori* which Facility goes where. There is a table of default locations of *Logfile Reference* for various OSs; if you're running something not listed there and you know where the various Facilities go, please either submit a PR on GitHub, or let me know in the forum.

4.3 fail2ban

`fail2ban` can be tricky to configure correctly; with so many flavours of Linux it's impossible to provide anything but general guidance.

4.3.1 Standard Filters

The filter files included are intended only as a starting point for those who want *WPf2b* to work "out of the box".

There is no "one size fits all" configuration possible for *fail2ban* - what may be a soft failure for one site should be treated as a hard failure for another, and vice versa. Careful thought should be given to what is appropriate for your environment.

Typical Settings

1. Copy *wordpress-hard.conf* and *wordpress-soft.conf* to your *fail2ban/filters.d* directory
2. Create a new file in *jail.d* called *wordpress.conf*:

```
[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
```

Note: Make sure you change `logpath` to the correct log for your OS. If your OS uses *systemd* it may be simpler and/or easier to install a real syslog service first.

---

3. Reload or restart *fail2ban*

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.

4.3.2 Custom Filters

You should never modify the standard *wordpress-hard.conf* and *wordpress-soft.conf* files. Instead, copy them to, for example, *wordpress-hard-custom.conf* and *wordpress-soft-custom.conf*, and edit those.

It is very rare that individual filter rules are modified, but new rules are common; there is always an entry in the "Updating" notes when there is any change to the rules. **It is your responsibility to ensure the rules in your custom filters are kept current.**

wordpress-extra.conf

Version 4 introduced a number of new logging options which didn't fit cleanly into either of the *hard* or *soft* filters - they're *extra*.

For example, if your site doesn't use WordPress comments at all, you could add the rules matching attempted comments to the *hard-custom* filter. Again, there is no "one size fits all" for these rules.

4.3.3 Updating

Whether you use the standard filter files or a highly-customised set of your own, **it is critical they are kept up to date**. There is always an entry in the "Updating" notes when the filter files need to be updated.

Obsolete filters may cause users to be blocked incorrectly, or attackers not to be detected.

WPf2b **cannot** update them for you.

See *Updating Filters*.

4.4 *mu-plugins* Support

There are two main reasons for using *mu-plugins*:

1. You need to load *WPf2b* before other security plugins[1],
2. You don't trust the site administrators.

4.4.1 Loading Early

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

```
# ln -s ../plugins/wp-fail2ban/wp-fail2ban.php
# ls -l
total 1
lrwxr-xr-x  1  www  www  38  4 Nov 16:24 wp-fail2ban.php -> ../plugins/wp-fail2ban/wp-
↪fail2ban.php
```

or for the Premium version:

```
# ln -s ../plugins/wp-fail2ban-premium/wp-fail2ban.php
# ls -l
total 1
lrwxr-xr-x  1  www  www  38  4 Nov 16:24 wp-fail2ban.php -> ../plugins/wp-fail2ban-
↪premium/wp-fail2ban.php
```

This has the advantage that you can update *WPf2b* as usual without having to update `mu-plugins` directly. For the free version you don't need to activate *WPf2b*, but you do for the Premium version.

4.4.2 Forcing Usage

The main objective here is to stop people fiddling with things, so there are necessarily some restrictions on configuring *WPf2b*.

[1] For example, WordFence, which assumes it's the only one.

WPf2b must be configured in `wp-config.php` - you can't use the Premium config UI; not only does it make no sense, it won't work[2].

The actual configuration itself is simple; for the **Free** version:

1. Extract the **Free** version of *WPf2b* into a directory called *wp-fail2ban* within *mu-plugins*.
2. symlink `wp-fail2ban.php`:

```
# ln -s wp-fail2ban/wp-fail2ban.php
# ls -l
total 1
lrwxr-xr-x  1  www  www  38  4 Nov 16:24 wp-fail2ban.php -> wp-fail2ban/wp-fail2ban.
↪php
```

3. **Keep** *WPf2b* **up-to-date**.

For the **Premium** version:

1. Extract the **Premium** version of *WPf2b* into a directory called *wp-fail2ban-premium* within *mu-plugins*.
2. symlink `wp-fail2ban.php`:

```
# ln -s wp-fail2ban-premium/wp-fail2ban.php
# ls -l
total 1
lrwxr-xr-x  1  www  www  38  4 Nov 16:24 wp-fail2ban.php -> wp-fail2ban-premium/wp-
↪fail2ban.php
```

3. **Keep** *WPf2b* **up-to-date**.

Keeping *WPf2b* up-to-date

It's that last step that catches out most people - WordPress doesn't check `mu-plugins` for updates, so by configuring *WPf2b* in this way **you are taking responsibility** for keeping *WPf2b* up-to-date. I do my best, but I cannot guarantee there will never be a critical problem with *WPf2b* - you and you alone are responsible for checking for updates and installing them.

4.5 Site Health Tool

New in version 5.0.0.

WP fail2ban uses the standard WordPress Site Health tool to check things are configured correctly.

4.5.1 Checking `fail2ban`

This works well for typical installations, but there are two things to note:

[2] It may look like it works now, but in a future release it will be blocked.

Running PHP with `chroot`

PHP will not be able to access files outside the `chroot`, so the health checks will not work.

If you **really** want them to work, you could ensure the `chroot` includes the parent directory of the document root (you may already have done this and moved `wp-config.php` outside the document root), and then use `nullfs` to mount the `fail2ban` install into the `chroot` above the document root. You will then need to tell WP fail2ban where to find your `fail2ban` install (see below).

Most people will simply disable the checks by adding this to `wp-config.php`

```
define('WP_FAIL2BAN_SITE_HEALTH_SKIP_FILTERS', true);
```

See also:

WP_FAIL2BAN_SITE_HEALTH_SKIP_FILTERS

Non-Standard Install Path for `fail2ban`

If your `fail2ban` install lives somewhere other than `/etc/fail2ban` or `/usr/local/etc/fail2ban` you will need to tell WP fail2ban where to find it by adding something like this to `wp-config.php`

```
/**
 * Be sure to change the path to point to your fail2ban install
 */
define('WP_FAIL2BAN_INSTALL_PATH', '/var/fail2ban');
```

Other Reasons

There are, of course, many other reasons why PHP won't be able to read the `fail2ban` filter files, e.g. tighter `chmod`, SELinux.

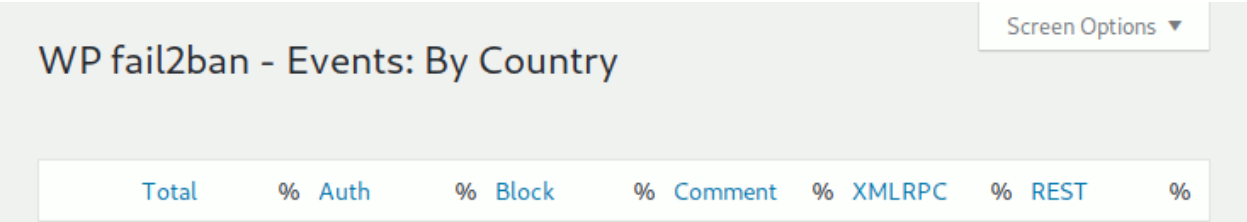
If you have a way to allow the health checks to run in any of these situations and think it may help others, please either write it up and submit a PR, or get in touch on the forums.

CHAPTER 5

Usage

5.1 Event Log

5.2 Report: Events by Country



5.3 Clearing the Cache

CHAPTER 6

Maintenance

While *WPf2b* needs no routine maintenance, from time to time there are changes that will require modifying your configuration.

6.1 Updating Filters

Keeping your `fail2ban` filters up-to-date is critical for the correct operation of *WPf2b*.

6.1.1 When to Update

Knowing when to update your filters has changed since 4.4.x. The release notes, as always, still say whether updating is necessary, but now that *WPf2b* strictly follows SemVer it's trivial to determine when an update is needed. In addition, the WordPress Site Health tool now checks if the filters are up-to-date.

SemVer

From v5.0.1 to v5.0.x You will **never** need to update the filters.

From v5.0.x to v5.1.0 You **may** need to update the filters to enable **new** features; existing configurations will continue to work as before.

From v5.x.y to v6.0.0 You **may** need to update the filters for **existing** features to work.

6.1.2 How to Update

The new filter files can be found in your WordPress install: `/wp-content/plugins/wp-fail2ban/filters.d` or `/wp-content/plugins/wp-fail2ban-premium/filters.d`.

The old filter files can be found in the `filter.d` directory in your `fail2ban` install.

It's usually a good idea to take a backup of the old filters.

Then, copy the new filter files to the `filter.d` directory, and reload the `fail2ban` jails:

```
# fail2ban-client reload
```

Tip: `fail2ban` can usually be found in:

- `/etc/fail2ban/`
- `/usr/local/etc/fail2ban/`

but may live somewhere else on your system.

If you're running Windows and know where `fail2ban` usually lives, please submit a PR for this page.

CHAPTER 7

Add-ons

Gravity Forms

CHAPTER 8

Developers

8.1 API

New in version 4.2.0: Added API to allow 3rd-party plug-ins.

8.1.1 Overview

The basic steps are:

Register Plugin

do_action (*string $action*, *string $slug*, *string $name*) → void
Register plugin.

Parameters

- **$action** (*string*) – Must be `wp_fail2ban_register_plugin`.
- **$slug** (*string*) – Plugin slug. This must be the actual plugin slug. Maximum length is 200.
- **$name** (*string*) – Plugin display name. This should be an unescaped string - HTML is allowed.

Throws

- **LengthException** – Either `$slug` or `$name` is too long; the `message` will say which.
- **RuntimeException** – Database error (Premium only).

Example

```
try {
    do_action('wp_fail2ban_register_plugin', 'my-plugin-slug', 'My Plugin Name');
} catch(\LengthException $e) {
    // slug or name too long
} catch(\RuntimeException $e) {
    // database error
}
```

Register Message

do_action (*string $action*, *string $slug*, *array $args*) → void

Parameters

- **$action** (*string*) – Must be `wp_fail2ban_register_message`.
- **$slug** (*string*) – The plugin slug used in *Register Plugin*.
- **$args['slug']** (*string*) – The message slug.
- **$args['fail']** (*string*) – Recommended action.
- **$args['priority']** (*int*) – *syslog* priority to use. Only the following priorities are supported:
  - LOG_CRIT
  - LOG_ERR
  - LOG_WARNING
  - LOG_NOTICE
  - LOG_INFO
  - LOG_DEBUG
- **$args['event_class']** (*string*) – Class of Event. This is one of:

  Auth Authentication-related Events.

  Block Blocking Events.

  Comment Comment-related Events.

  XMLRPC XML-RPC-related Events.

  Password Password-related Events.

  REST REST API-related Events.

  Spam Spam-related Events.
- **$args['event_id']** (*int*) – Event ID - 16 bits you may do with as you please.
- **$args['message']** (*string*) – Message with substitutions. Note that ” from <*IP*>” is appended.
- **$args['vars']** (*string[]*) – An array of substitutions mapped to regular expressions.

  When logging a message the substitutions are checked and substituted if present. The regex will be used to generate a matching rule for *fail2ban*.

Throws

- **InvalidArgumentException** – Missing entry or invalid type. The `message` will give details.
- **UnexpectedValueException** – Invalid value. The `message` will say which.

Example

```php
$args = [
    'slug'        => 'my-plugin-msg-slug-1',
    'fail'        => 'hard',
    'priority'    => LOG_NOTICE,
    'event_class' => 'Password',
    'event_id'    => 0x001F,
    'message'     => 'Message with ___VAR1___ and ___VAR2___',
    'vars'        => [
        'VAR1' => '\d+',
        'VAR2' => '*.'
    ]
];
try {
    do_action('wp_fail2ban_register_message', 'my-plugin-slug', $args);
} catch(\InvalidArgumentException $e) {
    // Missing entry or invalid type
} catch(\UnexpectedValueException $e) {
    // Invalid value
}
```

Log Message

do_action (*string $action*, *string $plugin_slug*, *string $message_slug*, *array $vars*) → void

Parameters

- **$action** (*string*) – Must be `wp_fail2ban_log_message`.
- **$plugin_slug** (*string*) – The plugin slug used in *Register Plugin*.
- **$message_slug** (*string*) – The message slug used in *Register Message*.
- **$vars** (*array*) – The variable substitutions registered with the message.

Throws **InvalidArgumentException** – Plugin or message not registered.

Example

```php
function myplugin_foobar()
{
    $vars = [
        'VAR1' => 12345,
        'VAR2' => 'xyz'
    ];
    do_action(
        'wp_fail2ban_log_message',
        'my-plugin-slug',
```

(continues on next page)

(continued from previous page)

```
        'my-plugin-msg-slug-1',
        $vars
    );
}
```

Design

To allow 3rd-party plugins to add support for *WPf2b* more easily, the API uses actions. This avoids the need to check if *WPf2b* is installed, then import a file, check for versions, and so on. Integration code can be written that will work if *WPf2b* is installed, and do nothing if not.

Note: Because `do_action` has no return value *WPf2b* will throw an Exception if there is an error.

8.1.2 Example

```
/**
 *
 */
function myplugin_wpf2b_register()
{
    // Register the plugin
    try {
        do_action(
            'wp_fail2ban_register_plugin',
            'my-plugin-slug',
            'My Plugin Name'
        );
    } catch(\LengthException $e) {
        // slug or name too long
    } catch(\RuntimeException $e) {
        // database error
    }

    // Register a message
    $args = [
        'slug'        => 'my-plugin-msg-slug-1',
        'fail'        => 'hard',
        'priority'    => LOG_NOTICE,
        'event_class' => 'Password',
        'event_id'    => 0x001F,
        'message'     => 'Message with ___VAR1___ and ___VAR2___',
        'vars'        => [
            'VAR1' => '\d+',
            'VAR2' => '*.'
        ]
    ];
    try {
        do_action(
            'wp_fail2ban_register_message',
            'my-plugin-slug',
            $args
```

(continues on next page)

(continued from previous page)

```
        );
    } catch(\InvalidArgumentException $e) {
        // Missing entry or invalid type
    } catch(\UnexpectedValueException $e) {
        // Invalid value
    }
}
add_action(
    'wp_fail2ban_register',
    __NAMESPACE__.'\myplugin_wpf2b_register'
);

/**
 *
 */
function myplugin_foobar()
{
    $vars = [
        'VAR1' => 12345,
        'VAR2' => 'xyz'
    ];
    do_action(
        'wp_fail2ban_log_message',
        'my-plugin-slug',
        'my-plugin-msg-slug-1',
        $vars
    );
}
```

8.2 Events

New in version 5.0.0: Add event actions.

8.2.1 EventData Class

```
final class EventData implements \ArrayAccess, \Iterator, \Countable
{
    /**
     * Database fields; read-only
     */
    int     $blog_id;
    int     $event;
    string  $ipv6;
    ?string $username;
    ?string $password;
    ?int    $ref_id;
    ?string $iso;
    ?int    $plugin;
    ?string $request_method;
    ?string $url;
    ?string $content_type;
    ?string $referer;
```

(continues on next page)

(continued from previous page)

```
    ?string $user_agent;
    ?string $post;
    ?string $headers;

    /**
     * Getters
     */
    function getBlogId(): int;
    function getEventId(): int;
    function getIp(): string;
    function getUsername(): ?string;
    function getPassword(): ?string;
    function getRefId(): ?int;
    function getIsoCountryCode(): ?string;
    function getPluginId(): ?int;
    function getRequestMethod(): ?string;
    function getUrl(): ?string;
    function getContentType(): ?string;
    function getReferer(): ?string;
    function getUserAgent(): ?string;
    function getPostData(): ?string;
    function getHttpHeaders(): ?string;
}
```

final class org\lecklider\charles\wordpress\wp_fail2ban\premium**EventData**

WPf2b Event data.

property $blog_id → int
Database field: `blog_id`

property $event → int
Database field: `event`

property $ipv6 → string
Database field: `ipv6`

property $username → ?string
Database field: `username`

property $password → ?string
Database field: `password`

property $ref_id → ?int
Database field: `ref_id`

property $iso → ?string
Database field: `iso`

property $plugin → ?int
Database field: `plugin`

property $request_method → ?string
Database field: `request_method`

property $url → ?string
Database field: `url`

property $content_type → ?string
Database field: `content_type`

property $referer → ?string
Database field: `referer`

property $user_agent → ?string
Database field: `user_agent`

property $post → ?string
Database field: `post`

property $headers → ?string
Database field: `headers`

public getBlogId() → int
Get the ID of the blog that generated the event.

Returns The Blog ID as an integer.

public getEventId() → int
Get the ID of the Event.

Returns Returns the Event ID as an integer.

public getIp() → string
Get the IP address of the host that caused the Event.

Returns Returns the IP address as a string.

public getUsername() → ?string
Get the username used to trigger the Event. Set by:

- *WPF2B_EVENT_AUTH_BLOCK_USER*
- *WPF2B_EVENT_AUTH_BLOCK_USERNAME_LOGIN*
- *WPF2B_EVENT_AUTH_FAIL*
- *WPF2B_EVENT_AUTH_OK*
- *WPF2B_EVENT_PASSWORD_REQUEST*

Returns The username as a string, or `null` if not set.

public getPassword() → ?string
Get the password used to trigger the Event. Set by;

- *WPF2B_EVENT_AUTH_BLOCK_USER*
- *WPF2B_EVENT_AUTH_BLOCK_USERNAME_LOGIN*
- *WPF2B_EVENT_AUTH_FAIL*

Returns The password as a string, or `null` if not set.

public getRefId() → ?int
Get the referenced ID for the Event. Set by:

- *WPF2B_EVENT_COMMENT_CLOSED*
- *WPF2B_EVENT_COMMENT_DRAFT*
- *WPF2B_EVENT_COMMENT_NOT_FOUND*
- *WPF2B_EVENT_COMMENT_PASSWORD*
- *WPF2B_EVENT_COMMENT_SPAM*

- *WPF2B_EVENT_COMMENT_TRASH*
- *WPF2B_EVENT_COMMENT*

Returns The Reference ID as an integer, or `null` if not set.

public getIsoCountryCode() → ?string
Get the 2-letter ISO country code for the Event.

Returns The country code as a string, or `null` if unknown.

public getPluginId() → ?int
Get the registered plugin ID. See *Register Plugin*.

Returns The plugin ID as an integer, or `null` for core Events.

public getRequestMethod() → ?string
Get the HTTP Request Method for the Event. See *WP_FAIL2BAN_EX_LOG_URL*.

Returns The request method as a string, or `null` if URL logging is not enabled.

public getUrl() → ?string
Get the HTTP URL for the Event. See *WP_FAIL2BAN_EX_LOG_URL*.

Returns The URL as a string, or `null` if URL logging is not enabled.

public getContentType() → ?string
Get the HTTP Content Type for the Event. See *WP_FAIL2BAN_EX_LOG_POST_DATA*.

Returns The content type as a string, or `null` if POST data logging is not enabled.

public getReferer() → ?string
Get the HTTP Referer for the Event. See *WP_FAIL2BAN_EX_LOG_REFERER*.

Returns The Referer as a string, or `null` if Referer logging is not enabled.

public getUserAgent() → ?string
Get the HTTP User Agent for the Event. See *WP_FAIL2BAN_EX_LOG_USER_AGENT*.

Returns The user agent as a string, or `null` if User Agent logging is not enabled.

public getPostData() → ?string
Get the HTTP POST data for the Event. See *WP_FAIL2BAN_EX_LOG_POST_DATA*.

Returns The POST data as a string, or `null` if POST data logging is not enabled.

public getHttpHeaders() → ?string
Get the HTTP headers for the Event. See *WP_FAIL2BAN_EX_LOG_HEADERS*.

Returns The HTTP headers as a string, or `null` if header logging is not enabled.

CHAPTER 9

Release Notes

9.1 4.3.0

- Add new dashboard widget: last 5 *syslog* messages.
- Add full multisite support.
- Add username login blocking (force login with email).
- Add separate logging for login attempts with an empty username.
- Improve user enumeration blocking compatibility with the WordPress block editor (Gutenberg).
- Bump the minimum PHP version to 5.6.

9.1.1 Patches

4.3.0.1

Premium Only

- Fix issue when *WP_FAIL2BAN_BLOCK_USERNAME_LOGIN* enabled and *WP_FAIL2BAN_BLOCKED_USERS* not configured.

4.3.0.2

Premium Only

- Fix issue where some events weren't logged.

4.3.0.3

Premium Only

- Fix incorrect total for Event Log.
- Fix database renumber for Pingbacks.

4.3.0.4

- Fix plugin event registration.
- Add colour to "Last 5 Messages" dashboard widget.

4.3.0.5

- Fix empty username detection for multisite.
- Fix harmless warning when activating new multisite install.
- Fix esoteric edge-case where *wp-load.php* is loaded via a script run from the CLI in a directory with a *functions.php* file.

4.3.0.6

- Fix Forbidden error on Posts page for roles below Editor when user enum blocking enabled. [WordPress only]

4.3.0.7

- Finish refactoring to allow inclusion of constants in *wp-config.php* (h/t @iCounsellor).

Premium Only

- Fix MaxMind database update.

9.1.2 Upgrade

To take advantage of the new features you will need up update your *fail2ban* filters; existing filters will continue to work as before.

Premium Users

Please backup your database before upgrading.

4.3.0.7

Premium Users

Please update your MaxMind database.

CHAPTER 10

define() Constants

10.1 All

10.1.1 WP_FAIL2BAN_AUTH_LOG

Facility for Auth class events.

Default: `LOG_AUTH` or `LOG_AUTHPRIV`

New in version 2.2.0.

Changed in version 4.4.0: Uses *WP_FAIL2BAN_USE_AUTHPRIV*

Listing 1: Example: Using LOG_LOCAL5

```
/**
 * Facility for Auth class events.
 */
define('WP_FAIL2BAN_AUTH_LOG', LOG_LOCAL5);
```

See also:

- *WP_FAIL2BAN_USE_AUTHPRIV*
- *Auth Events*

10.1.2 WP_FAIL2BAN_BLOCKED_USERS

Block login for specified usernames.

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.

WPf2b allows you to specify a regex that will shortcut the login process if the requested username matches.

For example,

Listing 2: Example: regex

```
/**
 * Block logic
 */
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 or later you can specify an array of users instead:

Listing 3: Example: Array of usernames

```
/**
 * Block login
 */
define('WP_FAIL2BAN_BLOCKED_USERS', ['admin', 'another', 'user']);
```

History

Based on a suggestion from *@jmadea*.

10.1.3 WP_FAIL2BAN_BLOCK_USERNAME_LOGIN

Force login with email address/prevent login with username.

Default setting: *disabled*

New in version 4.3.0.

```
/**
 * Force login with email address/prevent login with username.
 */
define('WP_FAIL2BAN_BLOCK_USERNAME_LOGIN', true);
```

Important: You should define this in `wp-config.php` even if you are using the Premium version of *WPf2b*.

See also:

- *WP_FAIL2BAN_BLOCK_USER_ENUMERATION*

10.1.4 WP_FAIL2BAN_BLOCK_USER_ENUMERATION

Block user enumeration.

Default setting: *disabled*

New in version 2.1.0.

Changed in version 4.0.0: Now also blocks enumeration via the REST API.

```
/**
 * Block user enumeration.
 */
define('WP_FAIL2BAN_BLOCK_USER_ENUMERATION', true);
```

Important: You should define this in `wp-config.php` even if you are using the Premium version of *WPf2b*.

Warning: If your theme has Author profile pages (e.g. TwentyTwenty) you will need to *block username logins* instead.

History

Based on a suggestion from *@geeklol* and a plugin by *@ROIBOT*.

See also:

- *WP_FAIL2BAN_BLOCK_USERNAME_LOGIN*

10.1.5 WP_FAIL2BAN_COMMENT_ATTEMPT_LOG

Facility for attempted comment events.

Default: `LOG_AUTH` or `LOG_AUTHPRIV`

New in version 5.0.0.

Listing 4: Example: Using LOG_LOCAL5

```
/**
 * Facility for attempted comment events.
 */
define('WP_FAIL2BAN_COMMENT_ATTEMPT_LOG', LOG_LOCAL5);
```

See also:

- *WP_FAIL2BAN_LOG_COMMENT_ATTEMPTS*
- *WP_FAIL2BAN_USE_AUTHPRIV*

10.1.6 WP_FAIL2BAN_COMMENT_EXTRA_LOG

Facility for extra comment events.

Default: `LOG_AUTH` or `LOG_AUTHPRIV`

New in version 4.0.5.

Changed in version 4.4.0: Uses *WP_FAIL2BAN_USE_AUTHPRIV*

Deprecated since version 5.0.0.

Listing 5: Example: Using LOG_LOCAL5

```
/**
 * Facility for extra comment events.
 */
define('WP_FAIL2BAN_COMMENT_EXTRA_LOG', LOG_LOCAL5);
```

See also:

- *WP_FAIL2BAN_LOG_COMMENT_ATTEMPTS*
- *WP_FAIL2BAN_LOG_COMMENTS_EXTRA*
- *WP_FAIL2BAN_USE_AUTHPRIV*

10.1.7 WP_FAIL2BAN_COMMENT_LOG

Facility for Comment class events.

Default facility: `LOG_USER`

New in version 3.5.0.

Listing 6: Example: Using LOG_LOCAL3

```
/**
 * Facility for Comment events.
 */
define('WP_FAIL2BAN_COMMENT_LOG', LOG_LOCAL3);
```

See also:

- *WP_FAIL2BAN_LOG_COMMENTS*
- *WP_FAIL2BAN_LOG_COMMENTS_EXTRA*
- *Facilities*

10.1.8 WP_FAIL2BAN_DISABLE_LAST_LOG

Disable logging last event messages.

Default setting: `false`

New in version 4.3.0.

WPf2b v4.3.0 introduced a new dashboard widget to display the last 5 `syslog` messages.

These messages are stored in the options table; for most sites this won't be an issue, but, if you're already doing a lot of updates to the options table or have some other esoteric configuration, you might want to disable this feature:

```
/**
 * Disable logging last event messages.
 */
define('WP_FAIL2BAN_DISABLE_LAST_LOG', true);
```

10.1.9 WP_FAIL2BAN_EX_BLOCK_COUNTRIES

Default setting: *disabled*

Premium only

New in version 4.3.2.0.

10.1.10 WP_FAIL2BAN_EX_BLOCK_COUNTRIES_LOG

Facility for blocked country event

Default facility: `LOG_USER`

Premium only

New in version 4.3.2.0.

Listing 7: Example: Using LOG_LOCAL2

```
/**
 * Facility for blocked country event.
 */
define('WP_FAIL2BAN_EX_BLOCK_COUNTRIES_LOG', LOG_LOCAL2);
```

10.1.11 WP_FAIL2BAN_EX_LOG_HEADERS

Default setting: *disabled*

Premium only

New in version 4.3.0.

Lorem

```
define('WP_FAIL2BAN_EX_LOG_HEADERS', true);
```

10.1.12 WP_FAIL2BAN_EX_LOG_POST_DATA

Default setting: ***disabled***

Premium only

New in version 4.3.0.

Lorem

```
define('WP_FAIL2BAN_EX_LOG_POST_DATA', true);
```

10.1.13 WP_FAIL2BAN_EX_LOG_REFERER

Default setting: ***disabled***

Premium only

New in version 4.3.0.

Lorem

```
define('WP_FAIL2BAN_EX_LOG_REFERER', true);
```

10.1.14 WP_FAIL2BAN_EX_LOG_URL

Default setting: ***disabled***

Premium only

New in version 4.3.0.

Lorem

```
define('WP_FAIL2BAN_EX_LOG_URL', true);
```

10.1.15 WP_FAIL2BAN_EX_LOG_USER_AGENT

Default setting: ***disabled***

Premium only

New in version 4.3.0.

Lorem

```
define('WP_FAIL2BAN_EX_LOG_USER_AGENT', true);
```

10.1.16 WP_FAIL2BAN_EX_MAXMIND_LICENSE

Premium only

New in version 4.3.0.

Lorem

```
define('WP_FAIL2BAN_EX_MAXMIND_LICENSE', true);
```

10.1.17 WP_FAIL2BAN_EX_PROXY_CLOUDFLARE

Premium only

New in version 4.3.2.0.

10.1.18 WP_FAIL2BAN_EX_WAF

Control the state of the WAF.

Default setting: *disabled*

Premium only

New in version 5.1.0.

The state can be one of:

on Enabled; blocks detected threats.

off Disabled.

logging Detects and logs threats.

Listing 8: Example: Enabling logging only

```
/**
 * WAF state.
 */
define('WP_FAIL2BAN_EX_WAF', 'logging');
```

10.1.19 WP_FAIL2BAN_EX_WAF_LOG

Facility for WAF class events.

Default facility: `LOG_USER`

Premium only

New in version 5.1.0.

Listing 9: Example: Using LOG_LOCAL5

```
/**
 * Facility for WAF events.
 */
define('WP_FAIL2BAN_EX_WAF_LOG', LOG_LOCAL5);
```

10.1.20 WP_FAIL2BAN_EX_WAF_SQLI_PLUGINS

Check plugin queries for SQLi.

Default setting: *disabled*

Premium only

New in version 5.1.0.

Listing 10: Example: Enabling SQLi detection for plugins

```
/**
 * WAF: check plugin queries for SQLi.
 */
define('WP_FAIL2BAN_EX_WAF_SQLI_PLUGINS', true);
```

10.1.21 WP_FAIL2BAN_EX_WAF_SQLI_WORDPRESS

Check WordPress core queries for SQLi.

Default setting: *disabled*

Premium only

New in version 5.1.0.

Note: This setting exists for testing; it is published for completeness.

Listing 11: Example: Enabling SQLi detection for WordPress core

```
/**
 * WAF: check WordPress core queries for SQLi.
 */
define('WP_FAIL2BAN_EX_WAF_SQLI_WORDPRESS', true);
```

Warning: Do not enable this in normal operation without good technical justification.

10.1.22 WP_FAIL2BAN_EX_WAF_UPDATE_OPTION

Check that current user may update core WordPress options.

Default setting: *disabled*

Premium only

New in version 5.1.0.

When a plugin tries to update a core WordPress option, check the current user has `update_options` or `update_network_options` capabilities.

Listing 12: Example: Enabling caps checking for update_option() on core WordPress options.

```
/**
 * WAF: check caps for update_option().
 */
define('WP_FAIL2BAN_EX_WAF_UPDATE_OPTION', true);
```

10.1.23 WP_FAIL2BAN_EX_XMLRPC_BLOCKED

Premium only

New in version 4.3.2.0.

10.1.24 WP_FAIL2BAN_EX_XMLRPC_JETPACK

Premium only

New in version 4.3.2.0.

10.1.25 WP_FAIL2BAN_EX_XMLRPC_LOG

Facility for XML-RPC class events.

Default facility: `LOG_USER`

Premium only

New in version 4.3.2.0.

Listing 13: Example: Using LOG_LOCAL7

```
/**
 * Facility for XML-RPC events.
 */
define('WP_FAIL2BAN_EX_WAF_LOG', LOG_LOCAL7);
```

10.1.26 WP_FAIL2BAN_EX_XMLRPC_TRUSTED_IPS

Premium only

New in version 4.3.2.0.

10.1.27 WP_FAIL2BAN_FREE_ONLY

New in version 4.4.0.

Default setting: `false`

Hide Freemius interface:

```
define('WP_FAIL2BAN_FREE_ONLY', true);
```

10.1.28 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.

10.1.29 WP_FAIL2BAN_INSTALL_PATH

New in version 5.0.0.

The path to the `fail2ban` install.

The Site Health tool looks in the following locations:

- `/etc/fail2ban`
- `/usr/local/etc/fail2ban`

If your `fail2ban` install lives elsewhere you should define it in `wp-config.php`:

```
define('WP_FAIL2BAN_INSTALL_PATH', '/var/fail2ban');
```

10.1.30 WP_FAIL2BAN_LOG_COMMENTS

Log submitted comments.

Default setting: *disabled*

New in version 3.5.0.

```
/**
 * Log submitted comments.
 */
define('WP_FAIL2BAN_LOG_COMMENTS', true);
```

The comment ID and IP will be written to *WP_FAIL2BAN_COMMENT_LOG* and matched by *wordpress-extra.conf*.

See also:

- *WP_FAIL2BAN_COMMENT_LOG*

10.1.31 WP_FAIL2BAN_LOG_COMMENTS_EXTRA

Log extra comment events.

New in version 4.0.0.

Deprecated since version 5.0.0: See *WP_FAIL2BAN_LOG_COMMENT_ATTEMPTS*

WPf2b can optionally log the following comment-related events:

Not found Attempted comment on a non-existent post

WPF2B_EVENT_COMMENT_NOT_FOUND

Closed Attempted comment on a post with closed comments

WPF2B_EVENT_COMMENT_CLOSED

Trash Attempted comment on a post in Trash

WPF2B_EVENT_COMMENT_TRASH

Draft Attempted comment on a Draft post

WPF2B_EVENT_COMMENT_DRAFT

Password-protected Attempted comment on a password-protected post

WPF2B_EVENT_COMMENT_PASSWORD

To enable this feature OR the event constants.

Listing 14: Example: enable *Closed* and *Draft*

```
/**
 * Log comments on 'Closed' and 'Draft' posts
 */
define('WP_FAIL2BAN_LOG_COMMENTS_EXTRA', WPF2B_EVENT_COMMENT_CLOSED | WPF2B_EVENT_
↪COMMENT_DRAFT);
```

You **must** also load the constants *before* trying to use them. In *wp-config.php* add:

```
include __DIR__.'/wp-content/plugins/wp-fail2ban/lib/constants.php';
```

or for the Premium version:

```
include __DIR__.'/wp-content/plugins/wp-fail2ban-premium/lib/constants.php';
```

If you have non-standard paths, e.g. plugins in a different place, you'll need to adjust the *include* path to suit.

The Post ID and IP will be written to *WP_FAIL2BAN_COMMENT_LOG* and matched by *wordpress-extra.conf*.

10.1.32 WP_FAIL2BAN_LOG_COMMENT_ATTEMPTS

Log attempted comments.

Default setting: *disabled*

New in version 5.0.0.

WPf2b can optionally log the following comment-related events:

Not found	Attempted comment on a non-existent post.
Closed	Attempted comment on a post with closed comments.
Trash	Attempted comment on a post in Trash.
Draft	Attempted comment on a Draft post.
Password-protected	Attempted comment on a password-protected post.

```
/**
 * Log attempted comments.
 */
define('WP_FAIL2BAN_LOG_COMMENT_ATTEMPTS', true);
```

The comment ID and IP will be written to *WP_FAIL2BAN_COMMENT_ATTEMPT_LOG* and matched by *wordpress-soft.conf*.

See also:

- *WP_FAIL2BAN_COMMENT_ATTEMPT_LOG*

10.1.33 WP_FAIL2BAN_LOG_PASSWORD_REQUEST

Log password reset requests.

Default setting: *disabled*

New in version 3.5.0.

```
/**
 * Log password reset requests.
 */
define('WP_FAIL2BAN_LOG_PASSWORD_REQUEST', true);
```

The username and IP will be written to *WP_FAIL2BAN_PASSWORD_REQUEST_LOG* and matched by *wordpress-extra.conf*.

10.1.34 WP_FAIL2BAN_LOG_PINGBACKS

Log pingbacks.

Default setting: *disabled*

New in version 2.2.0.

```
/**
 * Log pingbacks.
 */
define('WP_FAIL2BAN_LOG_PINGBACKS', true);
```

History

Based on a suggestion from *@maghe*.

See also:

- *WP_FAIL2BAN_PINGBACK_LOG*

10.1.35 WP_FAIL2BAN_LOG_SPAM

Log comments marked as spam.

Default setting: *disabled*

New in version 3.5.0.

```
/**
 * Log spam comments.
 */
define('WP_FAIL2BAN_LOG_SPAM', true);
```

The comment ID and IP will be written to *WP_FAIL2BAN_SPAM_LOG* and matched by *wordpress-hard.conf*.

See also:

- *WP_FAIL2BAN_SPAM_LOG*

10.1.36 WP_FAIL2BAN_OPENLOG_OPTIONS

New in version 3.5.0.

This allows you to change the advanced syslog connection parameters.

If you know you need this you'll know the options you need and what they do. If you don't, you won't.

If in doubt, leave this setting alone.

10.1.37 WP_FAIL2BAN_PASSWORD_REQUEST_LOG

Facility for logging password reset events.

Default facility: `LOG_USER`

New in version 4.0.0.

Listing 15: Example: Using LOG_LOCAL3

```
 /**
  * Facility for logging password reset events.
  */
define('WP_FAIL2BAN_PASSWORD_REQUEST_LOG', LOG_LOCAL3);
```

See also:

- *WP_FAIL2BAN_LOG_PASSWORD_REQUEST*

10.1.38 WP_FAIL2BAN_PINGBACK_ERROR_LOG

New in version 4.0.5: Reserved for future use.

```
define('WP_FAIL2BAN_PINGBACK_ERROR_LOG', LOG_LOCAL3);
```

10.1.39 WP_FAIL2BAN_PINGBACK_LOG

Facility for logging pingbacks.

Default facility: `LOG_USER`

New in version 2.2.0.

Listing 16: Example: Using LOG_LOCAL3

```
/**
 * Facility for logging pingbacks.
 */
define('WP_FAIL2BAN_PINGBACK_LOG', LOG_LOCAL3);
```

See also:

- *WP_FAIL2BAN_LOG_PINGBACKS*
- *Facilities*

10.1.40 WP_FAIL2BAN_PLUGIN_AUTH_LOG

Facility for "Auth" class plugin events.

Default: `LOG_AUTH` or `LOG_AUTHPRIV`

New in version 4.2.0.

Changed in version 4.4.0: Uses *WP_FAIL2BAN_USE_AUTHPRIV*

Listing 17: Example: Using LOG_LOCAL5

```
/**
 * Facility for "Auth" class plugin events.
 */
define('WP_FAIL2BAN_PLUGIN_AUTH_LOG', LOG_LOCAL5);
```

See also:

- *WP_FAIL2BAN_PLUGIN_LOG_AUTH*
- *Facilities*

10.1.41 WP_FAIL2BAN_PLUGIN_COMMENT_LOG

Facility for "Comment" class plugin events.

Default facility: `LOG_USER`

New in version 4.2.0.

Listing 18: Example: Using LOG_LOCAL3

```
/**
 * Facility for "Comment" class plugin events.
 */
define('WP_FAIL2BAN_PLUGIN_COMMENT_LOG', LOG_LOCAL3);
```

See also:

- *WP_FAIL2BAN_PLUGIN_LOG_COMMENT*
- *Facilities*

10.1.42 WP_FAIL2BAN_PLUGIN_LOG_AUTH

Enable logging plugin "Auth" class events.

Default setting: *disabled*

New in version 4.2.0.

```
/**
 * Enable logging plugin "Auth" class events.
 */
define('WP_FAIL2BAN_PLUGIN_LOG_AUTH', true);
```

See also:

- *WP_FAIL2BAN_PLUGIN_AUTH_LOG*
- *WP_FAIL2BAN_USE_AUTHPRIV*

10.1.43 WP_FAIL2BAN_PLUGIN_LOG_COMMENT

Enable logging plugin "Comment" class events.

Default setting: *disabled*

New in version 4.2.0.

```
/**
 * Enable logging plugin "Comment" class events.
 */
define('WP_FAIL2BAN_PLUGIN_LOG_COMMENT', true);
```

See also:

- *WP_FAIL2BAN_PLUGIN_COMMENT_LOG*

10.1.44 WP_FAIL2BAN_PLUGIN_LOG_OTHER

Enable logging plugin "Other" class events.

Default setting: *disabled*

New in version 4.2.0.

```
/**
 * Enable logging plugin "Other" class events.
 */
define('WP_FAIL2BAN_PLUGIN_LOG_OTHER', true);
```

See also:

- *WP_FAIL2BAN_PLUGIN_OTHER_LOG*

10.1.45 WP_FAIL2BAN_PLUGIN_LOG_PASSWORD

Enable logging plugin "Password" class events.

Default setting: *disabled*

New in version 4.2.0.

```
/**
 * Enable logging plugin "Password" class events.
 */
define('WP_FAIL2BAN_PLUGIN_LOG_PASSWORD', true);
```

See also:

- *WP_FAIL2BAN_PLUGIN_PASSWORD_LOG*

10.1.46 WP_FAIL2BAN_PLUGIN_LOG_REST

Enable logging plugin "REST" class events.

Default setting: *disabled*

New in version 4.2.0.

```
/**
 * Enable logging plugin "REST" class events.
 */
define('WP_FAIL2BAN_PLUGIN_LOG_REST', true);
```

See also:

- *WP_FAIL2BAN_PLUGIN_REST_LOG*

10.1.47 WP_FAIL2BAN_PLUGIN_LOG_SPAM

Enable logging plugin "Spam" class events.

Default setting: *disabled*

New in version 4.2.0.

```
/**
 * Enable logging plugin "Spam" class events.
 */
define('WP_FAIL2BAN_PLUGIN_LOG_SPAM', true);
```

See also:

- *WP_FAIL2BAN_PLUGIN_SPAM_LOG*

10.1.48 WP_FAIL2BAN_PLUGIN_LOG_XMLRPC

Enable logging plugin "XML-RPC" class events.

Default setting: *disabled*

New in version 4.2.0.

```
/**
 * Enable logging plugin "XML-RPC" class events.
 */
define('WP_FAIL2BAN_PLUGIN_LOG_XMLRPC', true);
```

See also:

- *WP_FAIL2BAN_PLUGIN_XMLRPC_LOG*

10.1.49 WP_FAIL2BAN_PLUGIN_OTHER_LOG

Facility for "Other" class plugin events.

Default facility: `LOG_USER`

New in version 4.2.0.

Listing 19: Example: Using LOG_LOCAL3

```
/**
 * Facility for "Other" class plugin events.
 */
define('WP_FAIL2BAN_PLUGIN_OTHER_LOG', LOG_LOCAL3);
```

See also:

- *WP_FAIL2BAN_PLUGIN_LOG_OTHER*
- *Facilities*

10.1.50 WP_FAIL2BAN_PLUGIN_PASSWORD_LOG

Facility for “Password” class plugin events.

Default facility: `LOG_USER`

New in version 4.2.0.

Listing 20: Example: Using LOG_LOCAL3

```
/**
 * Facility for "Password" class plugin events.
 */
define('WP_FAIL2BAN_PLUGIN_PASSWORD_LOG', LOG_LOCAL3);
```

See also:

- *WP_FAIL2BAN_PLUGIN_LOG_PASSWORD*
- *Facilities*

10.1.51 WP_FAIL2BAN_PLUGIN_REST_LOG

Facility for “REST” class plugin events.

Default facility: `LOG_USER`

New in version 4.2.0.

Listing 21: Example: Using LOG_LOCAL3

```
/**
 * Facility for "REST" class plugin events.
 */
define('WP_FAIL2BAN_PLUGIN_REST_LOG', LOG_LOCAL3);
```

See also:

- *WP_FAIL2BAN_PLUGIN_LOG_REST*
- *Facilities*

10.1.52 WP_FAIL2BAN_PLUGIN_SPAM_LOG

Facility for "Spam" class plugin events.

Default: `LOG_AUTH` or `LOG_AUTHPRIV`

New in version 4.2.0.

Changed in version 4.4.0: Uses *WP_FAIL2BAN_USE_AUTHPRIV*

Listing 22: Example: Using LOG_LOCAL5

```
/**
 * Facility for Spam class plugin events.
 */
define('WP_FAIL2BAN_PLUGIN_SPAM_LOG', LOG_LOCAL5);
```

See also:

- *WP_FAIL2BAN_PLUGIN_LOG_SPAM*
- *Facilities*

10.1.53 WP_FAIL2BAN_PLUGIN_XMLRPC_LOG

Facility for "XML-RPC" class plugin events.

Default facility: `LOG_USER`

New in version 4.2.0.

Listing 23: Example: Using LOG_LOCAL5

```
/**
 * Facility for XML-RPC class events.
 */
define('WP_FAIL2BAN_PLUGIN_XMLRPC_LOG', LOG_LOCAL5);
```

See also:

- *WP_FAIL2BAN_PLUGIN_LOG_XMLRPC*
- *Facilities*

10.1.54 WP_FAIL2BAN_PROXIES

New in version 2.0.0.

Changed in version 4.0.0: Entries can be ignored by prefixing with #

Changed in version 5.0.0: Entries can include IPv6 addresses. Added "Unknown Proxy in X-Forwarded-For" message.

A list of IP addresses for the trusted proxies that will appear as the remote IP for a request. When defined:

- If the remote address appears in the **WP_FAIL2BAN_PROXIES** list, *WPf2b* will use the IP address from the *X-Forwarded-For* header
- If the remote address does not appear in the **WP_FAIL2BAN_PROXIES** list and there is an *X-Forwarded-For* header, *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'
]);
```

Premium

The list is processed and cached for performance. Updating the list from the UI will automatically clear the cache, but you must do so manually if you are using a `define()`.

See also:

- *Clearing the Cache*

10.1.55 WP_FAIL2BAN_REMOTE_ADDR

IP address to use for anonymised requests.

Default setting: *disabled*

New in version 3.6.0.

Some themes and plugins anonymise requests; I'm sure there's a good reason.

```
/*
 * IP address to use for anonymised requests.
 */
define('WP_FAIL2BAN_REMOTE_ADDR', '172.16.123.123');
```

Attention: You **must** define this in `wp-config.php` even if you are using the Premium version of *WPf2b*.

10.1.56 WP_FAIL2BAN_SITE_HEALTH_SKIP_FILTERS

Ignore filter files during Health Check.

New in version 5.0.0.

Default setting: *disabled*

WPf2b uses the WordPress Site Heath tool to check for *obsolete* and *modified* filter files.

However, this test will not work with many server configurations, e.g. if PHP is using `chroot`. In that case you should disable these checks to give you cleaner output from the Site Health tool (they're otherwise harmless).

In `wp-config.php`:

```
/*
 * Ignore filter files during Health Check.
 */
define('WP_FAIL2BAN_SITE_HEALTH_SKIP_FILTERS', true);
```

Warning: **It is your responsibility to ensure your filters are kept current.**

10.1.57 WP_FAIL2BAN_SPAM_LOG

Facility for Spam class events.

Default: `LOG_AUTH` or `LOG_AUTHPRIV`

New in version 4.0.0.

Changed in version 4.4.0: Uses *WP_FAIL2BAN_USE_AUTHPRIV*

Listing 24: Example: Using LOG_LOCAL4

```
/*
 * Facility for Spam class events.
 */
define('WP_FAIL2BAN_SPAM_LOG', LOG_LOCAL4);
```

See also:

- *WP_FAIL2BAN_USE_AUTHPRIV*
- *Spam Events*

10.1.58 WP_FAIL2BAN_SYSLOG_SHORT_TAG

Use short tag for syslog.

Default setting: *disabled*

New in version 3.0.0.

Uses `wp` instead of `wordpress`.

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

See also:

- *WP_FAIL2BAN_HTTP_HOST*
- *WP_FAIL2BAN_TRUNCATE_HOST*

10.1.59 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.

10.1.60 WP_FAIL2BAN_USE_AUTHPRIV

Use AUTHPRIV by default.

Default setting: *disabled*

New in version 4.4.0.

By default, *WPf2b* uses **LOG_AUTH** for logging various events. 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`:

```
/**
 * Use AUTHPRIV
 */
define('WP_FAIL2BAN_USE_AUTHPRIV', true);
```

Note: This only changes the **default** use of **LOG_AUTH** - it doesn't override individual settings.

Attention: You **must** define this in `wp-config.php` even if you are using the Premium version of *WPf2b*.

See also:

- *Logfile Reference*

10.1.61 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');
```

10.2 Logging

10.2.1 Premium

10.2.2 Deprecated

10.3 syslog

10.4 Block

10.4.1 Premium

10.5 Remote IPs

10.5.1 Premium

10.6 Plugins

10.7 Site Health

10.8 WAF

10.8.1 Premium

10.9 Miscellaneous

10.10 Development

10.11 Reserved

CHAPTER 11

Facilities

While the full list of facilities is reproduced here for completeness, using anything but **LOG_AUTH**, **LOG_AUTHPRIV**, and/or **LOG_LOCAL0**..**7** is unlikely to have the desired results. **LOG_USER** can be used for Notices, but Info messages are generally not saved.

Facility	Description
LOG_AUTH	security/authorization messages (use LOG_AUTHPRIV instead in systems where that constant is defined)
LOG_AUTHPRIV	security/authorization messages (private)
LOG_CRON	clock daemon (cron and at)
LOG_DAEMON	other system daemons
LOG_KERN	kernel messages
LOG_LOCAL0…7	reserved for local use, these are not available in Windows
LOG_LPR	line printer subsystem
LOG_MAIL	mail subsystem
LOG_NEWS	USENET news subsystem
LOG_SYSLOG	messages generated internally by syslogd
LOG_USER	generic user-level messages
LOG_UUCP	UUCP subsystem

CHAPTER 12

Logfile Reference

OS	Level	LOG_AUTH	LOG_AUTHPRIV	LOG_USER
CentOS 7		*(not used)*	`/var/log/secure`	
FeeBSD	INFO	`/var/log/auth/log`	`/var/log/auth/log`	•
	NOTICE	`/var/log/auth/log`	`/var/log/auth/log`	`/var/log/messages`
Ubuntu 18	(all)	`/var/log/auth.log`	`/var/log/auth.log`	`/var/log/syslog`

CHAPTER 13

Default Facilities

Facility	Define
LOG_AUTH	*WP_FAIL2BAN_AUTH_LOG*
	WP_FAIL2BAN_COMMENT_EXTRA_LOG
	WP_FAIL2BAN_PINGBACK_ERROR_LOG
	WP_FAIL2BAN_PLUGIN_AUTH_LOG
	WP_FAIL2BAN_PLUGIN_SPAM_LOG
	WP_FAIL2BAN_SPAM_LOG
LOG_USER	*WP_FAIL2BAN_COMMENT_LOG*
	WP_FAIL2BAN_PASSWORD_REQUEST_LOG
	WP_FAIL2BAN_PINGBACK_LOG
	WP_FAIL2BAN_PLUGIN_COMMENT_LOG
	WP_FAIL2BAN_PLUGIN_OTHER_LOG
	WP_FAIL2BAN_PLUGIN_PASSWORD_LOG
	WP_FAIL2BAN_PLUGIN_REST_LOG
	WP_FAIL2BAN_PLUGIN_XMLRPC_LOG
	WP_FAIL2BAN_XMLRPC_LOG

13.1 Premium

Facility	Define
LOG_USER	*WP_FAIL2BAN_EX_BLOCK_COUNTRIES_LOG*
	WP_FAIL2BAN_EX_XMLRPC_LOG
	WP_FAIL2BAN_EX_WAF_LOG

CHAPTER 14

Events

Note: Events may belong to more than one class.

14.1 Auth Events

Authentication events. Broadly, anything to do with users.

14.1.1 WPF2B_EVENT_AUTH_OK

Authentication OK.

syslog	Facility	LOG_AUTH or LOG_AUTHPRIV
	Level	INFO
fail2ban	Filter	*n/a*
	Rule	`Accepted password for .* from <HOST>`
EventData	`$username`	Username

New in version 4.0.0.

See also:

WP_FAIL2BAN_USE_AUTHPRIV

14.1.2 WPF2B_EVENT_AUTH_FAIL

Authentication failed.

syslog	Facility	LOG_AUTH or LOG_AUTHPRIV
	Level	NOTICE
fail2ban	Filter	*wordpress-soft.conf*
	Rules	`Authentication failure for .* from <HOST>` `Authentication attempt for unknown user .* from <HOST>`
EventData	`$username`	Username
	`$password`	Password

New in version 4.0.0.

See also:

WP_FAIL2BAN_USE_AUTHPRIV

14.1.3 WPF2B_EVENT_AUTH_EMPTY_USER

Empty Username.

syslog	Facility	LOG_AUTH or LOG_AUTHPRIV
	Level	NOTICE
fail2ban	Filter	*wordpress-soft.conf*
	Rule	`Empty username from <HOST>`

New in version 4.3.0.

See also:

WP_FAIL2BAN_USE_AUTHPRIV

14.1.4 WPF2B_EVENT_AUTH_BLOCK_USER

Blocked username.

syslog	Facility	LOG_AUTH or LOG_AUTHPRIV
	Level	NOTICE
fail2ban	Filter	*wordpress-hard.conf*
	Rule	`Blocked authentication attempt for .* from <HOST>`
EventData	`$username`	Username
	`$password`	Password

New in version 4.3.0.

See also:

WP_FAIL2BAN_USE_AUTHPRIV
WP_FAIL2BAN_BLOCKED_USERS

14.1.5 WPF2B_EVENT_AUTH_BLOCK_USER_ENUM

Blocked user enumeration.

syslog	Facility	LOG_AUTH or LOG_AUTHPRIV
	Level	NOTICE
fail2ban	Filter	*wordpress-hard.conf*
	Rule	`Blocked user enumeration attempt from <HOST>`

New in version 4.3.0.

See also:

WP_FAIL2BAN_USE_AUTHPRIV
WP_FAIL2BAN_BLOCK_USER_ENUMERATION

14.1.6 WPF2B_EVENT_AUTH_BLOCK_USERNAME_LOGIN

Blocked login with username.

syslog	Facility	LOG_AUTH or LOG_AUTHPRIV
	Level	NOTICE
fail2ban	Filter	*wordpress-hard.conf*
	Rule	`Blocked username authentication attempt for .* from <HOST>`
EventData	`$username`	Username
	`$password`	Password

New in version 4.3.0.

See also:

WP_FAIL2BAN_USE_AUTHPRIV
WP_FAIL2BAN_BLOCK_USERNAME_LOGIN

14.1.7 WPF2B_EVENT_REST_AUTH_OK

REST Authentication OK.

Not currently used.

14.1.8 WPF2B_EVENT_REST_AUTH_FAIL

REST Authentication failed.

Not currently used.

14.2 Block Events

Things that have been actively prevented.

14.2.1 WPF2B_EVENT_BLOCK_COUNTRY

Attempted access from a blocked Country.

Premium only

syslog	Facility	LOG_AUTH or LOG_AUTHPRIV
	Level	NOTICE
fail2ban	Filter	*wordpress-hard.conf*
	Rule	`Blocked access from country '..' from <HOST>`

New in version 4.3.0.

See also:

WP_FAIL2BAN_USE_AUTHPRIV

14.2.2 WPF2B_EVENT_XMLRPC_BLOCKED

Blocked RPC-XML request.

Premium only

syslog	Facility	*WP_FAIL2BAN_EX_XMLRPC_LOG*
	Level	NOTICE
fail2ban	Filter	*wordpress-hard.conf*
	Rule	`XML-RPC request blocked from <HOST>`

New in version 4.3.0.

See also:

WP_FAIL2BAN_EX_XMLRPC_BLOCKED

14.3 Comment Events

Anything to do with comments.

14.3.1 WPF2B_EVENT_COMMENT

Comment submitted.

syslog	Facility	*WP_FAIL2BAN_COMMENT_LOG*
	Level	INFO
fail2ban	Filter	*wordpress-extra.conf*
	Rule	`Comment \d+ from <HOST>`
EventData	`$ref_id`	Comment ID

New in version 4.0.0.

14.3.2 WPF2B_EVENT_COMMENT_SPAM

Comment marked as spam.

syslog	Facility	*WP_FAIL2BAN_SPAM_LOG*
	Level	NOTICE
fail2ban	Filter	*wordpress-hard.conf*
	Rule	`Spam comment \d+ from <HOST>`
EventData	`$ref_id`	Comment ID

New in version 4.0.0.

14.3.3 WPF2B_EVENT_COMMENT_SPAM_AKISMET

Comment marked as spam.

syslog	Facility	*WP_FAIL2BAN_SPAM_LOG*
	Level	NOTICE
fail2ban	Filter	*wordpress-hard.conf*
	Rule	`Akismet discarded spam comment from <HOST>`

New in version 5.0.0.

14.3.4 WPF2B_EVENT_COMMENT_NOT_FOUND

Attempted comment on non-existent Post.

syslog	Facility	*WP_FAIL2BAN_COMMENT_EXTRA_LOG*
	Level	NOTICE
fail2ban	Filter	*wordpress-extra.conf*
	Rule	`Comment attempt on non-existent post \d+ from <HOST>`
EventData	`$ref_id`	Post ID

New in version 4.0.0.

14.3.5 WPF2B_EVENT_COMMENT_CLOSED

Attempted comment on closed Post.

syslog	Facility	*WP_FAIL2BAN_COMMENT_EXTRA_LOG*
	Level	NOTICE
fail2ban	Filter	*wordpress-extra.conf*
	Rule	`Comment attempt on closed post \d+ from <HOST>`
EventData	`$ref_id`	Comment ID

New in version 4.0.0.

14.3.6 WPF2B_EVENT_COMMENT_TRASH

Attempted comment on post in Trash.

syslog	Facility	*WP_FAIL2BAN_COMMENT_EXTRA_LOG*
	Level	NOTICE
fail2ban	Filter	*wordpress-extra.conf*
	Rule	`Comment attempt on trash post \d+ from <HOST>`
EventData	`$ref_id`	Comment ID

New in version 4.0.0.

14.3.7 WPF2B_EVENT_COMMENT_DRAFT

Attempted comment on draft Post.

syslog	Facility	*WP_FAIL2BAN_COMMENT_EXTRA_LOG*
	Level	NOTICE
fail2ban	Filter	*wordpress-extra.conf*
	Rule	`Comment attempt on draft post \d+ from <HOST>`
EventData	`$ref_id`	Comment ID

New in version 4.0.0.

14.3.8 WPF2B_EVENT_COMMENT_PASSWORD

Attempted comment on password-protected Post.

syslog	Facility	*WP_FAIL2BAN_COMMENT_EXTRA_LOG*
	Level	NOTICE
fail2ban	Filter	*wordpress-extra.conf*
	Rule	`Comment attempt on password-protected post \d+ from <HOST>`
EventData	`$ref_id`	Comment ID

New in version 4.0.0.

14.4 Other Events

Whatever doesn't fit better into another Class.

14.4.1 WPF2B_EVENT_OTHER_UNKNOWN_PROXY

Attempted access via an untrusted proxy.

syslog	Facility	LOG_AUTH or LOG_AUTHPRIV
	Level	NOTICE
fail2ban	Filter	*wordpress-hard.conf*
	Rule	`Untrusted X-Forwarded-For header from <HOST>`

New in version 5.0.0.

See also:

WP_FAIL2BAN_USE_AUTHPRIV

14.5 Password Events

Password-related events.

14.5.1 WPF2B_EVENT_PASSWORD_REQUEST

Password reset request.

syslog	Facility	*WP_FAIL2BAN_PASSWORD_REQUEST_LOG*
	Level	NOTICE
fail2ban	Filter	*wordpress-extra.conf*
	Rule	`Password reset requested for .* from <HOST>`
EventData	`$username`	Username

New in version 4.0.0.

14.6 REST Events

REST API events.

14.7 Spam Events

Anything that can be classified as spam, not just comments.

14.8 XML-RPC Events

XML-RPC events, including Pingbacks.

14.8.1 WPF2B_EVENT_XMLRPC_PINGBACK

Pingback.

syslog	Facility	*WP_FAIL2BAN_PINGBACK_LOG*
	Level	INFO
fail2ban	Filter	*wordpress-soft.conf*
	Rule	`Pingback requested from <HOST>`

New in version 4.0.0.

14.8.2 WPF2B_EVENT_XMLRPC_PINGBACK_BOGUS

Bogus Pingback.

Premium only

syslog	Facility	*WP_FAIL2BAN_PINGBACK_LOG*
	Level	NOTICE
fail2ban	Filter	*wordpress-hard.conf*
	Rule	`.*; Bogus Pingback from <HOST>`

New in version 4.0.0.

14.8.3 WPF2B_EVENT_XMLRPC_PINGBACK_ERROR

Pingback error.

syslog	Facility	*WP_FAIL2BAN_PINGBACK_LOG*
	Level	NOTICE
fail2ban	Filter	*wordpress-hard.conf*
	Rule	`Pingback error .* generated from <HOST>`

New in version 4.0.0.

14.9 WAF Events

Web Application Firewall (WAF) events.

14.9.1 WPF2B_EVENT_WAF_SQLI

SQLi detected.

Premium only

syslog	Facility	*WP_FAIL2BAN_EX_WAF_LOG*
	Level	WARNING if enabled, NOTICE if logging only
fail2ban	Filter	*wordpress-hard.conf*
	Rule	`SQLi blocked from <HOST>`

New in version 5.1.0.

14.9.2 WPF2B_EVENT_WAF_UPDATE_OPTION

Unauthorised call to `update_option()` detected.

Premium only

syslog	Facility	*WP_FAIL2BAN_EX_WAF_LOG*
	Level	WARNING if enabled, NOTICE if logging only
fail2ban	Filter	*wordpress-hard.conf*
	Rule	`WAF: update_option(<option_name>)="<option_` `from <HOST>` **<option_name>** Name of the core WordPress option being updated. **<option_value>** The JSON-encoded value being set. The following options are used for encoding: • JSON_NUMERIC_CHECK • JSON_UNESCAPED_SLASHES • JSON_PRESERVE_ZERO_FRACTION • JSON_INVALID_UTF8_SUBSTITUTE

New in version 5.1.0.

CHAPTER 15

Filter Files

15.1 wordpress-hard.conf

```
# Fail2Ban filter for WordPress hard failures
# Auto-generated: 2019-04-18T14:45:30+00:00
#

[INCLUDES]

before = common.conf

[Definition]

_daemon = (?:wordpress|wp)

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

ignoreregex =

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

15.2 wordpress-soft.conf

```
# Fail2Ban filter for WordPress soft failures
# Auto-generated: 2019-04-18T14:45:30+00:00
#

[INCLUDES]

before = common.conf

[Definition]

_daemon = (?:wordpress|wp)

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

ignoreregex =

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

15.3 wordpress-extra.conf

```
# Fail2Ban filter for WordPress extra failures
# Auto-generated: 2019-04-18T14:45:30+00:00
#

[INCLUDES]

before = common.conf

[Definition]

_daemon = (?:wordpress|wp)

failregex = ^%(__prefix_line)sComment \d+ from <HOST>$
            ^%(__prefix_line)sComment post not found \d+ from <HOST>$
            ^%(__prefix_line)sComments closed on post \d+ from <HOST>$
            ^%(__prefix_line)sComment attempt on trash post \d+ from <HOST>$
            ^%(__prefix_line)sComment attempt on draft post \d+ from <HOST>$
            ^%(__prefix_line)sComment attempt on password-protected post \d+ from
↪<HOST>$
            ^%(__prefix_line)sPassword reset requested for .* from <HOST>$

ignoreregex =

# DEV Notes:
# Requires the 'WP fail2ban' plugin:
# https://wp-fail2ban.com/
```

(continues on next page)

(continued from previous page)

```
#
# Author: Charles Lecklider
```

PHP Namespace Index

O

Index

U