INTRODUCTION TO WORDPRESS SECURITY

Joe McGill

STL WordPress Meetup • #STLWP

Human Made

HOW TO KEEP A WORDPRESS SITE SECURE

Common reasons WordPress sites get hacked

- Bad/Weak Passwords
- Not updating WordPress Core
- Not updating Plugins
- Using a cheap/free theme without long-term security updates

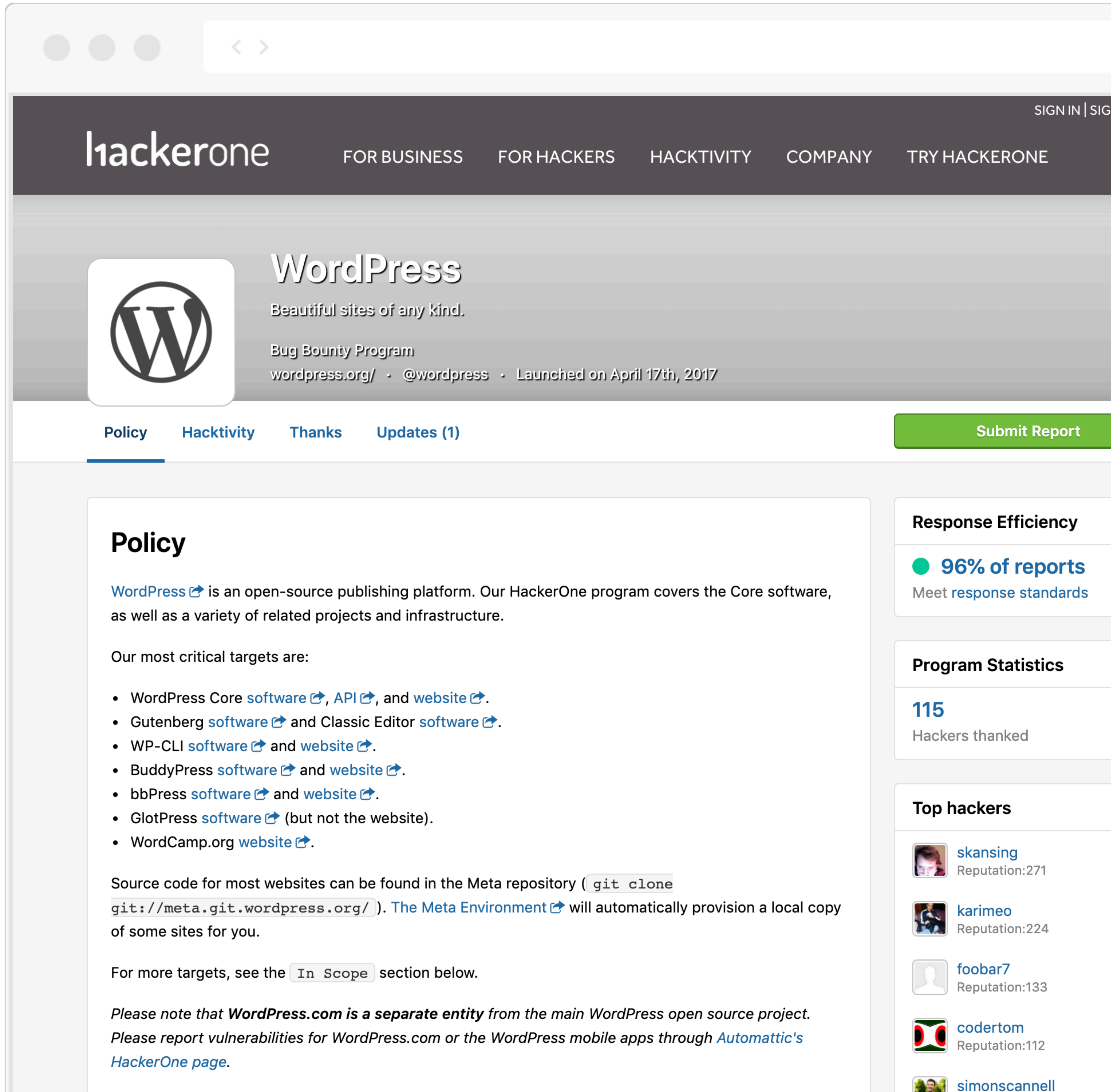
> *“Nearly all software has security flaws. The easiest ones to exploit are the ones you don’t fix once they’re publicly known.”*

#STLWP

Responsible disclosure: HackerOne

Vulnerabilities are communicated privately so the flaw can be fixed before the issue is publicly disclosed.

Bounty programs let people get paid to do the right thing.



WordPress

Beautiful sites of any kind.

Bug Bounty Program
wordpress.org/ · @wordpress · Launched on April 17th, 2017

Policy

WordPress is an open-source publishing platform. Our HackerOne program covers the Core software, as well as a variety of related projects and infrastructure.

Our most critical targets are:

- WordPress Core software, API, and website.
- Gutenberg software and Classic Editor software.
- WP-CLI software and website.
- BuddyPress software and website.
- bbPress software and website.
- GlotPress software (but not the website).
- WordCamp.org website.

Source code for most websites can be found in the Meta repository (`git clone git://meta.git.wordpress.org/`). The Meta Environment will automatically provision a local copy of some sites for you.

For more targets, see the `In Scope` section below.

*Please note that **WordPress.com is a separate entity** from the main WordPress open source project. Please report vulnerabilities for WordPress.com or the WordPress mobile apps through Automattic's HackerOne page.*

Response Efficiency

96% of reports
Meet response standards

Program Statistics

115
Hackers thanked

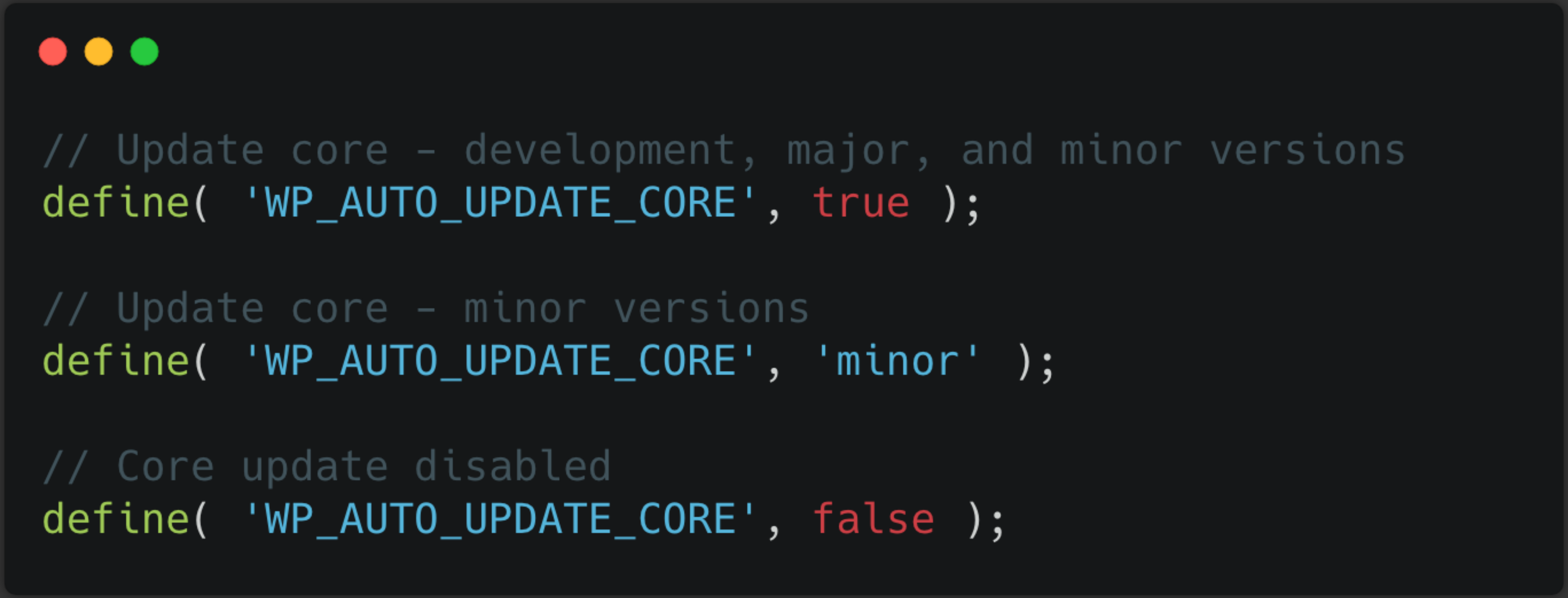
Top hackers

skansing	Reputation:271
karimeo	Reputation:224
foobar7	Reputation:133
codertom	Reputation:112
simonscannell	

Keeping your software up to date

- WordPress automatically updates minor versions.
- You can turn on auto-updates for major versions.
- You can turn on auto-updates for plugins and themes.
- *Auto updates can break things, so use judgement!*

Auto update WordPress

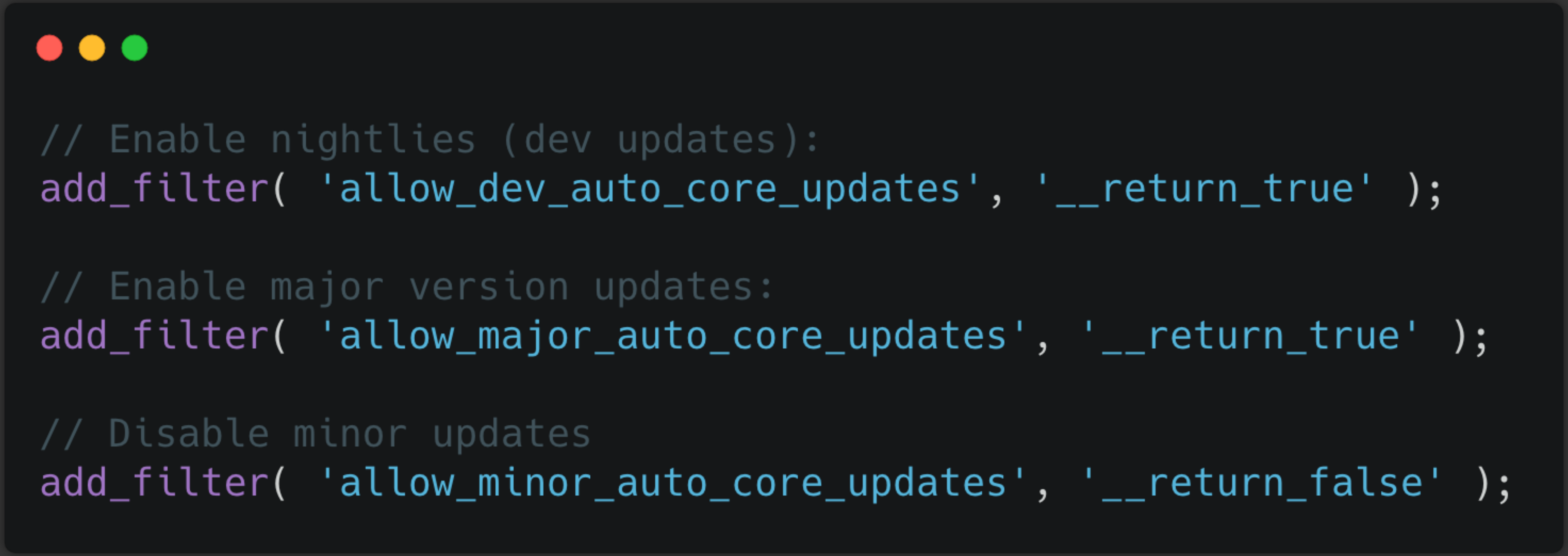


```
// Update core - development, major, and minor versions
define( 'WP_AUTO_UPDATE_CORE', true );

// Update core - minor versions
define( 'WP_AUTO_UPDATE_CORE', 'minor' );

// Core update disabled
define( 'WP_AUTO_UPDATE_CORE', false );
```

Auto update WordPress

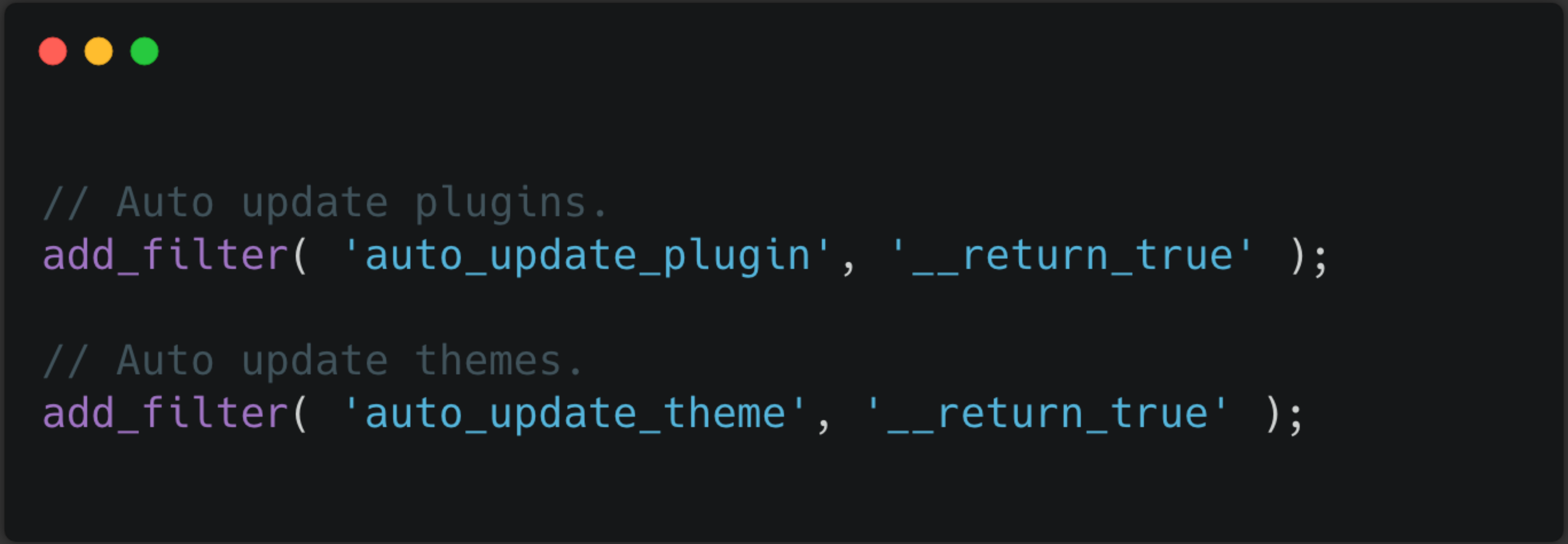


```
// Enable nightlies (dev updates):
add_filter( 'allow_dev_auto_core_updates', '__return_true' );

// Enable major version updates:
add_filter( 'allow_major_auto_core_updates', '__return_true' );

// Disable minor updates
add_filter( 'allow_minor_auto_core_updates', '__return_false' );
```

Auto update plugins and themes

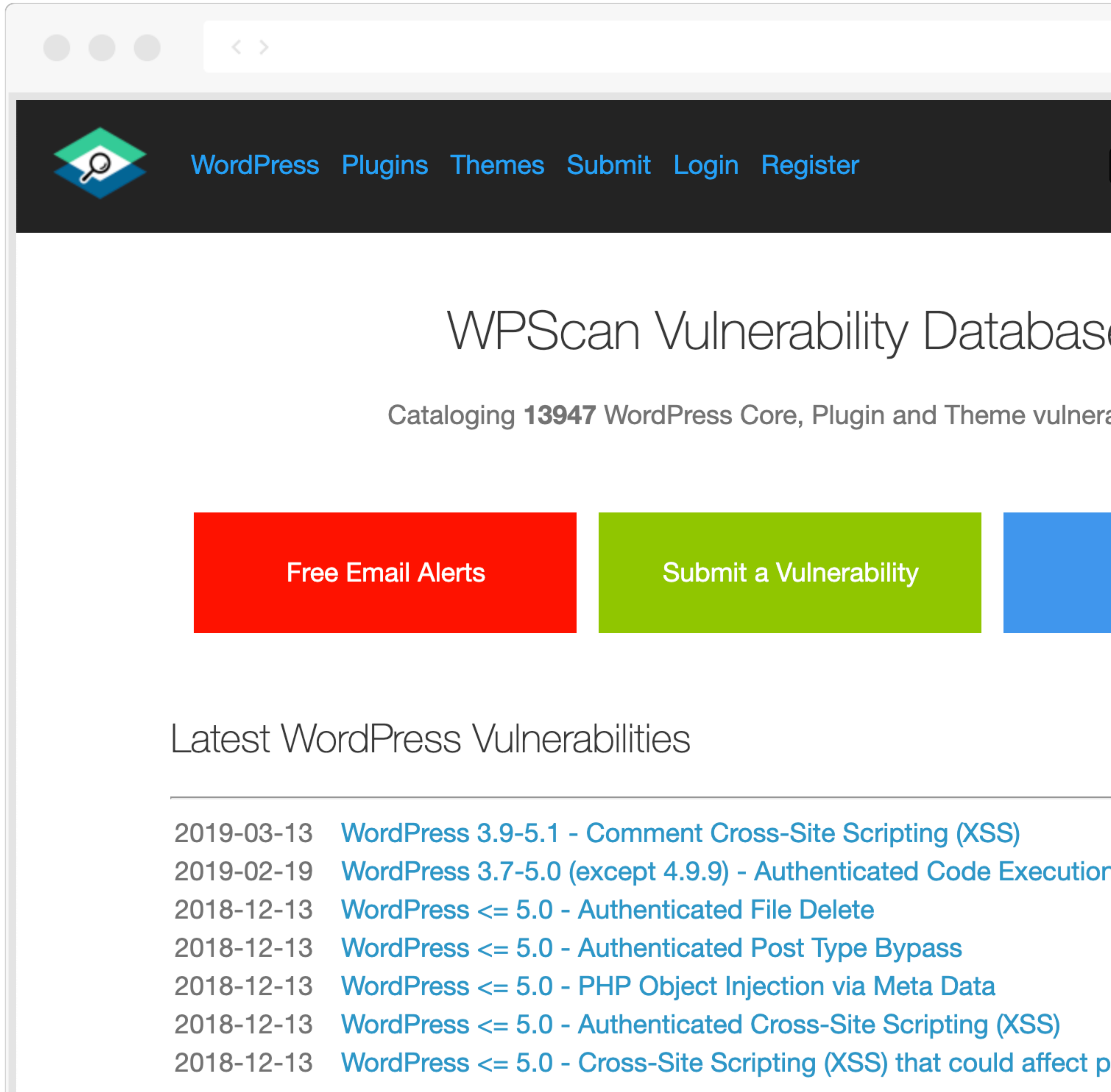


```
// Auto update plugins.
add_filter( 'auto_update_plugin', '__return_true' );

// Auto update themes.
add_filter( 'auto_update_theme', '__return_true' );
```

Track WordPress vulnerabilities

The WPScan Vulnerability Database (wpvulndb.com) posts vulnerabilities as they are available.



WPScan Vulnerability Databas

Cataloging **13947** WordPress Core, Plugin and Theme vulnera

Latest WordPress Vulnerabilities

2019-03-13 WordPress 3.9-5.1 - Comment Cross-Site Scripting (XSS)
2019-02-19 WordPress 3.7-5.0 (except 4.9.9) - Authenticated Code Executior
2018-12-13 WordPress <= 5.0 - Authenticated File Delete
2018-12-13 WordPress <= 5.0 - Authenticated Post Type Bypass
2018-12-13 WordPress <= 5.0 - PHP Object Injection via Meta Data
2018-12-13 WordPress <= 5.0 - Authenticated Cross-Site Scripting (XSS)
2018-12-13 WordPress <= 5.0 - Cross-Site Scripting (XSS) that could affect p

What to do if you get hacked

1. Restore a backup of your database
2. Change all your passwords
3. Download and install a fresh copy of WordPress
4. Download and install fresh copies of your plugins
5. Install a fresh copy of your theme

HOW TO WRITE MORE SECURE CODE

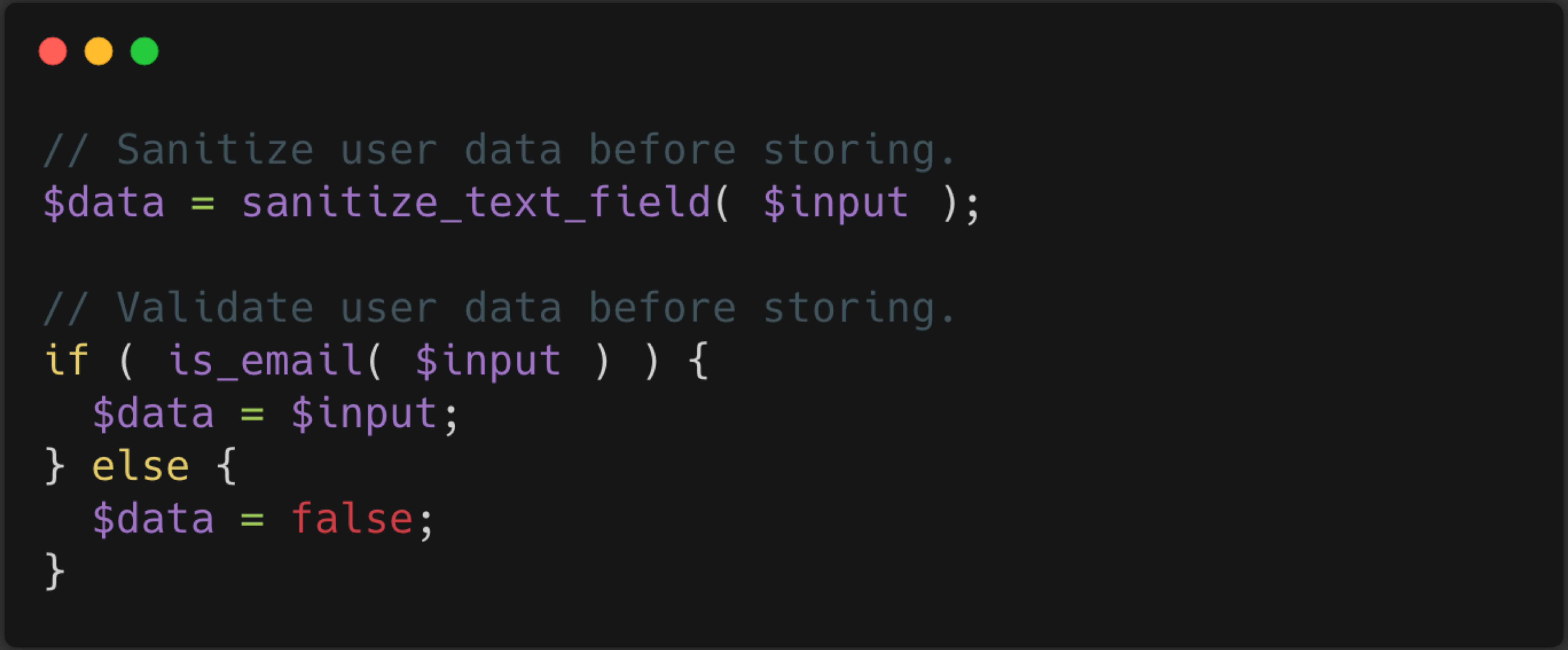
Security best practices

- Never trust external data (even from yourself)
- Sanitize/validate on input, escape on output
- Check user authorization and intent

Sanitization and validation

- **Sanitization** removes all undesirable data from input before saving it to the database.
- **Validation** checks if the data is what is expected and discards it if the data is not valid.

Sanitization/Validation examples



```php
// Sanitize user data before storing.
$data = sanitize_text_field( $input );

// Validate user data before storing.
if ( is_email( $input ) ) {
  $data = $input;
} else {
  $data = false;
}
```

sanitize_text_field()

Sanitize the data provided by text input fields in forms

- Removes all HTML tags.
- Removes whitespace from the start and end of the string.
- Removes extra whitespace between words.
- Removes tabs and line breaks.
- Converts stand-alone < characters into an HTML entity.
- Removes any invalid UTF-8 characters.
- Removes % encoded octets.

absint()

Converts any value to non-negative integer.

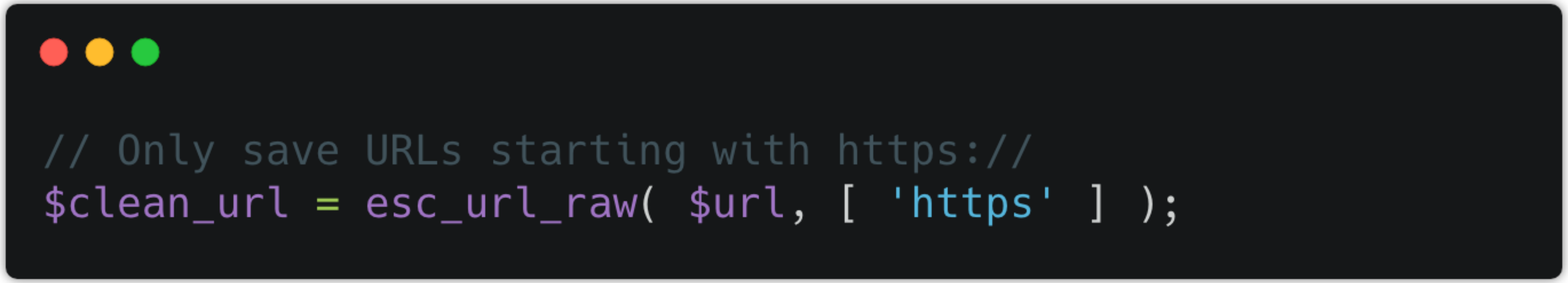
Useful for sanitizing IDs.

Integers are safe to use in any context. When you pass invalid data (like a text string) to `absint()`, the return is most likely a 0.

esc_url_raw()

Sanitizes URLs for safe storage in a database by stripping undesired characters and verifying the URL protocol.

The function accepts two arguments: the URL to clean, as well as an optional array of allowed protocols.



```
// Only save URLs starting with https://
$clean_url = esc_url_raw( $url, [ 'https' ] );
```

PHP validation functions

`is_bool()`: Returns true if the passed variable is of the type boolean.

`is_float()`: Returns true if the passed variable is of the type float.

`is_int()`: Returns true if the passed variable is of the type integer.

`is_numeric()`: Returns true if the passed variable contains a numeric value.

Escaping data before outputting

Escaping is used to ensure that data is safe to be output to the browser. WordPress offers a number of escaping functions. The type of escaping function to use depends on the context in which the data is output, e.g. HTML vs JS.

When writing code, **always escape immediately before output**. This makes it clear when and how data is escaped, making the code easy to review and to understand.

WordPress escaping functions

`esc_html()`: ensures text is safe to output in HTML.

`esc_attr()`: ensures that data is safe to be output inside of HTML attributes.

`esc_js()`: ensures data is safe to be output inside a JavaScript string. *Is usually not the right function.*

`wp_json_encode()`: ensures that data is safe to be printed inside of JavaScript code.

Human Made

Engineering Handbook

https://engineering.hmn.md

Additional information about security best practices, coding standards, and how we engineer enterprise level projects.

THANK YOU.

Joe McGill

joemcgill • joemcgill@humanmade.com

Human Made