

- Secrets
- ABAP
- Apex
- C
- C++
- CloudFormation
- COBOL
- C#
- CSS
- Flex
- Go
- HTML
- Java
- JavaScript
- Kotlin
- Objective C
- PHP**
- PL/I
- PL/SQL
- Python
- RPG
- Ruby
- Scala
- Swift
- Terraform
- Text
- TypeScript
- T-SQL
- VB.NET
- VB6
- XML



PHP static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your PHP code

All rules 268

Vulnerability 40

Bug 51

Security Hotspot 33

Code Smell 144

Tags ▾

Search by name... 🔍

Bug
Regular expressions should be syntactically valid
Bug
Only one method invocation is expected when testing exceptions
Bug
Reading the Standard Input is security-sensitive
Security Hotspot
Using command line arguments is security-sensitive
Security Hotspot
Using Sockets is security-sensitive
Security Hotspot
Encrypting data is security-sensitive
Security Hotspot
Using regular expressions is security-sensitive
Security Hotspot
Deserializing objects from an untrusted source is security-sensitive
Security Hotspot
Literal boolean values and nulls should not be used in equality assertions
Code Smell
"global" should not be used
Code Smell
"switch" statements should not be nested
Code Smell

WordPress option names should not be misspelled

Analyze your code

Code Smell

Major

?

WordPress relies a lot on the configuration located in a file named `wp-config.php`. This file contains mostly `define` statements and each of them creates a constant for a given WordPress option. However, no warning appears if an option is misspelled: the statement simply defines a constant which is never used.

This rule raises an issue when a file named `wp-config.php` defines a constant whose name is slightly different from a known WordPress option.

Noncompliant Code Example

```
define( 'DISALLOW_FILE_MOD', true ); // Noncompliant
define( 'Disallow_File_Mods', true ); // Noncompliant
```

Compliant Solution

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

Available In:

sonarlint

sonarcloud

sonarqube

Cyclomatic Complexity of functions should not be too high

 Code Smell


Control flow statements "if", "for", "while", "switch" and "try" should not be nested too deeply

 Code Smell

Cyclomatic Complexity of classes should not be too high

 Code Smell

"if ... else if" constructs should end with "else" clauses

 Code Smell

Expressions should not be too