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...

Cryptographic keys should be robust
Vulnerability

Weak SSL/TLS protocols should not be used
Vulnerability

Regular expressions should not be vulnerable to Denial of Service attacks
Vulnerability

Hashes should include an unpredictable salt
Vulnerability

Regular expressions should have valid delimiters
Bug

Regex lookahead assertions should not be contradictory
Bug

Back references in regular expressions should only refer to capturing groups that are matched before the reference
Bug

Regex boundaries should not be used in a way that can never be matched
Bug

Regex patterns following a possessive quantifier should not always fail
Bug

Assertion failure exceptions should not be ignored
Bug

References used in "foreach" loops should be "unset"
Bug

Raised Exceptions must derive from Throwable

Analyze your code

Bug Blocker

Instances of classes that do not derive from the "Throwable" interface cannot be used in a PHP "throw" statement.

Many built-in exceptions such as "Exception" and the SPL exception classes do implement the "Throwable" interface and can be extended when creating custom exceptions.

This rule raises an issue when an instance of a class that does not implement the "Throwable" interface is used in a "throw" statement .

Noncompliant Code Example

```
class NoThrowable {}

throw new NoThrowable(); // Noncompliant
```

Compliant Solution

```
<?php

class SomeThrowable implements Throwable {
    // Implementation of the Throwable methods
}

throw new SomeThrowable(); // Compliant

class SomeCustomException extends Exception {}

throw new SomeCustomException(); // Compliant{code}
```

Available In:
sonarlint | sonarcloud | sonarqube

Bug

Using clear-text protocols is security-sensitive

Security Hotspot

Expanding archive files without controlling resource consumption is security-sensitive

Security Hotspot

Signalling processes is security-sensitive

Security Hotspot

Configuring loggers is security-sensitive

Security Hotspot