sonar RULES

Products

- 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...*

Cognitive Complexity of functions should not be too high
Code Smell

Parentheses should not be used for calls to "echo"
Code Smell

Functions should not be nested too deeply
Code Smell

References should not be passed to function calls
Code Smell

"switch" statements should have "default" clauses
Code Smell

Control structures should use curly braces
Code Smell

String literals should not be duplicated
Code Smell

Methods should not be empty
Code Smell

Constant names should comply with a naming convention
Code Smell

Secret keys and salt values should be robust
Vulnerability

Authorizations should be based on strong decisions
Vulnerability

Server-side requests should not be vulnerable to forging attacks

## Weak SSL/TLS protocols should not be used

Analyze your code

Vulnerability | Critical | cwe privacy owasp sans-top25

This rule raises an issue when an insecure TLS protocol version (i.e. a protocol different from "TLSv1.2", "TLSv1.3", "DTLSv1.2", or "DTLSv1.3") is used or allowed.

It is recommended to enforce TLS 1.2 as the minimum protocol version and to disallow older versions like TLS 1.0. Failure to do so could open the door to downgrade attacks: a malicious actor who is able to intercept the connection could modify the requested protocol version and downgrade it to a less secure version.

### Noncompliant Code Example

```
$ctx = stream_context_create([
  'ssl' => [
    'crypto_method' =>
      STREAM_CRYPTO_METHOD_TLSv1_1_CLIENT // Noncomplia
  ],
]);
```

### Compliant Solution

```
$ctx = stream_context_create([
    'ssl' => [
        'crypto_method' => STREAM_CRYPTO_METHOD_TLSv1_2
    ],
]);
```

### See

- OWASP Top 10 2021 Category A2 - Cryptographic Failures
- OWASP Top 10 2021 Category A7 - Identification and Authentication Failures
- OWASP Top 10 2017 Category A3 - Sensitive Data Exposure
- OWASP Top 10 2017 Category A6 - Security Misconfiguration
- Mobile AppSec Verification Standard - Network Communication Requirements
- OWASP Mobile Top 10 2016 Category M3 - Insecure Communication
- MITRE, CWE-327 - Inadequate Encryption Strength
- MITRE, CWE-326 - Use of a Broken or Risky Cryptographic Algorithm
- SANS Top 25 - Porous Defenses
- SSL and TLS Deployment Best Practices - Use secure protocols

Available In:

sonarlint sonarcloud sonarqube

Vulnerability

The number of arguments passed to a function should match the number of parameters

Bug

Non-empty statements should change control flow or have at least one side-effect

Bug

Variables should be initialized before use

Bug

Replacement strings should reference existing regular expression groups