

- 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...



"default" clauses should be first or last

Code Smell

A conditionally executed single line should be denoted by indentation

Code Smell

Conditionals should start on new lines

Code Smell

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

### LDAP connections should be authenticated

Analyze your code

Vulnerability Critical cwe owasp

An LDAP client authenticates to an LDAP server with a "bind request" which provides, among other, a [simple authentication method](#).

Simple authentication in LDAP can be used with three different mechanisms:

- Anonymous Authentication Mechanism* by performing a bind request with a username and password value of zero length.
- Unauthenticated Authentication Mechanism* by performing a bind request with a password value of zero length.
- Name/Password Authentication Mechanism* by performing a bind request with a password value of non-zero length.

Anonymous binds and unauthenticated binds allow access to information in the LDAP directory without providing a password, their use is therefore strongly discouraged.

#### Noncompliant Code Example

```
$ldapconn = ldap_connect("ldap.example.com");

if ($ldapconn) {
    $ldapbind = ldap_bind($ldapconn); // Noncompliant; anonymous
}
```

#### Compliant Solution

```
$ldapprdn = 'uname';
$dappass = 'password';

$ldapconn = ldap_connect("ldap.example.com");

if ($ldapconn) {
    $ldapbind = ldap_bind($ldapconn, $ldapprdn, $dappass);
}
```


#### See

- [OWASP Top 10 2021 Category A7](#) - Identification and Authentication Failures
- [OWASP Top 10 2017 Category A2](#) - Broken Authentication
- [MITRE, CWE-521](#) - Weak Password Requirements
- [ldapwiki.com](#) - Simple Authentication

Available In:

sonarlint | sonarcloud | sonarqube


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

 Vulnerability

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

 Bug

SONARSOURCE, SONARLINT, SONARQUBE and SONARCLOUD are trademarks of SonarSource S.A. All other trademarks and copyrights are the property of their respective owners. All rights are expressly reserved.  
[Privacy Policy](#)