

JavaScript

Objective C

Kotlin

PHP

PL/I

PL/SQL

Python

RPG

Ruby

Scala

Swift

Text

T-SQL

VB.NET

VB6

XML

Terraform

TypeScript

JS

Php

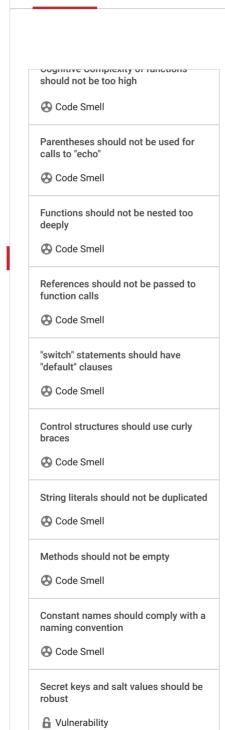
N.

■



PHP static code analysis

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

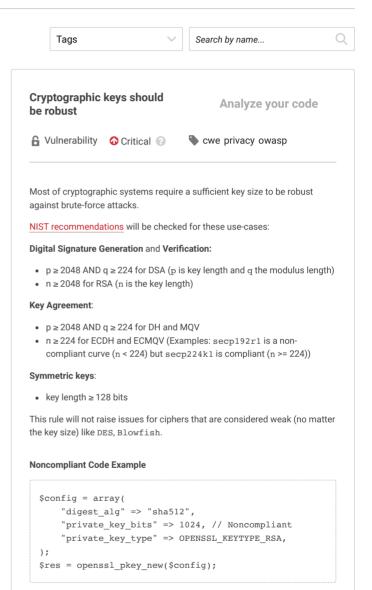


Authorizations should be based on

Server-side requests should not be vulnerable to forging attacks

strong decisions

Vulnerability



Compliant Solution

\$config = array(

"digest_alg" => "sha512",

\$res = openssl pkey new(\$config);

- OWASP Top 10 2021 Category A2 Cryptographic Failures
- OWASP Top 10 2017 Category A3 Sensitive Data Exposure
- OWASP Top 10 2017 Category A6 Security Misconfiguration

"private_key_bits" => 2048 // Compliant
"private_key_type" => OPENSSL_KEYTYPE_RSA,

- Mobile AppSec Verification Standard Cryptography Requirements
- OWASP Mobile Top 10 2016 Category M5 Insufficient Cryptography

6 Vulnerability

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



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



Variables should be initialized before

🖟 Bug

Replacement strings should reference existing regular expression groups

- NIST 800-131A Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths
- MITRE, CWE-326 Inadequate Encryption Strength

Available In:

sonarlint ⊕ | sonarcloud 👌 | sonarqube

© 2008-2022 SonarSource S.A., Switzerland. All content is copyright protected. SONAR, 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