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

TypeScript static code analysis

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

All rules 279 | Vulnerability 27 | Bug 51 | Security Hotspot 43 | Code Smell 158 | Quick Fix 50

Tags | Search by name...

security-sensitive
Security Hotspot

Using weak hashing algorithms is security-sensitive
Security Hotspot

Disabling CSRF protections is security-sensitive
Security Hotspot

Using pseudorandom number generators (PRNGs) is security-sensitive
Security Hotspot

Dynamically executing code is security-sensitive
Security Hotspot

Equality operators should not be used in "for" loop termination conditions
Code Smell

Tests should not execute any code after "done()" is called
Code Smell

Union and intersection types should not be defined with duplicated elements
Code Smell

"default" clauses should be last
Code Smell

"await" should only be used with promises
Code Smell

A conditionally executed single line should be denoted by indentation
Code Smell

Conditionals should start on new lines

Cipher algorithms should be robust

Analyze your code

Vulnerability | Critical | cwe privacy owasp sans-top25

Strong cipher algorithms are cryptographic systems resistant to cryptanalysis, they are not vulnerable to well-known attacks like brute force attacks for example.

A general recommendation is to only use cipher algorithms intensively tested and promoted by the cryptographic community.

More specifically for block cipher, it's not recommended to use algorithm with a block size inferior than 128 bits.

Noncompliant Code Example

crypto built-in module:

```
crypto.createCipheriv("DES", key, iv); // Noncompliant: DES
crypto.createCipheriv("DES-EDE", key, ""); // Noncompliant:
crypto.createCipheriv("DES-EDE3", key, ""); // Noncompliant:
crypto.createCipheriv("RC2", key, iv); // Noncompliant: RC2
crypto.createCipheriv("RC4", key, "");// Noncompliant: RC4 i
crypto.createCipheriv("BF", key, iv);// Noncompliant: Blowfi
```

Compliant Solution

crypto built-in module:

```
crypto.createCipheriv("AES-256-GCM", key, iv);
```

See

- OWASP Top 10 2021 Category A2 - Cryptographic Failures
- OWASP Top 10 2017 Category A3 - Sensitive Data Exposure
- MITRE, CWE-327 - Use of a Broken or Risky Cryptographic Algorithm
- SANS Top 25 - Porous Defenses

Available In:
sonarlint | sonarcloud | sonarqube

Code Smell

Cognitive Complexity of functions should not be too high

Code Smell

"void" should not be used

Code Smell

Loop counters should not be assigned to from within the loop body

Code Smell

"for" loop increment clauses should modify the loops' counters

Code Smell