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

# JavaScript static code analysis

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

All rules 285 | Vulnerability 29 | Bug 62 | Security Hotspot 43 | Code Smell 151 | Quick Fix 41

Tags | Search by name...

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

"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
Code Smell

Cognitive Complexity of functions should not be too high
Code Smell

"void" should not be used

## Encryption algorithms should be used with secure mode and padding scheme

Analyze your code

Vulnerability Critical cwe privacy owasp sans-top25

Encryption operation mode and the padding scheme should be chosen appropriately to guarantee data confidentiality, integrity and authenticity:

- For block cipher encryption algorithms (like AES):
  - The GCM (Galois Counter Mode) mode which works internally with zero/no padding scheme, is recommended, as it is designed to provide both data authenticity (integrity) and confidentiality. Other similar modes are CCM, CWC, EAX, IAPM and OCB.
  - The CBC (Cipher Block Chaining) mode by itself provides only data confidentiality, it's recommended to use it along with Message Authentication Code or similar to achieve data authenticity (integrity) too and thus to prevent padding oracle attacks.
  - The ECB (Electronic Codebook) mode doesn't provide serious message confidentiality: under a given key any given plaintext block always gets encrypted to the same ciphertext block. This mode should not be used.
- For RSA encryption algorithm, the recommended padding scheme is OAEP.

**Noncompliant Code Example**

crypto built-in module:

```
crypto.createCipheriv("AES-128-CBC", key, iv); // Noncomplia
crypto.createCipheriv("AES-128-ECB", key, ""); // Noncomplia
```

**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 A6 - Security Misconfiguration
- MITRE, CWE-327 - Use of a Broken or Risky Cryptographic Algorithm
- SANS Top 25 - Porous Defenses

Available In:
sonarlint | sonarcloud | sonarqube

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

Functions should not be empty

Code Smell

Server-side requests should not be vulnerable to forging attacks