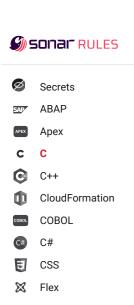
Search by name.



© Go

Js JavaScript

Kotlin

Kubernetes

ó Objective C

PHP

PL/I

PL/SQL PL/SQL

🦆 Python

RPG RPG

Ruby

Scala

Swift

Terraform

■ Text

Ts TypeScript

T-SQL

VB VB.NET

VB6 VB6

XML XML



C static code analysis

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

Tags

"memset" should not be used to delete sensitive data 6 Vulnerability POSIX functions should not be called with arguments that trigger buffer overflows ♠ Vulnerability XML parsers should not be vulnerable to XXE attacks Vulnerability Function-like macros should not be invoked without all of their arguments 👬 Bug The address of an automatic object should not be assigned to another object that may persist after the first object has ceased to exist 👬 Bug "pthread_mutex_t" should be unlocked in the reverse order they were locked "pthread_mutex_t" should be properly initialized and destroyed # Bua "pthread_mutex_t" should not be consecutively locked or unlocked # Bug Functions with "noreturn" attribute should not return # Bug

"memcmp" should only be called with pointers to trivially copyable types

with no padding

🖷 Bug

```
Jump statements should not be
                                             Analyze your code
redundant
redundant clumsv
Jump statements, such as return, break, goto, and continue let you change the
default flow of program execution, but jump statements that direct the control flow to
the original direction are just a waste of keystrokes.
Noncompliant Code Example
 void Foo()
    goto A; // Noncompliant
    while (condition1)
      if (condition2)
        continue; // Noncompliant
      else
        DoTheThing();
    return; // Noncompliant; this is a void method
Compliant Solution
 void Foo()
    while (condition1)
      if (!condition2)
        DoTheThing();
 Available In:
 sonarlint ⊖ | sonarcloud 🟡 | sonarqube ) Develop
```

© 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

Stack allocated memory and nonowned memory should not be freed

R
Bug

Closed resources should not be
accessed
Bug

Dynamically allocated memory should
be released
Bug

Freed memory should not be used