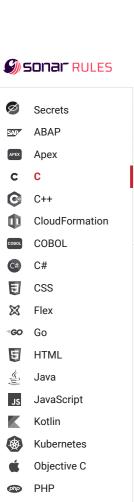
O Quick 14



PL/I

PL/SQL Python

RPG

Ruby Scala

Swift

Terraform

Text

**TypeScript** 

T-SQL

**VB.NET** 

VB6

XML



ΑII 311

rules

## C static code analysis

6 Vulnerability (13)

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

**₩** Bug (74)

Security

Hotspot

Tags

18

"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

<table-of-contents> Bug

🖷 Bug

with no padding

"memcmp" should only be called with

pointers to trivially copyable types

```
Loop variables should be declared
                                                Analyze your code
in the minimal possible scope
Code
                                              cppcoreguidelines
                              Quick
                O Minor
                                       0
   Smell
                             Fix
                                                 clumsy
When a loop variable is not used outside of a loop, it should be declared inside the
loop declaration:
  • It improves readability, the scope of the variable is clearly defined

    It reduces the number of mistakes, the variable can't be accidentally misused.

   outside of the loop
 · Resources are not retained longer than necessary
Noncompliant Code Example
  using namespace std;
  void f() {
    int i = 0; // Noncompliant: i is not used outside of the lo
    for (i = 0; i < 10; ++i) {
      cout << i << endl;
    }
  }
Compliant Solution
  using namespace std;
  void f() {
    for (int i = 0; i < 10; ++i) {
      cout << i << endl;
  }
 • C++ Core Guidelines ES.5 - Keep scopes small
 Available In:
 sonarlint ⊖ | sonarcloud ☆ | sonarqube |
```

Search by name.

Smell

© 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