



△BAP

Apex Apex

C C

© C++

CloudFormation

COBOL COBOL

C# C#

CSS

X Flex

GO Go

5 HTML

🐇 Java

Js JavaScript

Kotlin

Kubernetes

Objective C

PHP

PL/I

PL/SQL

Python

RPG RPG

Ruby

Scala

Swift

Terraform

■ Text

Ts TypeScript

T-SQL

VB VB.NET

VB6 VB6

xmL XML

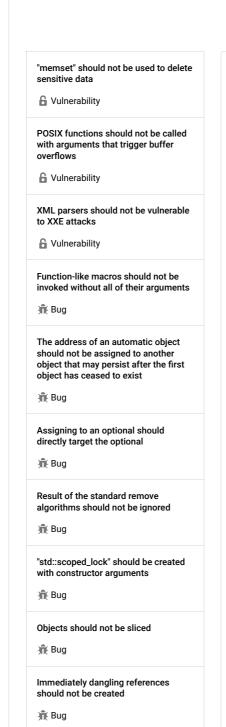


C++ static code analysis

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

All 578 rules Support Security 18 Security

Tags



"pthread_mutex_t" should be unlocked in the reverse order they were locked

"pthread_mutex_t" should be properly

Bug

```
Memory locations should not be
                                              Analyze your code
released more than once
# Bug Blocker
                          cwe symbolic-execution
Using free(...) or delete releases the reservation on a memory location, making
it immediately available for another purpose. So releasing the same memory location
twice can lead to corrupting the program's memory.
A best practice to avoid this bug calls for setting just-freed pointers to NULL, and
always null-testing before a free or delete.
Noncompliant Code Example
  void doSomething(int size) {
   char *cp = (char *) malloc(sizeof(char) * size);
    // ...
    if (condition) {
      free(cp);
    free(cp); // Noncompliant
Compliant Solution
  void doSomething(int size) {
    char *cp = (char *) malloc(sizeof(char) * size);
    // ...
    if (condition) {
      free(cp);
      cp = NULL; // This will prewent freeing the same memory a
    free(cp); // This is OK: if the memory was freed in the if-
    cp = NULL; // This will prevent freeing the same memory aga
```

Search by name.

© 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.

 $\textbf{sonarlint} \begin{tabular}{ll} \hline \textbf{sonarcloud} \begin{tabular}{ll} \hline \& \textbf{sonarqube} \\ \hline \end{tabular} \begin{tabular}{ll} \hline \textbf{Developer} \\ \hline \textbf{Edition} \\ \hline \end{tabular}$

MITRE, CWE-415 - Double Free
 OWASP, Doubly freeing memory

Available In:

Privacy Policy

initialized and destroyed	
"pthread_mutex_t" should not be consecutively locked or unlocked twice	
"std::move" and "std::forward" should not be confused	
A call to "wait()" on a "std::condition_variable" should have a	•