Secrets
ABAP
Apex
C
C++
CloudFormation
COBOL
C#
CSS
Flex
Go
HTML
Java
JavaScript
Kotlin
Kubernetes
Objective C
PHP
PL/I
PL/SQL
Python
RPG
Ruby
Scala
Swift
Terraform
Text
TypeScript
T-SQL
VB.NET
VB6
XML

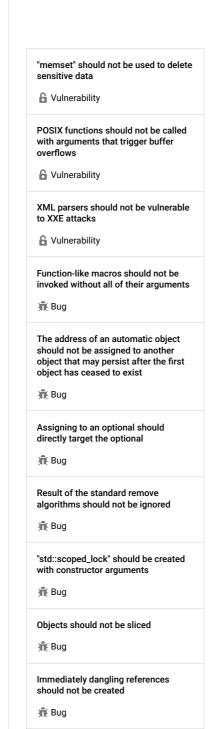# C++ static code analysis

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

All rules 578 | 🔒 Vulnerability 13 | 🐛 Bug 111 | 🛡 Security Hotspot 18 | ⊙ Code Smell 436 | ⚡ Quick Fix 68

Tags ⌄ | Search by name...

"memset" should not be used to delete sensitive data

🔒 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

Assigning to an optional should directly target the optional

🐛 Bug

Result of the standard remove algorithms should not be ignored

🐛 Bug

"std::scoped_lock" should be created with constructor arguments

🐛 Bug

Objects should not be sliced

🐛 Bug

Immediately dangling references should not be created

🐛 Bug

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

🐛 Bug

"pthread_mutex_t" should be properly

Switch labels should not be nested inside non-switch blocks

Analyze your code

⊙ Code Smell   ❗ Blocker ⍰   🏷 misra-c++2008   misra-c2004   confusing   misra-c2012

A switch-label can be placed anywhere within the statements that form the body of a switch statement, potentially leading to unstructured code. To prevent this from happening, the scope of a case-label or default-label shall be the statement forming the body of a switch statement. All case-clauses and the default-clause shall be at the same scope.

Noncompliant Code Example

```
switch (x) {
  case 1: // Compliant
    if (foo) {
      case 2: // Noncompliant
        break;
      default: // Noncompliant
        break;
    }
    break;
  default: // Compliant
    break;
}
```

See

- MISRA C 2004, 15.1 - A switch label shall only be used when the most closely-enclosing compound statement is the body of a switch statement.
- MISRA C++ 2008, 6-4-4 - A switch-label shall only be used when the most closely-enclosing compound statement is the body of a switch statement.
- MISRA C 2012, 16.2 - A switch label shall only be used when the most closely-enclsoing compound statement is the body of a switch statement

Available In:

sonarlint 😊 | sonarcloud ⌘ | sonarqube ⌇ Developer Edition

---

initialized and destroyed

🐞 Bug

"pthread_mutex_t" should not be consecutively locked or unlocked twice

🐞 Bug

"std::move" and "std::forward" should not be confused

🐞 Bug

A call to "wait()" on a "std::condition_variable" should have a