



ABAP Apex

С

C++

CloudFormation

COBOL

C#

CSS

Flex

=GO

HTML 5

Go

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 578 6 Vulnerability (13) rules

R Bug (111)

o Security Hotspot

⊕ Code (436)

Quick 68 Fix

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 initialized and destroyed

📆 Bug

"pthread_mutex_t" should not be consecutively locked or unlocked twice

Lines starting with "#" should contain valid preprocessing directives

🛊 Bug 🔷 Major 🕝

Analyze your code

Preprocessing directives (lines that start with #) can be used to conditionally include or exclude code from compilation. Malformed preprocessing directives could lead to the exclusion or inclusion of more code than was intended. Therefore all

preprocessing directives should be syntactically meaningful.

🖣 based-on-misra preprocessor

Noncompliant Code Example

```
#define AAA 2
int foo(void)
  int x = 0;
#ifndef AAA
  x = 1;
#else1 /* Noncompliant */
  x = AAA;
#endif
 return x;
```

Compliant Solution

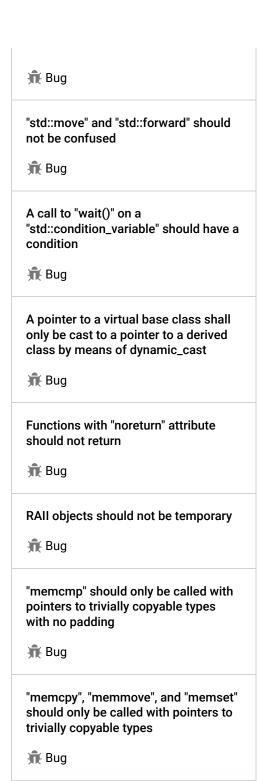
```
#define AAA 2
int foo(void)
  int x = 0;
#ifndef AAA
  x = 1;
#else
  x = AAA;
#endif
  return x;
```

See

- MISRA C:2004, 19.16 Preprocessing directives shall be syntactically meaningful even when excluded by preprocessor.
- MISRA C++:2008, 16-0-8 If the # token appears as the first token on a line, then it shall be immediately followed by a preprocessing token.
- MISRA C:2012, 20.13 A line whose first token is # shall be a valid preprocessing directive

Available In:

sonarlint in sonarcloud sonarqube Developer Edition



"std::auto_ptr" should not be used

Destructors should be "noexcept"

📆 Bug

📆 Bug

© 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