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 (311) | 🔒 Vulnerability (13) | 🐞 Bug (74) | 🛡 Security Hotspot (18) | ⊙ Code Smell (206) | ⚡ Quick Fix (14) |

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

"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

🐞 Bug

Functions with "noreturn" attribute should not return

🐞 Bug

"memcmp" should only be called with pointers to trivially copyable types with no padding

🐞 Bug

"enum" values should not be used as operands to built-in operators other than [ ], =, ==, !=, unary &, and the relational operators <, <=, >, >=

Analyze your code

⊙ Code Smell   ⚠ Major ⌕   🏷 based-on-misra suspicious

Enumerations are used to represent symbolic values, or sometimes bit fields. They are not supposed to be used in arithmetic contexts.

Additionally, even though comparing them with integer numbers can make sense (for instance, to test if an enum lies with a certain range), comparing them with floating point numbers does not (and is deprecated since C++20).

There are other restrictions related to the use of enums, see for instance {rule:cpp:S2753}.

Noncompliant Code Example

```
enum { COLOUR_0, COLOUR_1, COLOUR_2, COLOUR_COUNT } colour;
if ( COLOUR_0 == colour ) { ... }
if ( ( COLOUR_0 + COLOUR_1 ) == colour ) { ... } // Noncompli
if ( colour < COLOUR_COUNT ) { ... }
if ( colour > 3.14 ) { ... } // Noncompliant, comparison with
```

See

- MISRA C++:2008, 4-5-2 - Expressions with type enum shall not be used as operands to builtin operators other than the subscript operator [ ], the assignment operator =, the equality operators == and !=, the unary & operator, and the relational operators <, <=, >, >=

Available In:

sonarlint ◉◡◉   **sonarcloud** ◇   **sonarqube** ⧉ Developer Edition

---

Stack allocated memory and non-owned memory should not be freed

🐞 Bug

Closed resources should not be accessed

🐞 Bug

Dynamically allocated memory should be released

🐞 Bug

Freed memory should not be used