

- 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

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

Bug

Functions should not be defined with a variable number of arguments

Analyze your code

Code Smell

Critical

cert misra-c++2008 misra-c2004 pitfall

Passing arguments via an ellipsis bypasses the type checking performed by the compiler. Additionally, passing an argument with non-POD class type leads to undefined behavior. Note that the rule specifies "defined" (and not "declared") so as to permit the use of existing library functions.

Noncompliant Code Example

```
void MyPrintf ( char_t * pFormat, ... ) // Noncompliant
{
    // ...
}
```

See

- MISRA C:2004, 16.1 - Functions shall not be defined with a variable number of arguments.
- MISRA C++:2008, 8-4-1 - Functions shall not be defined using the ellipsis notation.
- [CERT, DCL50-CPP](#) - Do not define a C-style variadic function
- [C++ Core Guidelines ES.34](#) Don't define a (C-style) variadic function
- [C++ Core Guidelines F.55](#) Don't use va_arg arguments

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