

Kotlin

Kubernetes

Objective CPHP

PL/I

PL/SQL
Python

RPG RPG

Ruby

Scala

Swift

Terraform

■ Text

Ts TypeScript

T-SQL

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 311 rules & Vulnerability (13)

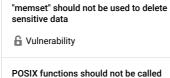
**∰** Bug **74** 

Security Hotspot 18

⊗ Code 206 Smell

O Quick 14

Tags V Search by name...



POSIX functions should not be called with arguments that trigger buffer overflows

■ Vulnerability

XML parsers should not be vulnerable to XXE attacks

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

<table-of-contents> Buç

"pthread\_mutex\_t" should be properly initialized and destroyed

📆 Bug

"pthread\_mutex\_t" should not be consecutively locked or unlocked twice

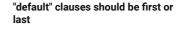
👬 Bug

Functions with "noreturn" attribute should not return

👬 Bug

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

👬 Bug



Analyze your code

Code Smell





switch can contain a default clause for various reasons: to handle unexpected values, to show that all the cases were properly considered.

For readability purpose, to help a developer to quickly find the default behavior of a switch statement, it is recommended to put the default clause at the end of the switch statement. This rule raises an issue if the default clause is not the first or the last one of the switch's cases.

## Noncompliant Code Example

```
switch (param) {
  case 0:
    doSomething();
    break;
  default: // default clause should be the first or last one
    error();
    break;
  case 1:
    doSomethingElse();
    break;
}
```

## Compliant Solution

```
switch (param) {
  case 0:
    doSomething();
    break;
  case 1:
    doSomethingElse();
    break;
  default:
    error();
    break;
}
```

## See

- MISRA C:2004, 15.3 The final clause of a switch statement shall be the default clause
- MISRA C++:2008, 6-4-6 The final clause of a switch statement shall be the default-clause
- MISRA C:2012, 16.4 Every switch statement shall have a default label
- MISRA C:2012, 16.5 A default label shall appear as either the first or the last switch label of a switch statement

Stack allocated memory and nonowned 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

© 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