



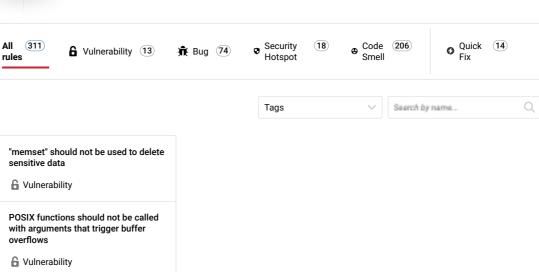
👬 Bug

👬 Bug

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



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



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

"goto" should jump to labels declared later in the same function

Analyze your code



based-on-misra pitfall

Unconstrained use of goto can lead to programs that are extremely difficult to comprehend and analyse. For C++, it can also lead to the program exhibiting unspecified behavior.

However, in many cases a total ban on $\verb"goto"$ requires the introduction of flags to ensure correct control flow, and it is possible that these flags may themselves be less transparent than the goto they replace.

Therefore, the restricted use of goto is allowed where that use will not lead to semantics contrary to developer expectations. "Back" jumps are prohibited, since they can be used to create iterations without using the well-defined iteration statements supplied by the core language.

Noncompliant Code Example

```
int f() {
 int j = 0;
L1:
 ++j;
 if (10 == j) {
  goto L2;
                  // forward jump ignored
 }
 // ...
 goto L1;
                  // Noncompliant
T.2:
 return ++j;
```

Compliant Solution

```
int f() {
for (int j = 0; j < 11; j++) {
   // ...
 return ++j;
```

See

- MISRA C++:2008, 6-6-2 The goto statement shall jump to a label declared later in the same function body
- MISRA C:2012, 15.2 The goto statement shall jump to a label declared later in the same function

Available In:

sonarlint ⊖ | sonarcloud ☆ | sonarqube | Developer Edition

© 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