Secrets
SAP ABAP
APEX Apex
C C
C++ C++
CloudFormation
COBOL COBOL
C# C#
CSS CSS
Flex
GO Go
HTML HTML
Java
JS JavaScript
Kotlin
Kubernetes
Objective C
PHP PHP
PL/I PL/I
PL/SQL PL/SQL
Python
RPG RPG
Ruby
Scala
Swift
Terraform
Text
TS TypeScript
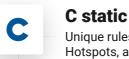T-SQL
VB VB.NET
VB6 VB6
XML XML

C static code analysis

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

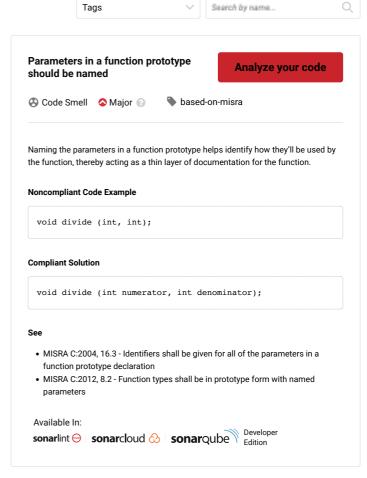All rules **311** | 🔒 Vulnerability **13** | 🐛 Bug **74** | 🛡 Security Hotspot **18** | ⊙ Code Smell **206** | ⚡ Quick Fix **14**

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

Parameters in a function prototype should be named

Analyze your code

⊙ Code Smell ⦿ Major ⍰ 🏷 based-on-misra

Naming the parameters in a function prototype helps identify how they'll be used by the function, thereby acting as a thin layer of documentation for the function.

Noncompliant Code Example

```
void divide (int, int);
```

Compliant Solution

```
void divide (int numerator, int denominator);
```

See

- MISRA C:2004, 16.3 - Identifiers shall be given for all of the parameters in a function prototype declaration
- MISRA C:2012, 8.2 - Function types shall be in prototype form with named parameters

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