



ABAP

Apex

С

C++

CloudFormation

COBOL

C#

CSS

Flex

Go **GO**

5 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



Objective C static code analysis

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

ΑII 315 6 Vulnerability (10) rules

R Bug (75)

• Security Hotspot

⊗ Code (212)

O Quick 13 Fix

Analyze your code

Tags

Octal values should not be used

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

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

Stack allocated memory and nonowned memory should not be freed

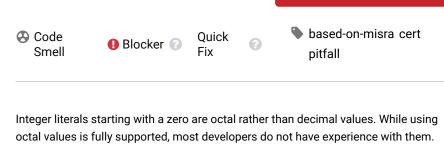
📆 Bug

Closed resources should not be accessed

📆 Bug

Dynamically allocated memory should be released

👬 Bug



They may not recognize octal values as such, mistaking them instead for decimal

Hexadecimal literals (0xdeadbeef) and binary literals (0b0101'0110'00011, available since C++14), on the other hand, have a clear marker (0x or 0b) and can be used to define the binary representation of a value.

Character literals starting with \ and followed by one to three digits are octal escaped literals. Character literals starting with \x and followed by one or more hexits are hexadecimal escaped literals, and are usually more readable.

Noncompliant Code Example

// Noncompliant. myNumber will hold 8, int mvNumber = 010: char myChar = '\40'; // Noncompliant. myChar will hold 32 rat

Compliant Solution

int myNumber = 8; // Use decimal when representing the value int myNumber = 0b1000; // Use binary or hexadecimal for a bit char myChar = $'\x20'$; // Use hexadecimal char myChar = '\n'; // Use the common notation if it exists f

Exceptions

- Octal values have traditionally been used for user permissions in Posix file systems, and this rule will ignore octal literals used in this context.
- '\0' is a common notation for a null character, so the rule ignores it.

See

- MISRA C:2004, 7.1 Octal constants (other than zero) and octal escape sequences shall not be used.
- MISRA C++:2008, 2-13-2 Octal constants (other than zero) and octal escape sequences (other than "\0") shall not be used
- MISRA C:2012, 7.1 Octal constants shall not be used
- CERT, DCL18-C. Do not begin integer constants with 0 when specifying a
- CERT, DCL50-J. Use visually distinct identifiers

Available In:

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

Freed memory should not be used
Memory locations should not be released more than once
Memory access should be explicitly bounded to prevent buffer overflows • Bug
Printf-style format strings should not lead to unexpected behavior at runtime
Recursion should not be infinite
Resources should be closed
n Bug
Hard-coded credentials are security- sensitive Security Hotspot
Hard-coded credentials are security- sensitive
Hard-coded credentials are security- sensitive Security Hotspot "goto" should jump to labels declared later in the same function