



Flex

Go =GO

HTML 5

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

All (315) 6 Vulnerability 10 rules

"memset" should not be used to delete

POSIX functions should not be called with arguments that trigger buffer

Function-like macros should not be

The address of an automatic object

object that may persist after the first

"pthread_mutex_t" should be unlocked

in the reverse order they were locked

"pthread_mutex_t" should be properly

"pthread_mutex_t" should not be

consecutively locked or unlocked

Functions with "noreturn" attribute

"memcmp" should only be called with pointers to trivially copyable types

Stack allocated memory and nonowned memory should not be freed

Closed resources should not be

Dynamically allocated memory should

initialized and destroyed

should not be assigned to another

object has ceased to exist

invoked without all of their arguments

sensitive data

overflows

📆 Bug

📆 Bug

📆 Bug

📆 Bug

twice

📆 Bug

📆 Bug

📆 Bug

📆 Bug

accessed

📆 Bug

be released

📆 Bug

should not return

with no padding

Vulnerability

♠ Vulnerability

R Bug (75)

o Security Hotspot

Tags

preprocessor directives harder to spot.

⊗ Code (212)

O Quick 13 Fix

be indented

Preprocessor directives should not

Analyze your code

convention pitfall

Search by name...

Indenting preprocessor directives reduces the code readability, because it make

Noncompliant Code Example

void optimal() /* Noncompliant - hard to spot */ #if INTEL specificIntelStuff(); /* Noncompliant - hard to spot */ #endif }

Compliant Solution

void optimal() #if INTEL /* Compliant */ specificIntelStuff(); #endif /* Compliant */

Available In:

sonarcloud 🐼 | sonarqube

© 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 Recursion should not be infinite Bug Recursion should not be infinite Bug Resources should be closed Bug Resources should be closed Code Smell Switch labels should not be nested inside non-switch blocks Code Smell Memory access should be explicitly bounded to prevent buffer overflows Replication should not lead to unexpected behavior at runtime Bug Recursion should not be infinite Security Bug Resources should be closed Code Smell Switch labels should not be nested inside non-switch blocks Code Smell	
Memory locations should not be released more than once	Freed memory should not be used
released more than once ## Bug Memory access should be explicitly bounded to prevent buffer overflows ## Bug Printf-style format strings should not lead to unexpected behavior at runtime ## Bug Recursion should not be infinite ## Bug Resources should be closed ## Bug Hard-coded credentials are security-sensitive ## Security Hotspot "goto" should jump to labels declared later in the same function ## Code Smell Only standard forms of the "defined" directive should be used ## Code Smell Switch labels should not be nested inside non-switch blocks	₩ Bug
Memory access should be explicitly bounded to prevent buffer overflows Bug Printf-style format strings should not lead to unexpected behavior at runtime Bug Recursion should not be infinite Bug Resources should be closed Bug Hard-coded credentials are security-sensitive Security Hotspot "goto" should jump to labels declared later in the same function Code Smell Only standard forms of the "defined" directive should be used Code Smell Switch labels should not be nested inside non-switch blocks	
Printf-style format strings should not lead to unexpected behavior at runtime	∰ Bug
Printf-style format strings should not lead to unexpected behavior at runtime Bug Recursion should not be infinite Bug Resources should be closed Bug Hard-coded credentials are security-sensitive Security Hotspot "goto" should jump to labels declared later in the same function Code Smell Only standard forms of the "defined" directive should be used Code Smell Switch labels should not be nested inside non-switch blocks	
lead to unexpected behavior at runtime Bug Recursion should not be infinite Bug Resources should be closed Bug Hard-coded credentials are security-sensitive Security Hotspot "goto" should jump to labels declared later in the same function Code Smell Only standard forms of the "defined" directive should be used Code Smell Switch labels should not be nested inside non-switch blocks	Rug
Recursion should not be infinite Resources should be closed Bug Hard-coded credentials are security- sensitive Security Hotspot "goto" should jump to labels declared later in the same function Code Smell Only standard forms of the "defined" directive should be used Code Smell Switch labels should not be nested inside non-switch blocks	lead to unexpected behavior at
Resources should be closed Bug Hard-coded credentials are security- sensitive Security Hotspot "goto" should jump to labels declared later in the same function Code Smell Only standard forms of the "defined" directive should be used Code Smell Switch labels should not be nested inside non-switch blocks	Rug
Resources should be closed **Bug Hard-coded credentials are security- sensitive **Security Hotspot "goto" should jump to labels declared later in the same function **Code Smell Only standard forms of the "defined" directive should be used **Code Smell Switch labels should not be nested inside non-switch blocks	Recursion should not be infinite
Hard-coded credentials are security- sensitive Security Hotspot "goto" should jump to labels declared later in the same function Code Smell Only standard forms of the "defined" directive should be used Code Smell Switch labels should not be nested inside non-switch blocks	∰ Bug
Hard-coded credentials are security- sensitive Security Hotspot "goto" should jump to labels declared later in the same function Code Smell Only standard forms of the "defined" directive should be used Code Smell Switch labels should not be nested inside non-switch blocks	Resources should be closed
sensitive Security Hotspot "goto" should jump to labels declared later in the same function Code Smell Only standard forms of the "defined" directive should be used Code Smell Switch labels should not be nested inside non-switch blocks	n Bug
"goto" should jump to labels declared later in the same function Code Smell Only standard forms of the "defined" directive should be used Code Smell Switch labels should not be nested inside non-switch blocks	
Iater in the same function Code Smell Only standard forms of the "defined" directive should be used Code Smell Switch labels should not be nested inside non-switch blocks	Security Hotspot
Only standard forms of the "defined" directive should be used Code Smell Switch labels should not be nested inside non-switch blocks	
directive should be used Code Smell Switch labels should not be nested inside non-switch blocks	Code Smell
Switch labels should not be nested inside non-switch blocks	
inside non-switch blocks	
Code Smell	