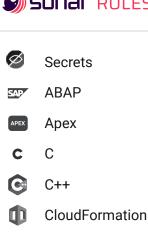
O Quick 13 Fix

Search by name...





COBOL COBOL

C# C#

∃ css

X Flex

€60 Go

THTML

Java

Js JavaScript

Kotlin

Kubernetes

Ó Objective C

PHP

PL/I

PL/SQL

Python

RPG RPG

Ruby

Scala

Swift

Terraform

Text

Ts TypeScript

T-SQL

VB VB.NET

VB6 VB6

XML XML



Objective C static code analysis

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

Tags

"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

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

twice

📆 Bug

📆 Bug

📆 Bug

📆 Bug

accessed

📆 Bug

be released

📆 Bug

should not return

with no padding

Blocking functions should not be Analyze your code called inside critical sections symbolic-execution When entering a critical section the program holds a lock that can be hold by a single entity at a time. It is then recommended that the critical section is reduced to the minimum required and executed without any interruption. **Noncompliant Code Example** void f() { std::mutex m; // ... m.lock(); // ... sleep(1); // Noncompliant m.unlock(); Available In: sonarcloud 🚳 sonarqube Developer

© 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	