






-  Secrets
-  ABAP
-  Apex
-  C
-  C++
-  CloudFormation
-  COBOL
-  C#
-  CSS
-  Flex
-  Go
-  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

- All rules** 315
-  Vulnerability 10
-  Bug 75
-  Security Hotspot 18
-  Code Smell 212
-  Quick Fix 13


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

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

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

 Bug

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

Type specifiers should be listed in a standard order

Analyze your code

 Code Smell  Minor   cppcoreguidelines

Shared coding conventions allow teams to collaborate efficiently. This rule checks that type specifiers always appear in the following order:

- typedef
- type name, spelling of built-in types with more than one type-specifier:
 - signedness - signed or unsigned
 - last single type-specifier or
 - short int
 - long int
 - long long int
 - long double

Since the positioning of the `const` keyword is controversial, this rule does not check it.

Noncompliant Code Example

```
int typedef T;

double long d;
char unsigned ch;
long signed int i;
```

Compliant Solution

```
typedef int T;

long double d;
unsigned char ch;
signed long int i;
```

See

- [C++ Core Guidelines NL.26](#) - Use conventional `const` notation

Available In:

sonarcloud  | **sonarqube**  Developer Edition

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