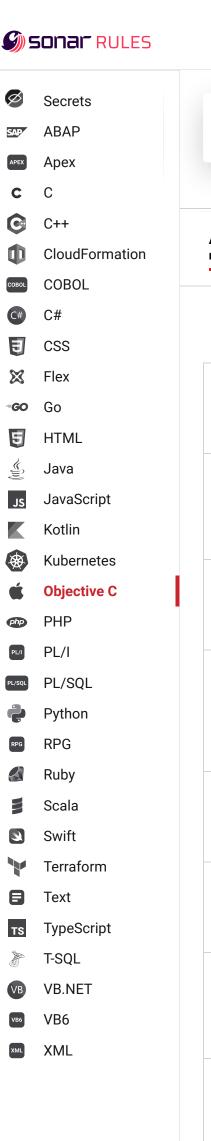
O Quick 13 Fix





should not return

with no padding

"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

📆 Bug

📆 Bug

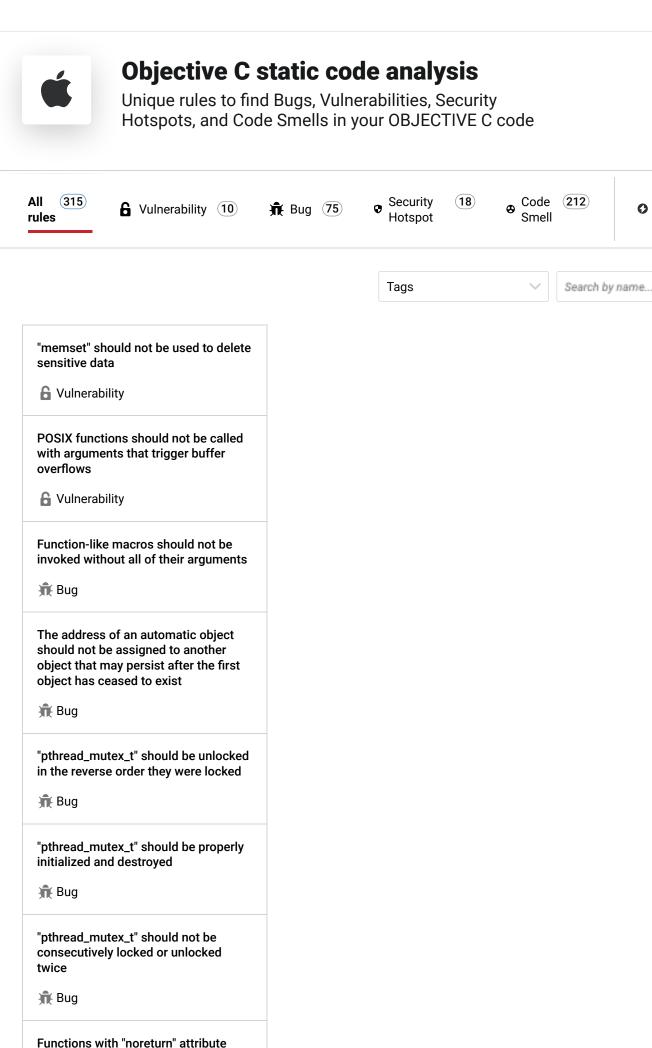
📆 Bug

accessed

📆 Bug

be released

📆 Bug



Freed memory should not be used

🗰 Bug

Memory locations should not be 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 securitysensitive

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

Code Smell

## Setting capabilities is securitysensitive

Analyze your code





Setting capabilities can lead to privilege escalation.

Linux capabilities allow you to assign narrow slices of root's permissions to files or processes. A thread with capabilities bypasses the normal kernel security checks to execute high-privilege actions such as mounting a device to a directory, without requiring (additional) root privileges.

### **Ask Yourself Whether**

Capabilities are granted:

- To a process that does not require all capabilities to do its job.
- To a not trusted process.

There is a risk if you answered yes to any of those questions.

## **Recommended Secure Coding Practices**

Capabilities are high privileges, traditionally associated with superuser (root), thus make sure that the most restrictive and necessary capabilities are assigned to files and processes.

### **Sensitive Code Example**

When setting capabilities:

```
cap_t caps = cap_init();
cap_value_t cap_list[2];
cap list[0] = CAP FOWNER;
cap_list[1] = CAP_CHOWN;
cap_set_flag(caps, CAP_PERMITTED, 2, cap_list, CAP_SET);
cap_set_file("file", caps); // Sensitive
cap_set_fd(fd, caps); // Sensitive
cap_set_proc(caps); // Sensitive
capsetp(pid, caps); // Sensitive
capset(hdrp, datap); // Sensitive: is discouraged to be used
```

When setting SUID/SGID attributes:

```
chmod("file", S_ISUID|S_ISGID); // Sensitive
fchmod(fd, S_ISUID|S_ISGID); // Sensitive
```

# See

- OWASP Top 10 2021 Category A1 Broken Access Control
- OWASP Top 10 2017 Category A5 Broken Access Control
- MITRE, CWE-250 Execution with Unnecessary Privileges
- MITRE, CWE-266 Incorrect Privilege Assignment
- False Boundaries and Arbitrary Code Execution
- Linux manual page capabilities(7)

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