






-  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



## C++ static code analysis

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

- All rules** 578
-  Vulnerability 13
-  Bug 111
-  Security Hotspot 18
-  Code Smell 436
-  Quick Fix 68


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

XML parsers should not be vulnerable to XXE attacks

 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

Assigning to an optional should directly target the optional

 Bug

Result of the standard remove algorithms should not be ignored

 Bug

"std::scoped\_lock" should be created with constructor arguments

 Bug

Objects should not be sliced

 Bug

Immediately dangling references should not be created

 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
<b>"std::move" and "std::forward" should not be confused</b>  Bug
<b>A call to "wait()" on a "std::condition_variable" should have a condition</b>  Bug
<b>A pointer to a virtual base class shall only be cast to a pointer to a derived class by means of dynamic_cast</b>  Bug
<b>Functions with "noreturn" attribute should not return</b>  Bug
<b>RAII objects should not be temporary</b>  Bug
<b>"memcmp" should only be called with pointers to trivially copyable types with no padding</b>  Bug
<b>"memcpy", "memmove", and "memset" should only be called with pointers to trivially copyable types</b>  Bug
<b>"std::auto_ptr" should not be used</b>  Bug
<b>Destructors should be "noexcept"</b>  Bug

Setting capabilities is security-sensitive

Analyze your code

Security HotspotMajor?cwe owasp

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