



ABAP

Apex

С

C++

CloudFormation

COBOL

C#

CSS

Flex

Go **GO**

5 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

578 ΑII 6 Vulnerability 13 rules

R Bug (111)

• Security Hotspot **⊗** Code (436)

Quick 68 Fix

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

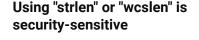
"pthread_mutex_t" should be unlocked in the reverse order they were locked

📆 Bug

"pthread_mutex_t" should be properly initialized and destroyed

📆 Bug

"pthread_mutex_t" should not be consecutively locked or unlocked twice



Analyze your code

cwe cert

The function size t strlen(const char *s) measures the length of the string s (excluding the final null character).

The function size_t wcslen(const wchar_t *s) does the same for wide characters, and should be used with the same guidelines.

Similarly to many other functions in the standard C libraries, strlen and wcslen assume that their argument is not a null pointer.

Additionally, they expect the strings to be null-terminated. For example, the 5-letter string "abcde" must be stored in memory as "abcde\0" (i.e. using 6 characters) to be processed correctly. When a string is missing the null character at the end, these functions will iterate past the end of the buffer, which is undefined behavior.

Therefore, string parameters must end with a proper null character. The absence of this particular character can lead to security vulnerabilities that allow, for example, access to sensitive data or the execution of arbitrary code.

Ask Yourself Whether

- There is a possibility that the pointer is null.
- There is a possibility that the string is not correctly null-terminated.

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

Recommended Secure Coding Practices

• Use safer functions. The C11 functions strlen_s and wcslen_s from annex K handle typical programming errors.

Note, however, that they have a runtime overhead and require more code for error handling and therefore are not suited to every case.

- Even if your compiler does not exactly support annex K, you probably have access to similar functions.
- If you are writing C++ code, using std::string to manipulate strings is much simpler and less error-prone.

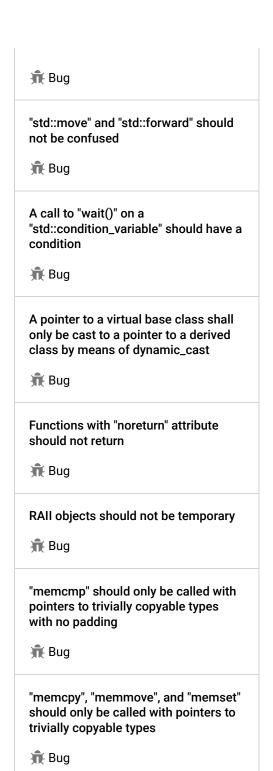
Sensitive Code Example

```
size t f(char *src) {
  char dest[256];
  strncpy(dest, src, sizeof dest); // Truncation may happen
  return strlen(dest); // Sensitive: "dest" will not be null-
```

Compliant Solution

```
size_t f(char *src) {
 char dest[256];
 strncpy(dest, src, sizeof dest); // Truncation may happen
 dest[sizeof dest - 1] = 0;
 return strlen(dest); // Compliant: "dest" is guaranteed to
```

- MITRE, CWE-120 Buffer Copy without Checking Size of Input ('Classic Buffer
- CERT, STR07-C. Use the bounds-checking interfaces for string manipulation



"std::auto_ptr" should not be used

Destructors should be "noexcept"

📆 Bug

📆 Bug



© 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