



ABAP

Apex

C

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

ΑII 578 6 Vulnerability 13 rules

R Bug (111)

• Security Hotspot ⊗ Code (436)

Quick 68 Fix

Tags

Search by name...

Using publicly writable directories

Analyze your code

is security-sensitive

cwe symbolic-execution owasp

Operating systems have global directories where any user has write access. Those folders are mostly used as temporary storage areas like /tmp in Linux based systems. An application manipulating files from these folders is exposed to race conditions on filenames: a malicious user can try to create a file with a predictable name before the application does. A successful attack can result in other files being accessed, modified, corrupted or deleted. This risk is even higher if the application runs with elevated permissions.

In the past, it has led to the following vulnerabilities:

- CVE-2012-2451
- CVE-2015-1838

This rule raises an issue whenever it detects a hard-coded path to a publicly writable directory like /tmp (see examples bellow). It also detects access to environment variables that point to publicly writable directories, e.g., TMP and TMPDIR.

- /tmp
- /var/tmp
- /usr/tmp
- /dev/shm
- /dev/mqueue
- /run/lock • /var/run/lock
- /Library/Caches
- /Users/Shared
- /private/tmp
- /private/var/tmp
- \Windows\Temp • \Temp
- \TMP

Ask Yourself Whether

- · Files are read from or written into a publicly writable folder
- The application creates files with predictable names into a publicly writable folder

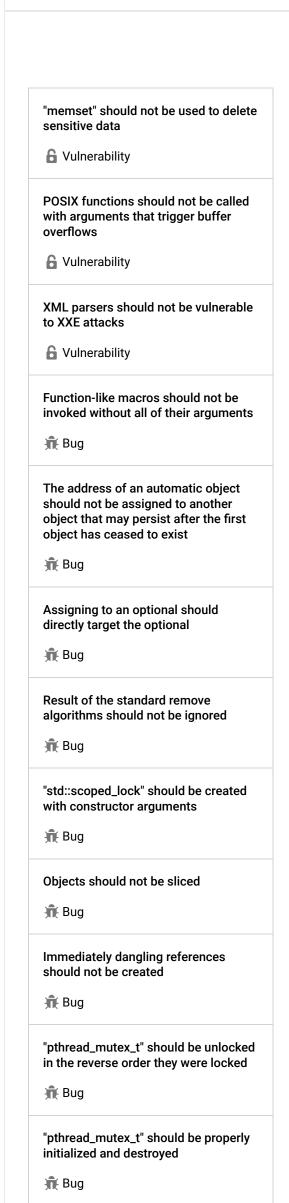
There is a risk if you answered yes to any of those questions.

Recommended Secure Coding Practices

- Use a dedicated sub-folder with tightly controlled permissions
- Use secure-by-design APIs to create temporary files. Such API will make sure:
 - o The generated filename is unpredictable
 - The file is readable and writable only by the creating user ID
 - The file descriptor is not inherited by child processes
 - The file will be destroyed as soon as it is closed

Sensitive Code Example

```
#include <cstdio>
// ...
void f() {
 FILE * fp = fopen("/tmp/temporary_file", "r"); // Sensitive
```



"pthread_mutex_t" should not be consecutively locked or unlocked



"std::move" and "std::forward" should not be confused



A call to "wait()" on a "std::condition_variable" should have a condition



A pointer to a virtual base class shall only be cast to a pointer to a derived class by means of dynamic_cast



Functions with "noreturn" attribute should not return



RAII objects should not be temporary



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



"memcpy", "memmove", and "memset" should only be called with pointers to trivially copyable types



"std::auto_ptr" should not be used

```
📆 Bug
```

Destructors should be "noexcept"

```
📆 Bug
```

```
#include <cstdio>
#include <cstdlib>
#include <sstream>
// ...
void f() {
  std::stringstream ss;
  ss << getenv("TMPDIR") << "/temporary_file"; // Sensitive</pre>
  FILE * fp = fopen(ss.str().c_str(), "w");
}
```

Compliant Solution

```
#include <cstdio>
#include <cstdlib>
// ...
void f() {
 FILE * fp = tmpfile(); // Compliant
```

See

- OWASP Top 10 2021 Category A1 Broken Access Control
- OWASP Top 10 2017 Category A5 Broken Access Control
- OWASP Top 10 2017 Category A3 Sensitive Data Exposure
- MITRE, CWE-377 Insecure Temporary File
- MITRE, CWE-379 Creation of Temporary File in Directory with Incorrect Permissions
- OWASP, Insecure Temporary File

Available In:







© 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