Avoiding insecure C++

How to avoid common C++ security vulnerabilities

Aaron Ballman
CERT
SEI/CMU
Pittsburgh, USA
aballman@cert.org

David Svoboda
CERT
SEI/CMU
Pittsburgh, USA
svoboda@cert.org

***Abstract*—Introducing the SEI CERT C++ Coding Standard, a new, security-focused coding standard for the C++ programming language that focuses on the security of applications written with modern versions of C++.**

Keywords—C++; secure coding; coding standard; coding guidelines

I. Introduction

Writing secure C++ code is hard. C++11 and C++14 have added new facilities that change the way programmers write C++ code with the introduction of features like lambdas and concurrency. However, there are few resources describing how these new facilities also increase the number of ways in which security vulnerabilities can be introduced into a program, or how to avoid using these facilities insecurely.

Previous secure coding efforts, including the SEI CERT C Coding Standard [5] and SEI CERT Oracle Coding Standard for Java [3], have proven successful in aiding programmers with identifying possible insecure code in C and Java, but do not provide sufficient information to cover C++. Other efforts, such as MISRA C++:2008 [4] and the C++ Core Guidelines [2], subset the C++ language and do not focus on security. In this presentation, we introduce the SEI CERT C++ Coding Standard [1] by discussing a sample of the areas of C++ that can result in security vulnerabilities and describing our methodology for discovering and preventing these vulnerabilities. The talk will reference C++-specific content that can be found in greater detail on the CERT Secure Coding Wiki, where these rules are being actively developed and maintained.

The SEI CERT C++ Coding Standard provides rules for secure coding in the C++ programming language. The goal of these rules is to develop safe, reliable, and secure systems, for example by eliminating undefined behaviors that can lead to exploitable vulnerabilities. Conformance to the coding rules defined in this standard are necessary (but not sufficient) to ensure the safety, reliability, and security of software systems developed in the C++ programming language. The application of this coding standard will result in high-quality systems that are reliable, robust, and resistant to attack.

Each guideline consists of a title, a description, and a noncompliant code example and compliant solutions. The title is a concise, but sometimes imprecise, description of the guideline. The description specifies the normative requirements of the rule. The noncompliant code examples are examples of code that would constitute a violation of the guideline. The accompanying compliant solutions demonstrate equivalent code that does not violate the guideline or any other rules in the coding standard.

A well-documented and enforceable coding standard is an essential element of coding in the C++ programming language. Coding standards encourage programmers to follow a uniform set of rules determined by the requirements of the project and organization rather than by the programmer's familiarity. Once established, these standards can be used as a metric to evaluate source code (using manual or automated processes).

ACKNOWLEDGMENT

DM-0003965

References

[1] A. Ballman. The SEI CERT C++ Coding Standard. unpublished. https://www.securecoding.cert.org/confluence/display/cplusplus

[2] C++ Core Guidelines. n.d. Retrieved June 23, 2016, from https://github.com/isocpp/CppCoreGuidelines/blob/master/CppCoreGuidelines.md

[3] F. Long, D. Mohindra, R. Seacord, D. Sutherland, D. Svoboda. The CERT Oracle secure coding standard for Java. Addison-Wesley, 2012 https://www.securecoding.cert.org/confluence/display/java/

[4] MISRA-C++ 2008: Guidelines for the Use of the C++ Language in Critical Systems. 2008.

[5] Seacord, Robert. The CERT C Coding Standard, 2 nd ed. Addison-Wesley, 2014 https://www.securecoding.cert.org/confluence/display/c