Blaze Halderman

CS405-SNHU

11/26/2022

Module Five Static Analysis

The process of examining the errors included first understanding and implementing the CppCheck plugin/application. This meant downloading the application and running it properly in order to show the errors/vulnerabilities found within the code. The vulnerabilities/errors found within the code involve 3 errors (vulnerabilities), 2 performance related issues, 12 style issues, and 7 warnings. The 3 errors involved Id autoVariables, CWE: 562, Dangerous assignment assigning local auto-variable, Id: throwInNoexceptFunction, CWE:398, exception thrown in function declared not to throw exceptions, and Id invalidContainer, CWE: 664, using iterator to local container may be invalid.

In order to mitigate these three issues, the first issue could be resolved by better implementing a new variable which stored a copy value and was assigned after use, the second issue could be resolved by creating a custom exception and throwing/catching it properly, and finally validate that the items vector has a length and that it can be iterated over, before creating an iterator. Also, the warning errors can cause the application to fail in the report. Using assert statements for assignments is dangerous and is also not possible since the comparison is using an integer with a Boolean. Ensuring that types are the same prior to assertions can break an application. Ensuring all variables are verified properly and can be accessed (memory location exists) prior to calling them will protect the program from crashing or encountering vulnerabilities.

On line 66, Id arrayIndexOutOfBoundsCond, CWE: 788, the warning of the count being set to 1000 could potentially mean an out of bounds issue, and could lead to a segmentation fault. If the count value is any other value then the assigned value of greater than 10 (allocated size of the buffer), then a segmentation fault will occur. This is a summary of what Cppcheck found along with my analysis within the code. Cppcheck does a great job making the user aware of potential risks versus errors or simple styling issues and provides explanations for all findings.

The only issue CPP found was related to potential buffer overflow of the buf variable, and the uninitialized variable x. This goes to show how powerful the Cppcheck tool is and how analysis tools can provide great context and evidence for where an application is vulnerable. The use of an IDE is to provide a tool to write code, and having a separate tool for analyzing code can prevent vulnerabilities and dangerous breaches from occurring.