Daniel Coffey

5-3 Activity: Static Code Analysis

CS 405 Secure Coding

02/10/2022

This activity is a comparison of errors and warnings between Visual Studio and CppCheck. After completing the static code analysis, I have concluded that CppCheck is more efficient than the analysis that is included in Visual Studio. Visual Studio’s analysis had a result of zero errors and five warnings. These warnings were as follows:

Code – C26495: Variable ‘A::x’ is unitialized. Always initialize a member variable(type.6).

Code – C6386: Buffer overrun while writing to ‘buf’: the writable size is ‘40’ bytes, but ‘count’ bytes might be written.

Code – C6282: Incorrect operator: assignment of constant in Boolean context. Consider using ‘==’ instead.

Code – C4297: ‘MySpecialType::DontThrow’: function assumed not to throw an exception but does.

Code – C4806: ‘==’ unsafe operation: no value of type ‘bool’ promoted to type ‘int’ can equal the given constant.

There are a lot more errors and warnings from the CppCheck but one important feature is that when you click on an error, it gives you notes that contain more information. Here is an example error from the file:

CWE: 664

Using iterator to local container 'items' that may be invalid.

After clicking on this error there are notes provided that give more details about the process such as:

CWE: 664

Iterator to container is created here.

CWE: 664

Assuming condition is true.

CWE: 664

Assuming condition is true.

CWE: 664

After calling 'erase', iterators or references to the container's data may be invalid.

CWE: 664

Variable created here.

The corresponding line for the errors and notes is in the summary section on the CppCheck. The CppCheck is far more detailed than the analyzer in Visual Studio. Visual Studio provided 0 errors and 5 warnings, whereas CppCheck provided 7 warnings, 3 errors, and 8 style problems. I can’t believe how drastic of a difference there was between the two. CppCheck caught the error throwing error, but Visual Studio only caught that when the program was ran, not analyzed. CppCheck also caught important errors such as:

CWE: 562

Dangerous assignment - the function parameter is assigned the address of a local auto-variable. Local auto-variables are reserved from the stack which is freed when the function ends. So the pointer to a local variable is invalid after the function ends.

CWE: 664

Using iterator to local container 'items' that may be invalid.

You can see from these errors CppCheck has found a way to mitigate the issue. We need to get rid of the dangerous assignment, get rid of the throw error, and fix the iterator because after calling ‘erase’, the iterators or references to the container’s data may be invalid.







