Carnegie Mellon University
Software Engineering Institute

ASSESSING SOFTWARE QUALITY USING A RISK-BASED METHODOLOGY

Jay Marchetti
Michael Riley
June 2023

Overview

On May 12, 2021, President Biden signed the "Executive Order on Improving the Nation's Cybersecurity". This order was part of the U.S. Government's response to the well-publicized attacks on SolarWinds and Microsoft software, as well as attacks targeting critical U.S. infrastructure such as the Colonial Pipeline. These attacks, while widely covered in the media, are among the numerous and ongoing attacks happening on key U.S. assets. President Biden's executive order calls on software suppliers to dramatically improve protective measures within their products to thwart our adversaries. Realizing that today's code quality issues are tomorrow's zero-day (i.e. novel) cybersecurity exploits, the importance of the risks posed by poor code quality are now more salient than ever. The SEI has developed a framework for assessing the risk of poor code quality in embedded systems utilized by the DoD and others. This framework may be applicable for assessing code quality risks in wider U.S. infrastructure components as well.

Code Quality Risks

According to a January 2021 report by The Consortium for Information and Software Quality (CISQ), poor software quality cost the United States economy over $2 trillion in 2020. Particularly for mission- and safety-critical embedded systems, poor code quality entails risks. For example, consider the risks posed when a critical system's code base is hard to test thoroughly, prone to failures and crashes, unmaintainable over its expected lifecycle, hard to extend for new capabilities, exploitable to cyber-attacks, difficult to harvest for engineering reuse, or plagued with numerous latent defects. Given this wide swath of risks, how can one characterize the degree of risk for a particular code base?

Standards

Certainly, existing software standards can help organizations avoid creating high-risk code as part of new, or greenfield, development. These include software quality standards such as ISO/IEC 25010, software safety standards such as DO-178C for avionics systems, and software security standards such the SEI CERT Secure Coding Standards for C, C++, and Java. However, these standards and others also provide a basis for characterizing the risks embodied within any currently existing code under consideration.

Assessing Existing Code

The SEI assesses quality risks in existing code bases for a variety of critical systems. The context most common for these assessments includes code that is specific to mission-, safety-, or security-critical

systems, code intended for embedded systems whose source is written in either C or C++ and often supplied without backing design documentation or access to developers, and code that may be classified or unclassified. Expert SEI code analysts apply multiple static analysis tools, both commercial and open license, and read a targeted subset of the source code as part of the assessment. The goal for SEI assessments is ascertaining *structural code quality*, i.e. the quality of the code itself, rather than *functional code quality*, which addresses how well the code fulfils mission requirements. This allows us to eliminate the functional concerns expressed within common software quality standards such as ISO/IEC 25010.

Code Risk Estimation

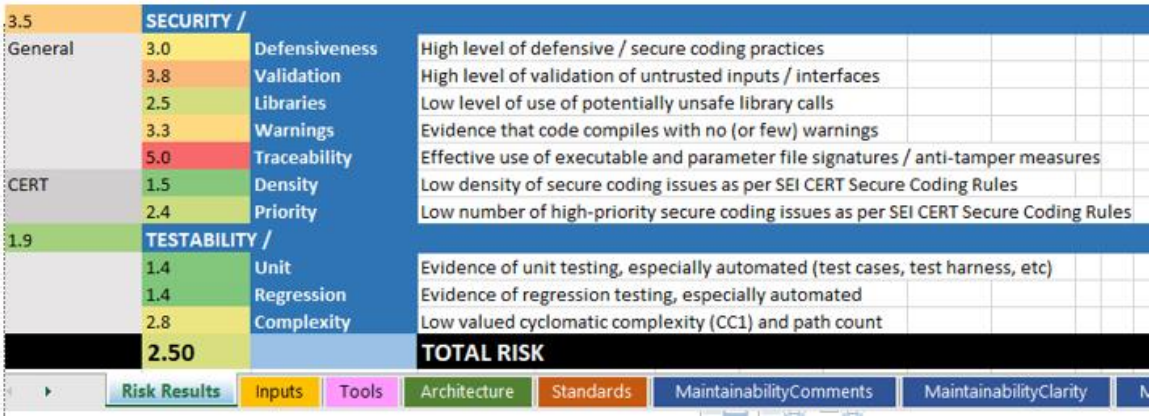
SEI assesses structural code quality risks by centering our analyses on six (6) aspects of code quality: implemented architecture, standards adherence, maintainability, fault tolerance, security, and testability. Each code quality aspect is further decomposed into two (2) or more foci. The six aspects and 31 foci (some grouped as sub-aspects under Maintainability and Security) are shown below in Figure 1.

Figure 1 – A Taxonomy of Structural Code Quality Consideration for Code Risk Estimation

Implementation – The Code Risk Estimation Worksheet

SEI developed an Excel worksheet-based approach to guide analysts through the code risk estimation process. The Code Risk Estimation Worksheet (CREW) presents the analyst with code-level questions aimed at objectifying the risk estimation and consistently directing the analysis. The questions, answerable using static analysis tools and by reading a subset of the source code, are specific to each focus. Answers are combined and a risk score in the closed interval [1.0, 5.0] is automatically calculated

for each focus. These are then rolled up to generate a risk score for each of the six aspects – and, ultimately, to a total risk estimate for the codebase. The foci, aspects, and total risk estimates, all of which utilize the same [1.0, 5.0] rubric, are shown on the CREW Risk Results tab which provides an easy to understand overview of the analysis results. A portion of a typical risk results tab is shown below in Figure 2.



3.5	SECURITY /		
General	3.0	Defensiveness	High level of defensive / secure coding practices
	3.8	Validation	High level of validation of untrusted inputs / interfaces
	2.5	Libraries	Low level of use of potentially unsafe library calls
	3.3	Warnings	Evidence that code compiles with no (or few) warnings
	5.0	Traceability	Effective use of executable and parameter file signatures / anti-tamper measures
CERT	1.5	Density	Low density of secure coding issues as per SEI CERT Secure Coding Rules
	2.4	Priority	Low number of high-priority secure coding issues as per SEI CERT Secure Coding Rules
1.9	TESTABILITY /		
	1.4	Unit	Evidence of unit testing, especially automated (test cases, test harness, etc)
	1.4	Regression	Evidence of regression testing, especially automated
	2.8	Complexity	Low valued cyclomatic complexity (CC1) and path count
	2.50		TOTAL RISK

Risk Results | Inputs | Tools | Architecture | Standards | MaintainabilityComments | MaintainabilityClarity

Figure 2 – A Partial View of the CREW Risk Results Tab

Summary

The Code Risk Estimation Worksheet was developed by SEI for conducting independent, third-party code analyses. CREW encourages stakeholders to view code quality through the lens of risk management, providing a statistical estimate of the high codebase risks areas that can then form a basis for actionable improvements or mitigations. By utilizing CREW, SEI at once enhances the thoroughness, objectivity, consistency, and traceability of critical embedded system code assessments.

Contact Us

Software Engineering Institute
4500 Fifth Avenue, Pittsburgh, PA 15213-2612

Phone: 412/268.5800 | 888.201.4479
Web: www.sei.cmu.edu
Email: info@sei.cmu.edu

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM23-0592

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution.