Views on the Framework for Improving Critical Infrastructure Cybersecurity
CISQ Response to NIST Cybersecurity Framework RFI

Objectives

1. To update the NIST Cybersecurity Framework to enhance the role of structural analysis of source code at both the architectural and modular level as a critical recommended practice.
2. To update the NIST Cybersecurity Framework by referencing two new standards approved by the Object Management Group, the Automated Source Code Security Measure and the Automated Source Code Reliability Measure (hereafter referred to as the CISQ measures).

Overview

This response is submitted on behalf of the Consortium for IT Software Quality (CISQ) to the Request For Information on the NIST Cybersecurity Framework. CISQ is a Special Interest Group of the Object Management Group (OMG) chartered to develop standards for automating the measurement of source code in software-intensive systems. CISQ was jointly formed by OMG and the Software Engineering Institute (SEI) at Carnegie Mellon University.

CISQ is submitting this recommendation for updating the NIST Cybersecurity Framework to strengthen its focus on structural weaknesses in the software of critical infrastructure systems that create the greatest risk to secure and dependable operations. Critical infrastructure systems are vulnerable to unauthorized penetration and undependable operation as long as severe weaknesses exist in the source code. Thus, analysis and measurement of structural weaknesses should be a standard element of any development, maintenance, acquisition, or related process to ensure that critical infrastructure systems operate on secure and dependable software.

CISQ Automated Source Code Measurement Standards

During 2015 four Automated Source Code measures (Reliability, Security, Performance Efficiency, and Maintainability) developed by CISQ were approved as OMG standards. They are the only international standards that measure the structural quality of software-intensive systems based on quantifying specific, named weaknesses in the source code. These measures were constructed using definitions in ISO/IEC 25010, the standard that defines eight software quality characteristics and elaborates them into sub-characteristics. The CISQ measures supplement ISO/IEC 25023, the standard that enumerates measures of the various sub-characteristics. However, ISO/IEC 25023 provides measures primarily at the behavioral level and does not enumerate or measure specific weaknesses in the source code that cause undesirable behaviors. The CISQ measures were designed to be quantified from automated

analysis of architectural and coding weaknesses in source code, since manual review is infeasible for large multi-layer, multi-language, multi-platform systems.

The CISQ measures were developed by the 24 companies that initially joined CISQ, with involvement of experts from the Software Engineering Institute and MITRE Corporation. The CISQ measures were developed as counts of critical weaknesses—weaknesses that the majority of members believed must be fixed. The CISQ measures are currently defined using weaknesses found in IT systems. They will be supplemented in upcoming work with versions focused on specific weaknesses in embedded systems. However, the vast majority of weaknesses enumerated in the CISQ measures are common to both IT and embedded systems. The measures are agnostic to programming language, although the structure of some languages avoid certain weaknesses.

Two of the CISQ measures approved as OMG standards are directly relevant to the NIST Cybersecurity Framework—the Automated Source Code Security Measure[1] and the Automated Source Code Reliability Measure[2]. The Reliability measure was constructed from weaknesses that have been shown to cause outages, data corruption, and excessive recovery time.

The CISQ Security measure was drawn from the Top 25 Common Weaknesses contained in the Common Weakness Enumeration managed by MITRE Corporation on behalf of the cybersecurity community with funding from the Department of Homeland Security. These weaknesses are also known as the 'CWE/SANS Top 25 Most Dangerous Software Errors'. Of these 25 weaknesses, 22 can be detected through static analysis of the source code. These weaknesses include such flaws as SQL injection, cross-site scripting, and buffer overflows. Counting their occurrences in a software-intensive system constitutes the CISQ Security measure. This measure can be normalized by a size measure to provide a density measure useful for benchmarking. Vendors of static analysis tools detect most, but not necessarily all of these weaknesses, and their coverage is increasing yearly.

Using the CISQ Measures

Cybersecurity Objectives. The CISQ measures provide critical assistance to organizations in achieving the primary objectives of the NIST Cybersecurity Framework in the following ways.

1) *Describe the current cybersecurity posture*—a critical aspect of the current cybersecurity posture is identification of the specific risks posed by exploitable weaknesses in the source code of critical systems. The CISQ measures provide a more factual assessment of the current cybersecurity of the software component based on standardized measures of weaknesses in the architecture and code.
2) *Describe their target state for cybersecurity*—a critical aspect of the target state for cybersecurity is specifying the level of risk that can be tolerated and the source code weaknesses that would have to be eliminated as part of achieving and sustaining the target risk profile. The CISQ measures provide a fact-based foundation for expressing

the target cybersecurity a software component based on standardized measures that can identify weaknesses to be eliminated.

3) *Identify and prioritize opportunities for improvement within the context of a continuous and repeatable process*—since traditional testing and penetration analysis, although important, do not detect all the structural weaknesses in a system, CISQ believes it is critical to supplement these traditional techniques with system-level static analysis to identify exploitable weaknesses in the architecture and code that are difficult to detect through other assurance methods.
4) *Assess progress toward the target state*—the presence of cybersecurity weaknesses in the source code of critical infrastructure systems must be analyzed, measured, and regularly tracked as part of accurately assessing progress toward the target state.
5) *Communicate among internal and external stakeholders about cybersecurity risk*—the most effective communication conveys cybersecurity risk through evidence-based measures on the elements creating risk, including valid measures of cybersecurity weaknesses in the source code as enumerated in the CISQ measures.

Cybersecurity Practices. CISQ recommends the following actions be incorporated into development and release practices for critical infrastructure systems.

1) All software critical infrastructure systems must be analyzed at each release for critical security or reliability weaknesses enumerated in the CISQ measures.
2) Security or reliability weaknesses detected in the system by analyzing the CISQ measures should be prioritized and incorporated into a backlog of must-fix weaknesses.
3) Measures of the security and reliability of source code should be incorporated as primary contributors to the assessment and measurement of risk in critical infrastructure systems.
4) Executive management should periodically review the state of their critical infrastructure systems with particular attention to progress on eliminating security and reliability weaknesses.

Fitting Structural Analysis and the CISQ Measures into the Framework

Implementation Tiers. Structural analysis of source code weaknesses in critical infrastructure systems contributes at all tiers of the Cybersecurity Implementation Framework as a risk management process. However, the use of structural analysis as a cybersecurity risk management practice will differ at each tier of the implementation framework as suggested in the following bullets.

- *Tier 1* — some but not necessarily all critical systems will be analyzed for cybersecurity weaknesses in the source code. Implementations of structural analysis will not be consistent across systems and may not be consistently used where implemented. The results of the analysis may or may not be acted upon. For instance, identified weaknesses may not be prioritized for remediation, or measures derived from the

identified weaknesses may not be tracked or incorporated into cybersecurity risk analyses.

- *Tier 2* — management expects structural analysis to be performed on each critical system but has not established a consistent policy requiring it. The organization has not defined a standard practice for conducting structural analysis. Information and measures resulting from structural analyses are not used in consistent ways across systems. As a result of these inconsistencies it is difficult to compare the cybersecurity state of systems internally or against external benchmarks.
- *Tier 3* — structural analysis is required by policy on all critical infrastructure systems and a process for incorporating structural analysis into the standard system development and maintenance process has been defined. Tailoring guidelines have been developed for adjusting practices to the characteristics of each system and its cybersecurity environment. Standardized metrics such as the CISQ measures have been adopted for analysis and reporting. Practices for translating results into prioritized remedial actions and tracking progress toward cybersecurity targets are integrated into recurring release processes. Data emerging from structural analyses are consistent across systems and can be incorporated into a repository for measuring and managing cybersecurity risk at the organizational level.
- *Tier 4* — structural analysis results and measures are periodically analyzed to identify improvements that can be made to structural analysis processes, as well as the interpretation and use of the measures. Structural quality measures are also analyzed to identify improvements in system development and maintenance practices that reduce or eliminate the creation or injection of cybersecurity weaknesses into the source code. Progress and benefits of improvements are tracked using structural analysis measures.

Organizational Tiers. Information resulting from structural analysis of the source code of critical infrastructure systems can be used at different tiers of the organization as suggested in the following bullets.

- *Executive Level* — executives establish organizational cybersecurity risk tolerance objectives that can include specific objectives measured against source code weaknesses in each critical infrastructure system. They monitor progress toward these objectives with risk data provided by the business/process level, and may track remediation progress and remaining risk on particularly sensitive or troubled systems.
- *Business/Process Level* — the business/process level translates executive decisions regarding risk tolerance into cybersecurity risk profiles for each critical infrastructure system. These profiles include measurable objectives for eliminating cybersecurity weaknesses in the system. Progress toward these objectives is tracked with structural analysis measures received from implementation level managers. These results can be used to allocate resources to systems exhibiting the highest cybersecurity risk. Structural analysis measures are also aggregated with other risk information into the organizational system risk profile that is communicated to executives.

- *Implementation/Operations Level* — structural analysis results are used by the those managing each critical infrastructure system to identify cybersecurity weaknesses within the system, assess its current level of risk and the distance from a target objective, prioritize remediation actions, track progress toward the target objective, and communicate risk information to stakeholders, especially those at the business/process level. Developers use the results to locate prioritized weaknesses in the system and eliminate them. Developers can also use the results to learn from mistakes, improve secure architectural and coding practices, and launch iterations with information about frequent weaknesses to avoid. Operations staff can use the information for understanding which attack patterns may be able to exploit identified weaknesses in the system that have yet to be eliminated.

Cybersecurity Implementation Processes. Structural analysis information fits within several steps for establishing or improving a cybersecurity program.

- *Step 3: Create a Current Profile* — a critical component of a current risk profile will be whether the existing system development and maintenance processes include practices for analyzing, measuring, prioritizing, and eliminating structural cybersecurity weaknesses in the source code.
- *Step 4: Conduct a Risk Assessment* — a cybersecurity risk assessment should include analysis and measurement of the structural weaknesses in the source code of critical infrastructure systems that can be exploited during an attack or result in undependable operation. Since these weaknesses are a primary source of risk, their detection and measurement should be a primary component of the risk profile emerging from the assessment. The software risk indicators incorporated into the risk profile should be backed by a measured assessment against relevant standards such as the CISQ measures.
- *Step 5: Create a Target Profile* — a detailed target profile should include descriptions of the type, number, density, location, or other characteristics of cybersecurity weaknesses that can be tolerated within the acceptable risk profile of a system, and the length of time allowed for removal for those weaknesses that must be eliminated to achieve the target profile. These target profiles will be specific to each critical infrastructure system and should include one or more source code-based measures for evaluating progress.
- *Step 6: Determine, Analyze, and Prioritize Gaps* — structural analysis will provide measures and input for determining the gap between the current and target profiles for the cybersecurity of a critical infrastructure system's source code. These measures and inputs enable prioritization of the various source code remediation actions that must be taken to achieve the overall risk profile for a critical infrastructure system.
- *Step 7: Implement the Action Plan* — structural analysis and measurement of cybersecurity weaknesses provide detailed input for developing action plans that include eliminating cybersecurity weaknesses to achieve the target profile. The plan should include specific actions to eliminate the types of weaknesses incorporated into

the CISQ measures. Progress toward achieving the target profile for the cybersecurity of the source code can then be measured and tracked at each milestone of the plan.

The CISQ measures provide source code-based information and measures that can be used to communicate the technical risk in critical systems and how that risk is being reduced by progress on the cybersecurity action plan. Since this risk information is grounded in measured fact rather than judgment, it will provide greater credibility to the communication. Information based on cybersecurity weaknesses does not have privacy implications, although organizations will typically treat the data as proprietary to their internal risk management activities.

Referencing CISQ Measures in the Framework Core

The CISQ measures provide relevant standards to be referenced for incorporating the analysis and measurement of cybersecurity weaknesses in source code into the following subcategories of the NIST Cybersecurity Framework.

- ***ID.GV-1****: Organizational information security policy is established* — the policy should include a statement about eliminating cybersecurity weaknesses in the source code of critical systems since this is a primary source of exploitation and unreliable operation, as well as providing fact-based measures for assessing risk.
- ***ID.RA-1****: Asset vulnerabilities are identified and documented* — the CISQ Security measure provides explicit guidance for the vulnerabilities in the source code that should be identified based on the SANS/CWE Top 25 Most Dangerous Software Errors, a list drawn from industry experience. The CISQ Reliability measure provides additional guidance on vulnerabilities that can result in outages, data corruption, unpredictable behavior, and other operational maladies.
- ***ID.RA-5****: Threats, vulnerabilities, likelihoods, and impacts are used to determine risk* — the CISQ measures provide objective quantification of source code weaknesses that can be used in expressing the software risk in critical infrastructure systems.
- ***ID.RA-6****: Risk responses are identified and prioritized* — analyzing source code against the CISQ measures yields a list of explicit weaknesses that can be prioritized for elimination during current or future releases. Organizations can refer to the Common Weakness Enumeration Repository for detailed information about each weakness type and potential methods for elimination.
- ***ID.RM-2****: Organizational risk tolerance is determined and clearly expressed* — the CISQ measures provide fact-based results whose scores can be represented as absolute counts, density measures, sigma levels, or other quantifications that can be used for expressing a level of risk tolerance for cybersecurity weaknesses in the source code.
- ***PR.AT-6:*** *Information systems personnel understand specific cybersecurity weaknesses in source code* — consider adding a PR.AT-6 where system developers are trained in cybersecurity weaknesses they should avoid, detect, and eliminate. The CISQ measures provide a list of weaknesses that should be incorporated into training. This subcategory could be expanded and made more generic to cover other technical personnel with

standards enumerating weaknesses in their domain such as hardware, networking, or communications.

- ***PR.DS-1****: Data-at-rest is protected* — the CISQ measures include weaknesses that can expose data to unauthorized access or corruption.
- ***PR.DS-6****: Integrity checking mechanisms are used to verify software, firmware, and information integrity* — structural analysis at both the architectural and modular level should be included as mechanisms for verifying software integrity. The CISQ measures enumerate weaknesses whose absence should be verified.
- ***PR.IP-2****: A System Development Life Cycle to manage systems is implemented* — the system development life cycle should include static and dynamic analysis of structural cybersecurity integrity as well as explicit measures of results.
- ***PR.IP-12****: A vulnerability management plan is developed and implemented* — a vulnerability management plan should include remediation actions for explicit cybersecurity weaknesses enumerated in the CISQ measures that exceed the target profile for risk tolerance.
- ***DE.CM-8****: Vulnerability scans are performed* — Source code of critical systems should be scanned periodically to determine the type, number, location, density, and other characteristics of the cybersecurity weaknesses contained in the CISQ measures.

References

(1) Object Management Group (2015). *Automated Source Code Security Measure*. http://www.omg.org/spec/ASCSM/1.0/PDF

(2) Object Management Group (2015). *Automated Source Code Reliability Measure*. http://www.omg.org/spec/ASCRM/1.0/PDF

Submitter on behalf of CISQ

Dr. Bill Curtis
Executive Director, CISQ
director@it-cisq.org