Carnegie Mellon University
Software Engineering Institute
CERT Division

SCALe

Evaluating Source Code for Adherence to Secure Coding Standards

EXPERIENCE SHOWS THAT MOST SOFTWARE CONTAINS CODING FLAWS THAT LEAD TO VULNERABILITIES. Static analysis tools produce many alerts, some of which are false positives, that engineers painstakingly examine to find legitimate flaws. Researchers in the Software Engineering Institute's (SEI's) CERT Division have developed Source Code Analysis Laboratory (SCALe) to help analysts be more efficient and effective at auditing source code for security flaws.

What is SCALe?

SCALe is a set of tools and processes developed by the SEI to help organizations address common problems in auditing code.

While every static analyzer can be used to look for source code flaws that might be exploitable, analyzers have different strengths. One critical issue is that no analyzer finds everything. Using multiple tools can increase coverage to find more security flaws, but many static analyzers provide their own interface for managing their own alerts, complicating attempts to use multiple analyzers on the same codebase. Further complicating this is that some static analysis checkers have high false-positive rates in some tools. A second critical issue is that all static analysis tools produce some false positives, unless the static analysis tool incorporates a formal prover (which is very rare and even when present, is only as good as the formally stated claims). These false positives lead to the need to adjudicate alerts as true or false. The traditional method to address this issue is manually adjudicating alerts, a process that requires expertise in the coding language, understanding the code flaw taxonomy, and tracing the code (for data flow, control flow, variable types, etc.) to make the adjudication; this is expensive and takes time.

SCALe—which has been used to analyze software for the Department of Defense (DoD), energy delivery systems, medical devices, and more—provides smart methods and tools to automate alert adjudication and related work. For example, we have developed tools that incorporate novel algorithms, DevSecOps integrations, and ways to use artificial intelligence (AI) (e.g., large language models [LLMs] and machine learning [ML]). To increase our tools' impact, some are released cost free and open source. SCALe tools and processes, some of which are detailed below, help analysts and developers to efficiently find and fix code flaws, thereby reducing exploitable bugs and malfunction in code.

SCALe Auditing Framework

The SCALe auditing framework aggregates output from commercial, open source, and experimental analysis tools. It maps alerts about possible code flaws from code analysis tools to code flaw taxonomies of interest (e.g., CERT Secure Coding Rules and Common Weakness Enumeration [CWE]). It provides a graphical interface that an analyst can use to filter, fuse, and prioritize alerts as well as examine code associated with an alert. The analyst can also mark alert adjudications (e.g., true or false) and store or export data for the audit project. Some static analysis tool output formats (including the SARIF standard format) are already integrated with the SCALe tools; the SCALe user manual explains the simple API enabling users to integrate new tools.

We provide the SCALe auditing framework tools to many DoD organizations and some non-DoD organizations for their use in evaluating their source code for adherence to secure coding standards. We provide services to help organizations adopt the SCALe auditing framework to improve their secure development lifecycle practices.

SCALe Research Prototype

We create SCALe research prototypes by adding new, experimental functionality to the SCALe auditing framework and processes. For example, a research project may use different rules for determining which alerts to audit or which alert determination lexicon to use. These prototypes may be distributed to collaborators during a project; we often integrate innovative technologies and processes from the prototypes into SCALe. For example, SEI's SCAIFE research focused on novel use of AI for static analysis, enhancing and integrating with SCALe tools in a modular API-defined framework for continuous integration (CI) systems.

SCALe Conformance Testing

SCALe conformance testing provides organizations with an evaluation of their source code for adherence to secure coding standards. We use the SCALe auditing framework and commercial, open source, and experimental analysis tools to provide this service. For each CERT secure coding standard, the source code for the software is certified at a level of conformance.

The SCALe Conformance Process

Conformance testing motivates organizations to invest in developing conforming systems by testing code against CERT secure coding standards, verifying that the code conforms with those standards, using the CERT seal, and maintaining a certificate registry of conforming systems. When you request SCALe conformance testing, the following process is initiated:

1. You submit your source code for analysis.
2. CERT staff examines the code using analyzer tools.
3. CERT staff validates and summarizes the results.
4. You receive a detailed report of findings to guide your repair of the source code.
5. You address the identified violations and resubmit the repaired code.
6. CERT staff reassesses the code to ensure that you have mitigated all violations properly.
7. Your certification for that version of the product is published in a registry of certified systems.

The CERT SCALe Seal

If CERT SCALe conformance testing determines that your software conforms to a secure coding standard, you may use the CERT SCALe seal. The seal must be specifically tied to the software passing conformance testing and not applied to untested products or the organization. Use of the CERT SCALe seal is contingent upon (1) the organization entering into a service agreement with Carnegie Mellon University and (2) the software being designated by the CERT Division as conforming. With some exceptions, modifications made to software after it is designated as conforming voids the conformance designation.

Related Research

At the SEI, we are conducting research on related topics, including adjudication assisted by AI (LLMs, ML, etc.) and automated program repair. Our research in AI, specifically alert classification and prioritization, is intended to help organizations secure their code more efficiently by using statistical methods to triage and prioritize static analysis alerts. Our research has demonstrated that LLM-assisted adjudication can provide step-by-step reasoning that a human can quickly follow to validate an automated adjudication. We have also used LLMs to provide examples that demonstrate a flaw's existence. Our automated program repair research and development develops automated patches that can be used to eliminate the flaw during development and/or as part of security reviews, resulting in less static analysis alerts.

Get SCALe Software
Scan the QR code with your smartphone camera to access the SCALe software.
insights.sei.cmu.edu/library/scale

Watch SCALe Video
Scan the QR code with your smartphone camera to view a 1-hour video demonstrating many SCALe features and how to use them.
youtube.com/live/Em_GABCmHbk

About the CERT Division

The CERT® Division of Carnegie Mellon University's Software Engineering Institute conducts valued, relevant, and trusted evidence-based research that fortifies the cyber ecosystem and protects national security and prosperity.

Contact Us

CARNEGIE MELLON UNIVERSITY
SOFTWARE ENGINEERING INSTITUTE
4500 FIFTH AVENUE; PITTSBURGH, PA 15213-2612

sei.cmu.edu
412.268.5800 | 888.201.4479
info@sei.cmu.edu