Too Many Issues: Automatically Prioritizing Analyzer Findings by Tracing Security Importance (Summary)

Sven Peldszus [1], Katharina Großer [2], Marco Konersmann [3], Wasja Brunotte [4], Maike Ahrens [4], Kurt Schneider [4], and Jan Jürjens [2,5]

Abstract: Our publication accepted in 2025 at the *ACM Transactions on Software Engineering and Methodology* (TOSEM) concerns the prioritization of issues identified by code-based analyzers. These often find too many potentially security-related issues to address them all. Thus, issues likely to lead to vulnerabilities should be fixed first. Prioritization requires project-specific knowledge, such as quality requirements, security-related decisions, and design, which is not accessible to code analyzers. We present *TraceSEC*, an automated technique for prioritizing issues according to their security-related importance to the project. Its core concept is to incorporate available design artifacts and trace links between them, considering the project context that the code lacks. We reduce issue prioritization to a maximum flow problem and quantify the importance of each issue by the flow from user-defined quality aspects to the issue, i.e., its impact on project-specific security preferences. Our evaluation shows that *TraceSEC* effectively provides automated prioritization and can be tailored to project-specific quality goals. Its prioritization correlates stronger with manual expert prioritization than SonarQube rule severities, and it scales reasonably well for codebases up to 4 M lines of code.

Keywords: Static Code Analysis, Security Issues, Traceability, Issue Prioritization

1 Summary

To help in identifying and fixing potential security-related issues, state-of-the-art static analyzers like SonarQube focus on detecting code smells, or bugs and flaws resembling them. Unfortunately, they show overwhelmingly many issues—too many to address when applied to typical, non-trivial codebases. Developers are overstrained and frequently overlook the few essential issues, i.e, misjudging their importance with respect to the design of the system. Often, static analyzers cannot prioritize issues effectively because they lack information about the project context, which is mostly shaped by requirements and design. Thus, in our paper [Pe25], we investigate the following research question:

"How can project context information be systematically used to prioritize issues reported by code analyzers according to their relevance for the security of the project?"

[1] IT University of Copenhagen, Denmark, svpe@itu.dk, https://orcid.org/0000-0002-2604-0487
[2] University of Koblenz, IST, Germany, grosser@uni-koblenz.de, https://orcid.org/0000-0003-4532-0270; juerjens@uni-koblenz.de, https://orcid.org/0000-0002-8938-0470
[3] ista SE, Essen; Software Engineering, RWTH Aachen; Germany, konersmann@se-rwth.de, https://orcid.org/0000-0003-3452-5772
[4] Leibniz University Hannover, Germany, -, https://orcid.org/0000-0003-4127-6508; -, https://orcid.org/0000-0002-9577-0598; kurt.schneider@inf.uni-hannover.de, https://orcid.org/0000-0002-7456-8323
[5] Fraunhofer ISST, Dortmund, Germany, juerjens@uni-koblenz.de, https://orcid.org/0000-0002-8938-0470

doi:10.18420/se2026-18

We define relevance as the strength of an issue's connection to the project-specific security qualities captured in a quality model [Sc12], i.e., how many of them can be affected by a specific issue and how severely. We base the prioritization on trace links and map it to a *maximum flow problem* [HR55], a graph-theoretical problem to which many connectivity-based optimizations are mapped. Our *TraceSEC* issue prioritization technique leverages project-specific knowledge captured in project artifacts as well as trace links to construct a flow network. On this flow network, it quantifies the connection between issues and project-specific security qualities by calculating the maximum flow between each issue and the quality model. Our tool support comes with a domain-specific language (DSL) that allows arbitrary model-based artifacts to be considered in prioritization and to tailor the prioritization for individual projects.

We evaluated *TraceSEC* on two real-world projects from the healthcare domain, the iTrust electronic health records system and the German Corona Warn App server. We considered requirements, (UML) architecture models, and (Java) program code in our evaluation. Our results show that *TraceSEC* provides effective automated issue prioritization that is closer to manual expert prioritization than prioritization based on SonarQube rule severities, which are commonly used in practice. Specifically, *TraceSEC* is more effective at identifying high-priority issues. Further, it can be tailored to project-specific quality requirements. While it has been proven that the maximum flow problem can be solved in polynomial time, we also show that the construction of the flow network scales reasonably well for codebases of up to 4 million lines of code on a developer's notebook. The initial setup overhead is likely to be recouped after the first automated prioritization, so automated prioritization is likely to be faster than manual prioritization. To this end, *TraceSEC* provides the required tooling for investigating security-related traceability. In general, it enables researchers to examine how security considerations at different development phases correspond with each other. We provide tool providers with an effective technique for prioritizing issues, taking project-specific security considerations into account.

2 Data Availability

A full replication package is available at https://doi.org/10.5281/zenodo.15535226.

Bibliography

[HR55] Harris, T.; Ross, F.: Fundamentals of a Method for Evaluating Rail Net Capacities, tech. rep. RM-1573, The RAND Corporation, 1955.

[Pe25] Peldszus, S. et al.: Too Many Issues: Automatically Prioritizing Analyzer Findings by Tracing Security Importance. Transactions on Software Engineering and Methodology (TOSEM), 2025, https://doi.org/10.1145/3744708.

[Sc12] Schneider, K.: Abenteuer Softwarequalität, Grundlagen und Verfahren für Qualitätssicherung und Qualitätsmanagement. In. 2nd ed., dpunkt. verlag, chap. Qualitätsaspekte, -Anforderungen und Qualitätsmodelle, 2012, ISBN 978-3-89864-784-7.