Thesis for the Degree of Doctor of Philosophy

Efficiency and Automation in Threat Analysis of Software Systems

Katja Tuma

Presentation:
January 11th, 2021, 14:15
Online and room 473, Jupiter Building
Hörselgången 5,
University of Gothenburg, Campus Lindholmen

Discussion leader:
Prof. Maritta Heisel
University of Duisburg-Essen, Germany

The thesis is available at:
Department of Computer Science & Engineering
University of Gothenburg
Gothenburg, Sweden, 2021

Phone: +46 (0)31 77 26 814

Abstract

Context: Security is a growing concern in many organizations. Industries developing software systems plan for security early-on to minimize expensive code refactorings after deployment. In the design phase, teams of experts routinely analyze the system architecture and design to find potential security threats and flaws. After the system is implemented, the source code is often inspected to determine its compliance with the intended functionalities.

Objective: The goal of this thesis is to improve on the performance of security design analysis techniques (in the design and implementation phases) and support practitioners with automation and tool support.

Method: We conducted empirical studies for building an in-depth understanding of existing threat analysis techniques (Systematic Literature Review, controlled experiments). We also conducted empirical case studies with industrial participants to validate our attempt at improving the performance of one technique. Further, we validated our proposal for automating the inspection of security design flaws by organizing workshops with participants (under controlled conditions) and subsequent performance analysis. Finally, we relied on a series of experimental evaluations for assessing the quality of the proposed approach for automating security compliance checks.

Findings: We found that the eSTRIDE approach can help focus the analysis and produce twice as many high-priority threats in the same time frame. We also found that reasoning about security in an automated fashion requires extending the existing notations with more precise security information. In a formal setting, minimal model extensions for doing so include security contracts for system nodes handling sensitive information. The formally-based analysis can to some extent provide completeness guarantees. For a graph-based detection of flaws, minimal required model extensions include data types and security solutions. In such a setting, the automated analysis can help in reducing the number of overlooked security flaws. Finally, we suggested to define a correspondence mapping between the design model elements and implemented constructs. We found that such a mapping is a key enabler for automatically checking the security compliance of the implemented system with the intended design. The key for achieving this is two-fold. First, a heuristics-based search is paramount to limit the manual effort that is required to define the mapping. Second, it is important to analyze implemented data flows and compare them to the data flows stipulated by the design.

Keywords

Secure Software Design, Threat Analysis (Modeling), Automation, Security Compliance