Software Engineering Seminar

Symbolic Side-Channel Analysis

Description

Symbolic Execution is a program analysis, which uses symbolic values (instead of concrete ones) to execute the program under test. The symbolic analysis leads to a systematic exploration of the search space, which can be directed by various heuristics. Recent work like CoCo-Channel by Brennan et al. [1] and Malacaria et al. [4] leverages symbolic execution for the detection of side-channel vulnerabilities. For example, CoCo-Channel [1] provides a compositional generation of symbolic path cost expressions, which is used to find imbalanced program paths. Adversarial Symbolic Execution by Guo et al. [3] focuses specifically on concurrency-related cache timing leaks. The CacheFix framework [2] makes a step further by also attempting to fix a potential side-channel vulnerability: It either verifies the cache side-channel freedom of the program or synthesizes a series of patches to ensure cache side-channel freedom during program execution.

The goal of this seminar topic is to collect the current research directions in symbolic side-channel analysis. Therefore, it is necessary to perform an initial literature analysis based on the provided publications. The student should examine and discuss the approaches given in the papers and compare them to each other and to similar existing techniques. Additionally, the student is asked to provide a critical discussion of the current research directions, which should also include an outlook for possible future work.

References


Contacts

Yannic Noller (noller@informatik.hu-berlin.de)
Software Engineering Group
Institut für Informatik
Humboldt-Universität zu Berlin