STAnalyzer: A simple static analysis tool for detecting cache-timing leakages

Alexander Schaub Sylvain Guilley Olivier Rioul

April 30, 2019

Abstract

Cache-timing attacks are a class of side-channel attacks that target software implementations of cryptographic algorithms. If the cache-access pattern of the implementation depends on sensitive information, then a cache-timing attack can retrieve this information, which can potentially lead to a secret-key recovery. Implementations which branch on conditions depending on sensitive information, or that access memory locations whose address depend on sensitive information, are potentially vulnerable to such attacks.

This paper presents an algorithm for verifying that a program, implemented in the C language, is free from cache-timing leakages. It consists in computing the dependencies of all the variables used in the program, and listing all sensible values that leak due to branching and memory accesses. An implementation of this algorithm, STAnalyzer, is also provided. It allows to flag sensitive values, and those are tracked across computations, function calls, etc. Therefore, only leakages of sensitive values are reported. Because the algorithm runs directly on an abstract syntaxic tree (AST) of the C program, the output is straightforward to interpret: dependencies between C variables are reported, as well as the stack of function calls and instructions that lead to the leakage of sensitive values.