Abstract

Hassan Shokry El-Dib

Formal verification of software countermeasures against side-channel attacks

A common strategy for designing countermeasures against power-analysis-based side-channel attacks is using random masking techniques to remove the statistical dependency between sensitive data and side-channel emissions. However, this process is both labor intensive and error prone and, currently, there is a lack of automated tools to formally assess how secure a countermeasure really is. We propose the first SMT-solver-based method for formally verifying the security of a masking countermeasure against such attacks. In addition to checking whether the sensitive data are masked by random variables, we also check whether they are perfectly masked, that is, whether the intermediate computation results in the implementation of a cryptographic algorithm are independent of the secret key. We encode this verification problem using a series of quantifier-free first-order logic formulas, whose satisfiability can be decided by an off-the-shelf SMT solver. We have implemented the proposed method in a software verification tool based on the LLVM compiler frontend and the Yices SMT solver. Our experiments on a set of recently proposed masking countermeasures for cryptographic algorithms such as AES and MAC-Keccak show the method is both effective in detecting power side-channel leaks and scalable for practical use.