Contactless Side Channel Based Disassembling
A time shift resilient machine learning approach

Tobias Kovats, Georg Merzdovnik, Christian Kudera

Problem & Motivation

Side Channel Based Disassembling
Refers to the process of recovering the source code running on a device from one of its side channels, e.g., its power consumption or electromagnetic emanation.

This can be divided into two phases:
▶ Profiling: Behaviour of the chosen side channel is learned during execution of corresponding instructions.
▶ Attacking: Unknown instruction traces are reconstructed from side channel leakage.

Problems
Side channel obfuscation is predominantly caused by
▶ Non-consecutive execution: Instruction phases overlap due to processor pipeline.
▶ Register value dependencies: Side channel behaviour depends not only on instruction, but also on register values. [1]
▶ Synchronization: Attacker needs to be in-phase with the target device.
▶ Background noise: Peripherals and other unwanted radiation sources cause interference.

Our Approach

Contribution: We incorporate previously proposed approaches from secret key recovery [2] and image recognition [3] in building an electromagnetic side channel based disassembler with enhanced time shift resilience to eliminate the need for synchronization.

▶ Profiling: To ensure independence of all other factors except the instruction itself, all registers are initialized with random values, and two random instructions are inserted before and after the target instruction, each working with random registers.
▶ Data Augmentation: Several data augmentation methods are evaluated. The following aims to enhance time-shift resilience: Several trace recordings are concatenated; a window of length $T$ and a uniform random offset $T'$ from $t_0$ cut out.
▶ Attacking: For performance evaluation, random instruction traces are generated and reconstructed.
▶ Sliding Window: A window of length $T$ slides over the full trace using a predefined overlap. The resulting windows are fed into the classifier.
▶ Model: A one-dimensional convolutional neural network similar to the VGG16 implementation is used for classification.

Application

▶ Reverse Engineering: IP theft, localization of critical code segments, code recognition and code flow analysis
▶ Monitoring: Detection of deviations from regular execution, realtime code execution tracking
▶ Complex Embedded Systems: Non-invasive application to systems that don’t allow time synchronization, e.g., programmable logic controllers

Preliminary Conclusions*

▶ Complex model architectures and high computational resources are necessary to cope with side channel obfuscations.
▶ Leakage cartography of local electromagnetic emanations shows high gradient.
▶ Promising results using a reduced instructions set have already been obtained.

*This is a work in progress and final conclusions are to follow.