USENIX Security '25 Artifact Appendix: Reality Check on Side-Channels: Lessons learnt from breaking AES on ARM Cortex-A72 processor with Out-of-Order Execution

Harishma Boyapally[1,2], Dirmanto Jap[1,2], Qianmei Wu[3,4], Fan Zhang[3], Shivam Bhasin[1,2]*
[1]Temasek Laboratories, Nanyang Technological University, Singapore.
[2]National integrated Centre For Evaluation, Nanyang Technological University, Singapore.
[3]School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore.
[4]School of Cyber Science and Technology, College of Computer Science and Technology, Zhejiang University, China
Email:{harishma.boyapally, djap, sbhasin}@ntu.edu.sg, {qianmei, fanzhang}@zju.edu.cn

A Artifact Appendix

A.1 Abstract

This artifact provides the code and dataset for replicating the side-channel analysis (SCA) attack on an AES C implementation running on a Raspberry Pi 4B, specifically targeting its ARM Cortex-A72 processor with out-of-order execution. The work evaluates the practical effort required for key recovery attacks under various threat models, highlighting challenges like noise, jitter, OS interference, and out-of-order execution. The accompanying Python scripts enable training and testing of deep learning models for SCA, and plotting of results, as reported in the WOOT'25 paper.

A.2 Description & Requirements

This section details the environment necessary to reproduce the experimental setup used to generate the results.

A.2.1 Security, privacy, and ethical concerns

None

A.2.2 How to access

The Artifact can be found at:
https://doi.org/10.5281/zenodo.15524301.

*This research is supported by the National Research Foundation, Singapore, and Cyber Security Agency of Singapore under its National Cybersecurity Research & Development Programme (Development of Secured Components & Systems in Emerging Technologies through Hardware & Software Evaluation NRF-NCR25-DeSNTU-0001). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the view of National Research Foundation, Singapore and Cyber Security Agency of Singapore.

A.2.3 Hardware dependencies

The experiments were conducted on a workstation with the following specifications:

- **CPU:** Intel(R) Xeon(R) W-2123 CPU @ 3.60GHz
- **GPU:** NVIDIA Quadro P6000 GPU

The target device on which the side-channel traces were collected has the following specifications:

- **Device:** Raspberry Pi 4B board with a Broadcom BCM2711 SoC.
- **Processor:** Quad-core out-of-order ARM Cortex-A72 (ARMv8, 64-bit) processor, operating at a maximum frequency of 1.8 GHz.
- **Operating System:** Bullseye version of the Raspberry Pi OS (Debian Linux).

A.2.4 Software dependencies

The following operating system and software packages are required to evaluate this artifact:

- **Operating System:** Linux (specifically, the tests were conducted on a Linux distribution compatible with PyTorch 1.9.0).
- **Python Libraries:**
  - h5py (HDF5 library): Required for handling the dataset in HDF5 format.
  - numpy (Scientific computing library): Essential for numerical operations and data manipulation.
  - matplotlib (Plotting library): Used for generating figures and plots.

– pytorch (Deep learning library): The core framework for training and testing the deep learning models. The tests were specifically conducted on version 1.9.0.

These libraries can typically be installed using pip.

A.2.5 Benchmarks

The primary dataset used for the experiments is F_trace_raw.h5. This dataset comprises two main parts:

- **"Profiling_traces":** Contains traces used for training models.
- **"Attack_traces":** Contains traces used for testing the models and evaluating attack effectiveness.

Each part contains:

- **"traces":** A 2D dataset with 7,000 sample points per trace. There are 150,000 traces for training and 50,000 traces for testing.
- **"metadata":** Includes "plaintext", "key", and "label". The "label" represents the output of the first round S-box, derived from the "plaintext" and "key".

More detailed information on how to load and process this data can be found in dataloader.py within the src folder of the repository.

A.3 Set-up

A.3.1 Installation

To prepare the environment for evaluating this artifact, follow these steps:

- Clone the repository: Obtain the repository containing the artifact's files.
- Install Python and pip: Ensure Python 3.x and its package installer pip are installed on your system.
- Install required libraries: Open a terminal and run the following commands to install the necessary Python libraries:

```
pip install h5py numpy matplotlib "torch==1.9.0"
```

(Note: PyTorch installation might vary based on your CUDA version or if you intend to use CPU-only. Refer to the official PyTorch website for specific installation instructions if torch==1.9.0 fails.)

After these steps, the environment should be ready to run the artifact's functionalities.

A.3.2 Basic Test

To perform a simple functionality test and verify that the setup is correct, you can run the main_challenge.py script.

- Navigate to the repository directory: Open your terminal and change the directory to where you cloned the repository.
- Run the main challenge script:

```
python main_challenge.py
```

Expected Successful Output: The main_challenge.py script will execute the training and testing of the CNN model with default settings (ID leakage model, random search for 100 iterations for hyperparameter tuning). Upon completion, the function will return the index of the best-performing model found during the random search. A successful output will indicate that the script ran without errors and provided a valid index, confirming that all required software components are functioning correctly and can process the dataset.

A.4 Version

Based on the LaTeX template for Artifact Evaluation V20231005. Submission, reviewing and badging methodology followed for the evaluation of this artifact can be found at https://secartifacts.github.io/usenixsec2025/.